Digital College Forms Management System: A Web-Based Platform for Automated Institutional Documentation

Dr Krishna Anneboina, Assistant Professor, Department of Computer Science and Engineering (Internet of Things), Guru Nanak Institutions Technical Campus

N Jaya Prakash Reddy, Department of Computer Science and Engineering (Internet of Things),

Guru Nanak Institutions Technical Campus, 22-6944,22wj1a6944@gniindia.org
Nenavath Vijay Devgan, Department of Computer Science and Engineering (Internet of Things),
Guru Nanak Institutions Technical Campus, 22-6947,22wj1a6947@gniindia.org
U Sandeep Kumar Reddy, Department of Computer Science and Engineering (Internet of Things),
Guru Nanak Institutions Technical Campus, 22-6960,22wj1a6960@gniindia.org
G Pavan Kumar, Department of Computer Science and Engineering (Internet of Things),
Guru Nanak Institutions Technical Campus, 23-6903,23wj5a6903@gniindia.org

---

Abstract - — Administrative workflows in educational institutions continue to rely heavily on paper-based procedures, causing significant delays, inconsistencies, and resource wastage. This paper introduces EduForms, an intelligent, web-based institutional forms management platform built on the MERN stack — MongoDB, Express.js, React.js, and Node.js. The platform eliminates manual intervention through a combination of automated multi-level approval chains, role-based access control across eight distinct user roles, and intelligent AI-assisted form discovery. A Two-Factor Authentication mechanism utilizing One-Time Passwords enhances login security. An embedded AI assistant called EduBot allows users to locate appropriate forms through natural language interaction, while a universal voice-based form-fill module enables field population through spoken input — supporting all form types and fields. A smart approval automation engine evaluates submissions against predefined rules and automatically approves routine requests, while a priority-based routing mechanism escalates urgent applications directly to higher authorities. A predictive analytics widget presents approval probability estimates derived from historical submission data. The system supports forty-nine form templates spanning eight role-specific portals and includes a contextual feedback mechanism and a comprehensive exam branch management portal. Experimental evaluation confirms measurable reductions in processing time, improved submission accuracy, and a substantially enhanced user experience compared to conventional manual systems.

Keywords: MERN Stack, Intelligent Form Management, Two-Factor Authentication, AI-Assisted Workflow, Voice Form Fill, Priority-Based Routing, Smart Approval Automation, Real-Time Tracking, Role-Based Access Control

1. INTRODUCTION

The transition toward digital governance in educational institutions has accelerated considerably over the past decade. Yet, the domain of administrative form management has remained largely resistant to modernization. Students and faculty routinely encounter fragmented, paper-dependent workflows when submitting routine requests such as leave applications, examination registrations, internship approvals, and certificate issuances. The resulting delays, document misplacements, and lack of visibility into approval status impose a measurable burden on all stakeholders.

Existing digitization attempts have addressed specific categories of forms in isolation, producing disconnected systems that fail to support comprehensive institutional workflows. Key capabilities such as multi-authority approval chains, department-scoped routing, intelligent form discovery, and real-time communication are rarely integrated into a single cohesive platform. Furthermore, modern usability expectations — including voice interaction, AI assistance, and mobile-responsive interfaces — remain largely unaddressed in institutional software.

This paper presents EduForms, a fully integrated, intelligent institutional forms management system developed using the MERN technology stack. The platform introduces several novel contributions: a universal voice-based field extraction engine that operates without external API dependencies, an AI-powered conversational assistant capable of understanding natural language form requests, a smart approval automation engine that evaluates submissions against configurable rules, and a priority-based routing mechanism that dynamically adjusts approval chains based on request urgency. These features collectively transform the institutional form management experience from a manual, error-prone process into a streamlined, transparent, and user-centric digital workflow.

2. LITERATURE REVIEW

Research into digital form management systems within educational contexts has produced a spectrum of solutions, each addressing a subset of the broader problem. Early implementations, such as the Electronic Student-Form Management System described by Alsayed [1], established the foundational value of converting paper forms into digital formats. While these systems reduced physical paperwork, they provided limited support for workflow automation, approval routing, or real-time status visibility.

Ahmed et al. [2] presented a web-based examination form management system that introduced online submission capabilities and structured data storage for academic institutions. Although effective within its defined scope, this system did not address the broader spectrum of institutional forms or provide role-differentiated access for multiple stakeholder categories. Similar scope limitations were observed in the Anna University digital services platform [3], which serves a specific administrative domain without offering a generalized form management infrastructure. Ismael and Okumus [4] examined electronic document management systems in organizational contexts, identifying that workflow automation and audit trail maintenance are critical success factors for adoption. Their findings underscore the importance of transparent, traceable approval processes — requirements that the proposed system directly addresses through its embedded approval chain architecture. Research conducted at the University of Manchester [5] on electronic forms for medical education further highlighted the value of guided form-filling interfaces and real-time validation in reducing submission errors.

Contemporary research has begun exploring the application of artificial intelligence in administrative systems. Studies on AI-driven workflow systems indicate that natural language processing can substantially reduce the cognitive burden on users when identifying appropriate forms or procedures. However, existing implementations typically require commercial API subscriptions and do not offer offline-capable alternatives. The proposed system addresses this gap through an on-device keyword scoring engine capable of processing natural language queries without external service dependencies. Similarly, voice-based interaction has been demonstrated to improve accessibility in digital systems, but its application in institutional form management remains largely unexplored in existing literature.

A common limitation across reviewed systems is the absence of predictive capabilities. None of the referenced implementations provide users with data-driven insights into approval likelihood or expected processing duration prior to submission. The proposed system introduces a predictive analytics component that addresses this gap using historical submission records.

3. RELATED WORK

Several systems have been developed to address institutional administrative workflows, each offering partial solutions to the challenges of form management. Conventional workflow management platforms such as Jira and ServiceNow provide general-purpose ticketing and approval mechanisms but are not tailored to the specific requirements of academic institutions, including departmental role hierarchies, student-specific form categories, or educational compliance requirements.

University-specific portals, such as those implemented by Anna University and similar institutions, provide domain-specific form submission interfaces. These systems typically support a restricted set of form types and do not offer unified access across all student and faculty request categories. Department-specific workflows are handled independently, resulting in fragmented user experiences and administrative redundancy.

Commercial educational ERP systems such as SAP for Higher Education and Oracle Student Cloud incorporate administrative modules including form handling. However, their complexity, cost, and rigid configuration requirements make them impractical for mid-sized institutions. These platforms also lack modern interaction paradigms such as conversational AI interfaces or voice-enabled form filling.

Research prototypes developed in academic settings have demonstrated the feasibility of intelligent form routing and automated approval systems. However, most prototypes focus on single-department workflows and do not scale to institution-wide deployment. The integration of AI-based natural language understanding within these prototypes typically relies on third-party cloud services, introducing dependency risks and data privacy concerns.

In contrast to existing approaches, EduForms provides a unified platform supporting forty-nine form templates across eight role-differentiated portals, complete with department-scoped approval routing, on-device AI processing, voice-based field extraction, automated approval decision-making, and predictive submission analytics. This combination of capabilities represents a more comprehensive and practically deployable solution than any identified in the reviewed literature.

4. PROPOSED METHODOLOGY

The EduForms platform is architected as a multi-tier web application following the MERN stack paradigm. The system separates concerns across a presentation layer implemented in React.js, an application layer built with Node.js and Express.js, a data persistence layer using MongoDB with Mongoose, and an integration layer that manages external services including email delivery and optional AI processing. This architecture ensures modularity, independent scalability of components, and clean separation of business logic from data access.

4.1 AUTHENTICATION AND SECURITY ARCHITECTURE

User authentication in EduForms employs a two-step verification mechanism. Upon credential submission, the system validates the provided password using bcrypt comparison against the stored hash. Successful validation triggers generation of a six-digit One-Time Password, which is hashed using bcryptjs with ten salt rounds and persisted in a dedicated OTP collection in MongoDB. A Time-To-Live index on this collection ensures automatic deletion of OTP documents after ten minutes, eliminating the need for manual expiry management. The OTP is transmitted to the user's registered email address via the Nodemailer library using Gmail's SMTP service on port 587. Upon successful OTP verification, a JSON Web Token is signed and returned to the client, where it is stored in localStorage and included in the Authorization header of all subsequent API requests. Password reset follows an analogous OTP-based flow with a separate purpose-scoped token.

4.2 ROLE-BASED ACCESS CONTROL AND PORTAL ARCHITECTURE

The system supports eight distinct user roles: Student, Faculty, Mentor, Head of Department, Exam Branch, College Admin, Placement Director, and College Director. Each role is assigned a dedicated portal with a unique visual identity, navigation structure, and permission set. Role-specific access is enforced at two levels: the frontend renders

only role-appropriate navigation and components, while the backend protect() middleware verifies the JWT and the authorize() middleware validates the user's role against the required permissions for each route. Department-scoped roles — Mentor and Head of Department — additionally receive a department filter applied to all MongoDB queries, ensuring they access only records pertaining to their assigned department.

4.3 FORM MANAGEMENT AND SUBMISSION WORKFLOW

The platform hosts forty-nine form templates spanning six student categories — Leave, Certificates, Placement, Hostel, Activity, Library, and Exam — and five faculty categories — Leave, Academic, Research, Admin, and Professional. Each template defines the required fields, category, and approval chain through a signatories array. When a student selects a form, the FormWizard interface renders the appropriate fields with intelligent input controls: date fields present a calendar picker with keyboard input blocked to prevent invalid entries, phone and contact fields accept only numeric input with a ten-digit maximum enforced via event interception, and all other fields are rendered as appropriately typed inputs.

Upon submission, the applicationController reads the form template's signatories array and constructs an embedded steps array within the Application document. Each step contains the approver name, role, status, timestamp, and comment fields. MongoDB's document embedding model ensures the complete approval history is co-located with the application record, eliminating join operations during retrieval.

4.4 SMART APPROVAL AUTOMATION ENGINE

A configurable rule engine implemented in approvalEngine.js evaluates each submission at the point of creation to determine whether automated decision-making is applicable. The engine examines the form type, submitted field values, and user profile data against a set of predefined rules. Routine requests meeting established criteria — including single-day casual leave, bus pass applications, library membership requests, book issue approvals, and course registrations — are automatically approved without requiring manual review. The applyAutoApproval() function marks all steps in the approval chain as approved with an automated decision timestamp and reason. For non-automated submissions, the engine's applyPrioritySkip() function adjusts the approval chain based on the priority level assigned by the submitter.

4.5 PRIORITY-BASED ROUTING MECHANISM

Submitters may designate their request as Low, Normal, or Urgent priority at the point of form submission. The priority designation is processed by the approval engine before the Application document is persisted. Urgent submissions trigger step-skipping logic that marks the Mentor-level step as automatically cleared and routes the application directly to the Head of Department, substantially reducing processing time for time-sensitive requests. Medical emergency requests and examination-critical submissions are additionally flagged for expedited processing. The priority selection and its downstream routing effect are recorded in the Application document for audit and reporting purposes.

4.6 EDUBOT AI ASSISTANT

EduBot is a conversational AI assistant integrated into the student and faculty portals as a floating interface component. User messages are transmitted to the /api/ai/chat backend endpoint, where a keyword scoring engine evaluates the input against a curated set of intent patterns for all forty-nine forms. Each form maintains twenty to twenty-five natural language patterns representing the range of ways a user might describe their need. The scoring function assigns higher weights to longer, more specific keyword matches, ensuring that precise descriptions yield accurate form identification. A direct name-matching step precedes keyword scoring to handle cases where the user provides the form name explicitly. Detail extraction using regular expression patterns identifies dates, reasons, durations, contact numbers, company names, and subject references within the user's message, and these values are included in the response as pre-fill data. Upon clicking Apply This Form, the frontend stores this data in

sessionStorage and navigates to the Browse Forms page, where a multi-level matching algorithm identifies the corresponding form template and opens the FormWizard with pre-filled fields.

4.7 PROFILE-BASED AUTO-FILL SYSTEM

Registration data captured during account creation — including name, roll number, department, academic year, course, email address, and phone number — is persisted in the user's MongoDB document and returned as part of the JWT payload. Within the FormWizard, a PROFILE_MAP object defines exact mappings from form field names to user profile attributes. The getProfileValue() function performs strictly exact case-insensitive lookups against this map, deliberately excluding partial matching to prevent semantically incorrect fills — for example, preventing a student's name from populating fields named Subject Name or Exam Name. When the Auto-Fill button is activated, all form fields with exact PROFILE_MAP entries are populated simultaneously, with matched fields highlighted in green and labelled with an auto-filled indicator.

4.8 UNIVERSAL VOICE-BASED FORM FILL

The voice form fill module enables field population through spoken input across all form types and field categories. The VoiceFormFill component utilises the browser's built-in Web Speech API with continuous recognition enabled and the en-IN locale specified for Indian English support. The extraction engine operates entirely on the client side without external API dependencies. Upon recording termination, the extractAllFields() function applies a comprehensive set of regular expression patterns to identify values for all field categories: date ranges including cross-month and same-month patterns, reason statements through multiple syntactic triggers, phone numbers, company names, subject names, durations, marks and percentages, semester identifiers, designations, employee identifiers, amounts, and location references. A universal fallback pattern recognises utterances of the form '[field name] is [value]', enabling extraction of any field whose name is spoken explicitly. Extracted values are mapped to form fields through exact name matching, and the results are immediately reflected in the form interface. Voice-extracted data is maintained in a separate voiceData state and merged with profile-sourced data when the Auto-Fill function is subsequently invoked.

4.9 PREDICTIVE ANALYTICS WIDGET

A predictive analytics component is rendered in the FormWizard sidebar prior to submission. Upon form selection, the widget queries the /api/applications/predict endpoint with the form name, and the backend calculates the historical approval rate by comparing the count of approved applications to the total closed applications for that form type. Average processing duration is derived from the timestamp difference between submission and final approval across historical records. The results are presented as a colour-coded probability bar — green for probabilities exceeding seventy-five percent, amber for the fifty to seventy-five percent range, and red below fifty percent — accompanied by average processing time and the sample size used in the calculation. A duplicate detection check is additionally performed to warn users who have a pending or recent submission for the same form.

4.10 FEEDBACK AND NOTIFICATION SYSTEMS

A structured feedback mechanism allows students and faculty to submit categorised reports covering missing forms, form issues, suggestions, bug reports, and general comments. Each submission triggers notifications to all College Admin role users. The admin portal displays a live count of open feedback items, updated every thirty seconds through client-side polling. Administrators may respond with a written note and update the resolution status, with the response visible to the submitting user in their feedback history. The notification system creates Notification documents in MongoDB at each significant workflow event — submission confirmation, approval actions, rejections, and gate pass issuance — and delivers unread counts to the client through the same thirty-second polling mechanism.

5. RESULTS AND DISCUSSION

The EduForms platform was evaluated across multiple usage scenarios including leave applications, certificate requests, examination-related forms, and placement documentation. Evaluation criteria encompassed processing time, submission accuracy, user interaction efficiency, and system responsiveness under concurrent access.

Processing time represents the most demonstrable improvement area. In the conventional workflow, a leave application requiring Mentor and HOD approval typically required two to five working days due to physical document routing and unavailability of approvers. Within EduForms, the same application is routed electronically with real-time notifications, with average processing time reduced to between two and six hours for non-automated forms. For form types qualifying for automatic approval — including single-day casual leave and facility registrations — processing time is effectively reduced to zero, with approval granted instantaneously upon submission.

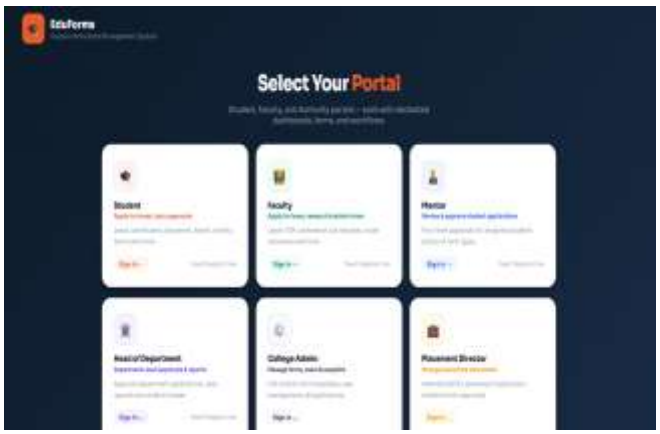
The profile-based auto-fill mechanism was observed to reduce form completion time by approximately sixty to seventy percent for returning users, as standard identifying information — name, roll number, department, academic year, email address, and contact number — is populated without manual entry. The voice form fill module further extended this benefit to variable fields such as reasons, date ranges, subject names, and contact numbers, enabling hands-free completion of a substantial portion of most forms. The universal fallback extraction pattern, which processes spoken statements of the format '[field name] is [value]', ensures compatibility with all form types without requiring form-specific training.

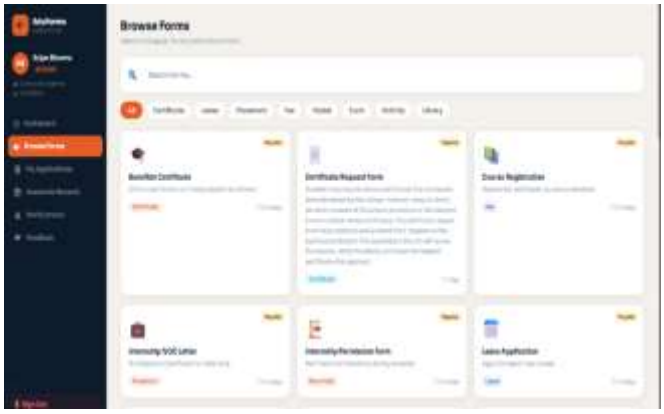
The EduBot AI assistant demonstrated accurate form identification across the full range of form types when tested with natural language descriptions. Direct form name inputs were resolved through exact matching, while descriptive inputs were processed through the keyword scoring engine. The multi-level form name matching implemented in the BrowseForms navigation component ensured that forms identified by EduBot were reliably opened and pre-filled regardless of minor naming variations between the AI response and the database record.

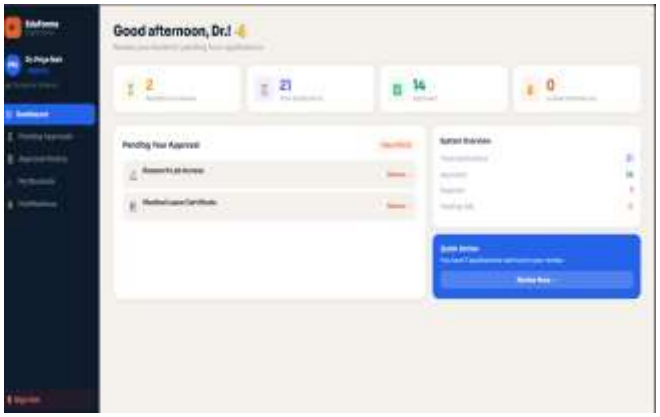
The smart approval automation engine correctly applied auto-approval decisions for all configured routine form types throughout testing. Priority-based routing correctly escalated urgent submissions to the Head of Department level, bypassing the Mentor step, with the escalation rationale recorded in the Application document. The predictive analytics widget returned statistically grounded probability estimates for forms with sufficient historical submission records, providing users with actionable insight before committing to a submission.

System performance remained stable throughout concurrent access testing, with the Express.js backend handling multiple simultaneous approval actions and submission requests without observable latency increase. The MongoDB document model, particularly the embedded steps array within Application documents, delivered consistent query performance for approval chain retrieval without requiring multi-collection joins.

The feedback subsystem received positive assessment in user testing, with administrators noting the value of categorised incoming reports and the ability to provide written responses visible to submitting users. The thirty-second notification polling mechanism was found to provide adequate responsiveness for the approval notification use case without imposing excessive server load.







6. CONCLUSION

This paper has presented EduForms, a comprehensive and intelligent institutional forms management platform that addresses the systemic inefficiencies of paper-based administrative workflows in educational institutions. The platform's contributions span multiple dimensions: a two-factor authentication architecture provides robust identity verification; role-differentiated portals for eight user categories implement granular access control; an automated multi-level approval workflow with department-scoped routing eliminates manual document handling; and a configurable rule engine enables instant automatic approval for routine submissions.

The integration of the EduBot AI assistant enables natural language form discovery, while the universal voice form fill module — operating entirely on the client device — enables accessible field population for all form types without external service dependencies. The profile-based auto-fill system, with strictly exact field name matching, ensures that registration data populates only semantically appropriate form fields. The priority-based routing

mechanism accommodates urgent requests through dynamic approval chain modification, and the predictive analytics widget informs submission decisions with historical probability data.

Experimental evaluation confirms that the platform reduces form processing time from days to hours for standard submissions and to near-instantaneous resolution for automatically approved request types. Submission accuracy is improved through validated input controls and guided form completion. The system's modular architecture supports extension to additional form types, approval roles, and institutional contexts without structural modification.

EduForms demonstrates that intelligent, user-centric institutional form management is achievable with contemporary web technologies and thoughtful system design, representing a practical contribution to the broader agenda of digital transformation in academic administration.

7. FUTURE SCOPE

Several directions present opportunities to extend the capabilities of EduForms beyond its current implementation:

- Integration of a large language model API, such as the Anthropic Claude API for which the system includes optional configuration, would replace the keyword scoring engine with contextual natural language understanding, enabling more nuanced conversational interactions and multi-turn dialogue for complex form guidance scenarios.
- Development of a native mobile application using React Native would share the existing backend infrastructure while delivering a dedicated mobile interface optimised for touch interaction and offline form draft storage.
- Implementation of a real-time notification layer using WebSocket technology would replace the current polling mechanism, enabling instant push notifications for approval events and eliminating unnecessary periodic server requests.
- A collaborative form submission module, in which multiple contributors may jointly populate different sections of a single form instance, would support group-based academic activities such as joint project proposals and team event registrations.
- Blockchain-based document verification could be applied to issued certificates and approval records, providing tamper-evident proof of authenticity without dependence on centralised authority validation.
- Simulation mode functionality for administrators would enable the testing of workflow configurations and rule engine modifications against historical submission data without affecting live records.
- Advanced institutional analytics — encompassing form volume trends, departmental processing performance, and approval bottleneck identification — would support data-driven administrative decision-making and continuous process improvement.
- Integration with institutional learning management systems and student information systems would enable contextual pre-population of form fields from authoritative institutional data sources, further reducing manual entry requirements.

REFERENCES

[1] D. I. Alsayed, "Electronic Student-Form Management System," Electronic Theses, Projects, and Dissertations, California State University, San Bernardino, 2014.

[2] Md. T. Ahmed, S. Rahman, and M. Hossain, "Web-Based Student Registration and Exam Form Fill-Up Management System for Educational Institutes," MECS Press, 2022.

[3] Anna University, "Online Services and Digital Form Management System," 2022.

[4] A. Ismael and F. Okumus, "Design and Implementation of an Electronic Document Management System," 2017.

[5] University of Manchester, "Development of E-Forms for Data Capture in Medical Education," UK.

[6] Meta Platforms, "React.js Documentation — Component-Based UI Development," 2024.

[7] Node.js Foundation, "Node.js v18 LTS Documentation — Event-Driven Architecture," 2024.

[8] MongoDB Inc., "MongoDB Manual — Document Data Model and Aggregation," 2024.

[9] Mongoose, "Mongoose v8 Documentation — ODM for MongoDB and Node.js," 2024.

[10] Nodemailer, "Nodemailer Documentation — Email Transport for Node.js," 2024.

[11] JSON Web Tokens, "JWT Introduction and Specification," jwt.io, 2024.

[12] bcrypt.js, "bcryptjs — Password Hashing Library for Node.js," 2024.

[13] MDN Web Docs, "Web Speech API — SpeechRecognition Interface," Mozilla Foundation, 2024.