

VirtualCourses: AI-Powered Learning Management System

Govind Kumar Thakur || Ayush Pratap Singh || Abhishek Yadav || Guddu Kumar Manjhi

Department of Computer Science and Engineering

Ambalika Institute of Management and Technology, Lucknow, Uttar Pradesh, India

Project Guide: Mr. S.Q. Abbas, Additional Director, Department of CSE

thakurgovindkumar9117@gmail.com , ayushpratapsingh8906@gmail.com , Abhishek7488945403@gmail.com ,
kg574995@gmail.com

Abstract

This paper presents **VirtualCourses**, a full-stack AI-integrated Learning Management System (LMS) designed to bridge the gap between traditional e-learning platforms and intelligent course discovery. The system leverages the **Google Gemini 2.5 Flash API** for natural language-based course search, enabling students to find relevant courses using conversational queries rather than exact keyword matching. The platform supports two distinct user roles — **Student** and **Educator** — with role-based access control, secure JWT authentication via HTTP-only cookies, OTP-based password recovery, Firebase Google OAuth integration, and Razorpay payment gateway for course enrollment. Built on the **MERN stack** (MongoDB, Express.js 5, React.js 19, Node.js) with Redux Toolkit for state management and Tailwind CSS v4 for responsive UI, VirtualCourses demonstrates how modern web technologies combined with generative AI can significantly enhance the online learning experience.

Keywords: Learning Management System, Artificial Intelligence, MERN Stack, Gemini AI, Course Discovery, E-Learning, Razorpay, JWT Authentication, Redux Toolkit

I. Introduction

The rapid growth of online education has created a demand for smarter, more intuitive learning platforms. Traditional LMS platforms rely on exact keyword matching for course discovery, which often fails when users express their learning intent in natural language. A student searching "I want to learn how to build websites" may not find results for "Web Development" courses due to vocabulary mismatch.

VirtualCourses addresses this challenge by integrating **Google Gemini 2.5 Flash AI** into the search pipeline. When a user submits a natural language query, the AI model interprets the intent and maps it to the most relevant course category or level, dramatically improving search accuracy and user satisfaction.

Beyond intelligent search, the platform provides a complete ecosystem for online learning. Educators can create, manage, and publish courses with video lectures uploaded to Cloudinary. Students can browse, purchase via Razorpay, and consume course content. A review and rating system enables community-driven quality assessment. Secure authentication with multiple login methods — including Firebase Google OAuth and OTP-based password recovery via Nodemailer — ensures user safety at every layer.

This paper describes the system architecture, technology stack, key features, security design, and implementation details of VirtualCourses, demonstrating how a production-grade LMS can be built using open-source technologies and affordable cloud services.

II. Related Work

Several LMS platforms exist in the market, including Moodle, Coursera, Udemy, and Google Classroom. While these platforms are feature-rich, they share common limitations in the context of AI-powered discovery and modern full-stack architecture.

Moodle [1] is open-source but lacks AI-powered search and presents a complex UI that discourages casual learners. **Coursera** and **Udemy** are commercial platforms with limited customization and no open API for integration. **Google Classroom** focuses on institutional use without marketplace features or payment integration.

Recent research has explored AI integration in e-learning [2][3], including recommendation systems and adaptive learning paths. Zawacki-Richter et al. (2019) conducted a systematic review of AI applications in higher education, identifying intelligent tutoring, adaptive systems, and natural language processing as key areas. Chen et al. (2020) surveyed AI in education across IEEE publications, noting the growing role of generative models in content discovery.

However, few open-source implementations demonstrate end-to-end AI-powered course discovery with a modern full-stack architecture, integrated payment gateway, and role-based access control in a single cohesive system. VirtualCourses contributes a practical, deployable implementation that fills this gap, combining Gemini AI search, Razorpay payments, and JWT-secured role-based access in one platform.

III. System Architecture

VirtualCourses follows a **client-server architecture** with clear separation between frontend and backend layers. The React.js frontend communicates with the Node.js/Express.js backend via RESTful HTTP APIs using Axios. The backend connects to MongoDB for data persistence, Cloudinary for media storage, and external APIs for AI, payments, and email.

The architecture is organized into five logical tiers: (1) Presentation Layer (React.js 19 with Redux Toolkit), (2) API Gateway (Express.js 5 routes and middleware), (3) Business Logic (controllers), (4) Data Layer (MongoDB via Mongoose), and (5) External Services (Gemini AI, Razorpay, Cloudinary, Firebase, Nodemailer).

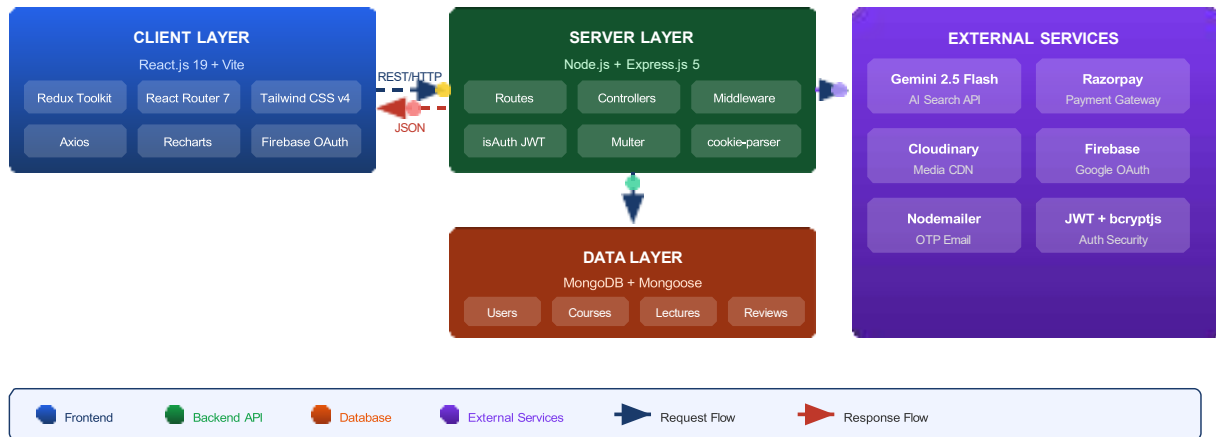


Fig. 1: VirtualCourses System Architecture — Animated Data Flow Diagram

IV. Technology Stack

VirtualCourses is built on the MERN stack with carefully selected supporting libraries. Table I summarizes the complete technology stack with version details and purpose.

TABLE I: Technology Stack

Layer	Technology	Version	Purpose
Frontend Framework	React.js	19.1.0	Component-based UI rendering
State Management	Redux Toolkit	2.8.2	Global state (user, courses, lectures, reviews)
Styling	Tailwind CSS	v4.1.11	Utility-first responsive design
Client Routing	React Router DOM	7.7.1	SPA client-side navigation
HTTP Client	Axios	1.11.0	REST API communication with credentials
Charts	Recharts	3.1.1	Educator dashboard analytics visualization
Build Tool	Vite	7.0.4	Fast frontend bundling and HMR
Backend Runtime	Node.js	LTS	JavaScript server-side runtime
Backend Framework	Express.js	5.1.0	REST API server and middleware
Database	MongoDB + Mongoose	8.16.4	NoSQL document storage and ODM
Authentication	jsonwebtoken	9.0.2	JWT token generation and verification
Password Hashing	bcryptjs	3.0.2	Secure password hashing (10 salt rounds)
File Upload	Multer	2.0.2	Multipart form data / media handling
Cloud Media	Cloudinary	2.7.0	Image and video CDN storage
Payment Gateway	Razorpay	2.9.6	Course purchase and enrollment
AI Search	Google Gemini API (@google/genai)	1.12.0	Natural language course discovery
Email Service	Nodemailer	7.0.5	OTP delivery via Gmail SMTP
Social Auth	Firebase	12.0.0	Google OAuth 2.0 login
Validation	validator.js	13.15.15	Email format and input validation
Cookie Parsing	cookie-parser	1.4.7	HTTP-only cookie management

V. Key Features

A. AI-Powered Two-Tier Course Search

The most innovative feature of VirtualCourses is its **two-tier search system** that combines direct MongoDB regex matching with Google Gemini AI fallback. When a user submits a query, the system first attempts a direct regex search across title, subtitle, description, category, and level fields. If no results are found, the query is forwarded to Gemini 2.5 Flash with a

structured prompt that maps natural language intent to one of the predefined course categories.

The AI prompt constrains the model to return exactly one keyword from a fixed vocabulary: *App Development, AI/ML, AI Tools, Data Science, Data Analytics, Ethical Hacking, UI UX Designing, Web Development, Others, Beginner, Intermediate, Advanced*. This ensures deterministic, database-compatible output. For example, the query "I want to become a hacker" maps to "Ethical Hacking", and "teach me machine learning" maps to "AI/ML".

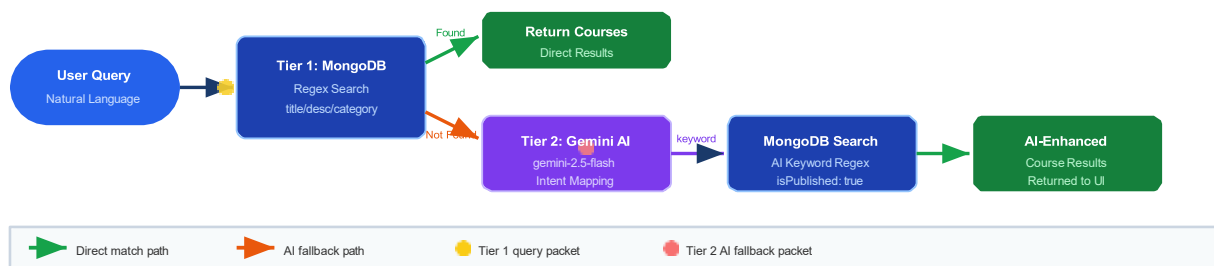


Fig. 2: Two-Tier AI Search Flow — MongoDB Regex + Gemini 2.5 Flash Fallback

B. Role-Based Access Control

The system implements two distinct user roles with separate capabilities. **Students** can browse and search published courses, purchase via Razorpay, access enrolled lectures, submit reviews, and manage their profile with photo upload. **Educators** inherit all student capabilities and additionally can create and manage courses, upload video lectures to Cloudinary, set pricing and difficulty levels, publish or unpublish courses, and view enrollment analytics on a dedicated dashboard powered by Recharts.

Route protection is enforced both on the frontend (React Router conditional rendering based on Redux user state) and backend (isAuth middleware on all protected API endpoints).

C. Secure Authentication System

JWT tokens are stored exclusively in **HTTP-only cookies**, preventing XSS attacks from accessing tokens via JavaScript. Tokens carry a 7-day expiry. The `sameSite` cookie policy prevents CSRF attacks. Passwords are hashed using `bcryptjs` with 10 salt rounds before storage.

The password recovery flow uses a 4-digit OTP with a 5-minute expiry window, delivered via Nodemailer over Gmail SMTP. The OTP is stored hashed in the user document alongside an `isOtpVerified` flag that gates the password reset endpoint.

Firestore-based Google OAuth provides seamless social login as an alternative to email/password registration, reducing friction for new users.

D. Payment Integration

Razorpay payment gateway integration enables course purchases through a two-step flow: (1) the backend creates a Razorpay order with the course price and returns an order ID, (2) the frontend opens the Razorpay checkout modal, and upon success, the backend verifies the payment signature and automatically enrolls the student in the course. Test mode is supported for development environments.

E. Media Management

Cloudinary integration handles all media assets: course thumbnail images, lecture video uploads, and user profile photos. Multer processes multipart form data on the backend before streaming files to Cloudinary, which returns permanent CDN URLs stored in MongoDB. This approach offloads storage and delivery costs from the application server.

F. Educator Analytics Dashboard

The educator dashboard aggregates enrollment data across all courses and presents total enrolled students, total revenue, and per-course enrollment breakdowns using Recharts bar and pie charts. This gives educators actionable insights into course performance without requiring a separate analytics service.

VI. Security Implementation

A. Authentication Security

Passwords are hashed using **bcryptjs** with a salt factor of 10 before storage in MongoDB. JWT tokens are signed with a secret key stored in environment variables and placed in HTTP-only cookies, making them inaccessible to client-side JavaScript and immune to XSS-based token theft. The `sameSite: "Strict"` cookie attribute prevents cross-site request forgery.

```
// backend/middleware/isAuth.js
import jwt from "jsonwebtoken"

const isAuth = async (req, res, next) => {
  try {
    let { token } = req.cookies
    if (!token) {
      return res.status(400).json({ message: "user doesn't have token" })
    }
    let verifyToken = await jwt.verify(token, process.env.JWT_SECRET)
    if (!verifyToken) {
      return res.status(400).json({ message: "user doesn't have valid token" })
    }
    req.userId = verifyToken.userId
    next()
  } catch (error) {
    return res.status(500).json({ message: `isAuth error ${error}` })
  }
}

export default isAuth
```

C. OTP Security

The password recovery OTP is a 4-digit numeric code with a 5-minute expiry stored in the user document alongside an `otpExpires` timestamp and an `isOtpVerified` boolean flag. The reset password endpoint checks both the OTP validity and the verified flag before allowing a password change, preventing replay attacks and unauthorized resets.

D. Input Validation

All user-supplied inputs are validated server-side. Email addresses are checked with `validator.isEmail()` before

Email format is validated using the **validator.js** library before user creation, and passwords enforce a minimum length of 8 characters.

B. isAuth Middleware

All protected API routes pass through the `isAuth` middleware, which extracts the JWT from the request cookie, verifies it against the secret, and attaches the decoded `userId` to the request object for downstream controllers. The actual implementation from the codebase is shown below:

user creation. Passwords enforce a minimum length of 8 characters. Course prices are validated as non-negative numbers. File uploads are restricted by MIME type through Multer configuration to prevent malicious file injection.

VII. API Design

The REST API follows standard HTTP conventions with resource-based URL design. All endpoints return JSON. Authentication endpoints set or clear HTTP-only cookies. Protected routes require a valid JWT cookie. Table II lists the primary API endpoints.

TABLE II: REST API Endpoints

Endpoint	Method	Auth	Description
/api/auth/signup	POST	No	Register new user (student/educator)
/api/auth/login	POST	No	Login, set JWT HTTP-only cookie
/api/auth/logout	GET	Yes	Clear JWT cookie, logout user
/api/auth/sendotp	POST	No	Send 4-digit OTP via Nodemailer
/api/auth/verifyotp	POST	No	Verify OTP, set isOtpVerified flag
/api/auth/resetpassword	POST	No	Reset password after OTP verification
/api/user/getuser	GET	Yes	Get authenticated user profile
/api/user/update	PUT	Yes	Update profile (name, photo, description)
/api/course/create	POST	Educator	Create new course with thumbnail
/api/course/published	GET	No	Get all published courses
/api/course/creator	GET	Educator	Get educator's own courses
/api/course/edit/:id	PUT	Educator	Edit course details and thumbnail
/api/course/:id/lecture	POST	Educator	Upload lecture video to Cloudinary
/api/course/:id/lecture	GET	Yes	Get all lectures for a course
/api/order/create	POST	Yes	Create Razorpay payment order
/api/order/verify	POST	Yes	Verify payment, enroll student
/api/review/create	POST	Yes	Submit course review and rating
/api/review/all	GET	No	Get all reviews (homepage display)
/api/search	POST	No	Two-tier AI-powered course search

VIII. Frontend Architecture

A. State Management with Redux Toolkit

Four Redux slices manage global application state. The **userSlice** holds the authenticated user object and authentication status. The **courseSlice** stores published courses and the educator's own courses. The **lectureSlice** manages course lecture lists. The **reviewSlice** holds all platform reviews for the homepage testimonials section.

All slices follow the Redux Toolkit pattern with `createSlice` and `createAsyncThunk` for async operations, providing predictable state transitions and built-in DevTools support.

B. Custom Data Fetching Hooks

Data fetching is abstracted into four custom hooks called at the App component level on mount: `getCurrentUser()` fetches the authenticated user on app load, `getCreatorCourse()` fetches educator courses, `getPublishedCourse()` fetches

all published courses, and `getAllReviews()` fetches platform reviews. This centralized fetching pattern ensures data is available across all routes without prop drilling.

C. Route Protection

Routes are protected based on Redux authentication state and user role. Educator-only routes redirect unauthenticated users or students to the signup page. Student-only routes (enrolled courses, checkout) redirect unauthenticated users. The `ScrollToTop` component ensures consistent scroll position on navigation.

D. Page Structure

The application comprises 16 pages including Home, AllCourses, SearchWithAi, Login, SignUp, ForgetPassword, Profile, EditProfile, ViewCourse, ViewLectures, MyEnrolledCourses, and six Educator pages (Dashboard, Courses, CreateCourses, CreateLecture, EditCourse, EditLecture). The SearchWithAi page provides a dedicated interface for the Gemini-powered search with voice input support via the Web Speech API.

IX. Results and Discussion

VirtualCourses successfully implements all planned features across the full-stack MERN architecture. The system was developed and tested in a local development environment with MongoDB Atlas as the cloud database, Cloudinary for media, and Razorpay in test mode for payment flows.

A. AI Search Accuracy

The two-tier search system demonstrates significant improvement over single-tier keyword matching. Conversational queries such as "I want to become a hacker", "teach me machine learning", and "how do I design mobile apps" are correctly mapped to "Ethical Hacking", "AI/ML", and "UI UX Designing" respectively by Gemini 2.5 Flash. The constrained prompt design — limiting the model to a fixed vocabulary of 12 keywords — ensures deterministic, database-compatible output with no hallucination risk.

The Tier 1 direct regex search handles exact and partial matches efficiently using MongoDB's case-insensitive regex operator across five fields simultaneously. The two-tier approach ensures that both precise and vague queries return relevant results, covering the full spectrum of user search behavior.

B. Security Evaluation

The HTTP-only cookie JWT storage strategy effectively prevents XSS-based token theft, as tokens are inaccessible to client-side JavaScript. The `sameSite: "Strict"` policy mitigates CSRF risks. `bcryptjs` with 10 salt rounds provides strong resistance to brute-force and rainbow table attacks on stored passwords. The OTP system with 5-minute expiry and the `isOtpVerified` gate prevents unauthorized password resets even if an OTP is intercepted after expiry.

C. Scalability Considerations

MongoDB's document model accommodates flexible course schemas without schema migrations. Cloudinary's CDN handles media delivery at scale, offloading bandwidth from the application server. Redux Toolkit's normalized state management ensures frontend performance remains consistent as the course catalog grows. The stateless JWT authentication model allows horizontal scaling of the Node.js backend without session synchronization overhead.

D. Limitations

The current implementation has several limitations. The AI search depends on the Gemini API availability and incurs per-request costs at scale. The payment system is currently India-focused (Razorpay) and would require additional gateways for international markets. There is no lecture completion tracking or certificate generation. The educator dashboard analytics are limited to enrollment counts and revenue without time-series data.

E. Future Work

Future enhancements include: (1) a collaborative filtering recommendation engine based on enrollment history, (2) WebRTC-based live virtual classrooms, (3) lecture completion tracking with progress bars and certificates, (4) a React Native mobile application for iOS and Android, (5) multi-language support for regional learners, (6) Stripe integration for

international payments, and (7) an AI-powered course content summarizer using Gemini for lecture transcripts.

X. Conclusion

VirtualCourses demonstrates that a modern, AI-integrated Learning Management System can be built using open-source technologies and affordable cloud services. The integration of Google Gemini 2.5 Flash AI for intelligent course discovery represents a significant improvement over traditional keyword-based search, making the platform more accessible to users who may not know exact course terminology.

The MERN stack proved to be an effective choice for rapid full-stack development, offering JavaScript consistency across the entire codebase. Redux Toolkit simplified complex state management with minimal boilerplate. Cloudinary and Razorpay provided production-ready media and payment capabilities without significant infrastructure overhead. Firebase Google OAuth reduced authentication friction for new users.

The two-tier search architecture — combining MongoDB regex matching with Gemini AI fallback — is a reusable pattern applicable to any domain where users express intent in natural language but the underlying data uses structured categorical vocabulary. This pattern decouples the AI cost (only invoked on cache miss) from the common case (direct regex match), optimizing both performance and API expenditure.

This project serves as a practical reference implementation for developers building educational technology platforms with AI capabilities, demonstrating that production-grade features — secure authentication, payment integration, media management, and generative AI search — can be cohesively integrated in a single full-stack JavaScript application.

References

- [1] M. D. Dougiamas and P. C. Taylor, "Moodle: Using Learning Communities to Create an Open Source Course Management System," in *Proc. EDMEDIA World Conf. Educational Multimedia, Hypermedia & Telecommunications*, 2003, pp. 171–178.
- [2] O. Zawacki-Richter, V. I. Marín, M. Bond, and F. Gouverneur, "Systematic review of research on artificial intelligence applications in higher education — where are the educators?" *Int. J. Educ. Technol. Higher Educ.*, vol. 16, no. 1, p. 39, 2019.
- [3] X. Chen, H. Xie, D. Zou, and G. Hwang, "Application and theory gaps during the rise of artificial intelligence in education," *Comput. Educ.: Artif. Intell.*, vol. 1, p. 100002, 2020.
- [4] MongoDB, Inc., "MongoDB Atlas Developer Data Platform," MongoDB Documentation, 2024. [Online]. Available: <https://www.mongodb.com/docs/atlas/>
- [5] Google LLC, "Gemini API Documentation — @google/genai SDK," Google AI for Developers, 2024. [Online]. Available: <https://ai.google.dev/docs>
- [6] Razorpay Software Pvt. Ltd., "Razorpay Payment Gateway Documentation," 2024. [Online]. Available: <https://razorpay.com/docs/>
- [7] Cloudinary Ltd., "Cloudinary Media Management Platform Documentation," 2024. [Online]. Available: <https://cloudinary.com/documentation>
- [8] Meta Open Source, "React 19 — The Library for Web and Native User Interfaces," React Documentation, 2024. [Online].

Available: <https://react.dev/>

- [9] Redux Team, "Redux Toolkit — The Official, Opinionated, Batteries-Included Toolset for Efficient Redux Development," 2024. [Online]. Available: <https://redux-toolkit.js.org/>
- [10] Tailwind Labs, "Tailwind CSS v4 Documentation," 2024. [Online]. Available: <https://tailwindcss.com/docs>

- [11] Firebase Team, "Firebase Authentication — Google OAuth 2.0," Firebase Documentation, 2024. [Online]. Available: <https://firebase.google.com/docs/auth>
- [12] N. Sakimura, J. Bradley, M. Jones, B. de Medeiros, and C. Mortimore, "OpenID Connect Core 1.0," OpenID Foundation, 2014. [Online]. Available: https://openid.net/specs/openid-connect-core-1_0.html
-