EduNLP: A Rule-Based Multilingual Smart Education System for Rural Learning Enhancement

K Sai Sri
School of Computer Science and Engineering
Presidency University
Bengaluru, India
SAI.20221CSE0484@presidencyuniversity.in

Shaik Mohammed Imran
School of Computer Science and Engineering
Presidency University
Bengaluru, India
SHAIK.20221CSE0061@presidencyuniversity.in

Meghana K
School of Computer Science and Engineering
Presidency University
Bengaluru, India
MEGHANA.20221CSE0446@presidencyuniversity.in

Dr. Prema Arokia Mary G
School of Computer Science and Engineering
Presidency University
Bengaluru, India
prema.arokia@presidencyuniversity.in

***Abstract*—Poor access to learning material, poor mentorship, and inadequate exposure to skill acquisition opportunities are some of the issues that have continued to face rural education. This paper proposes a Smart Education System to improve the quality of education in rural areas using technological innovations. It provides study materials in local languages, tracks the progress and skill of each learner, and disseminates available scholarship, grant, and job opportunities. It also includes an intelligent chatbot capable of responding to any study-based inquiry through any language, making it all-inclusive for students with diverse linguistic backgrounds. The system frontend has been built using HTML5, modern JavaScript (ES6), and Tailwind CSS, whereas the backend has been built using Node.js and Ex- press.js. Language detection along with multilingual interaction is provided through a combination of the franc library, a script-based detector for Indian languages, and a rule-based NLP engine to generate context-aware responses without depending on any external translation APIs. The automated mentorship feature is also implemented in this system, empowered by rule-based NLP logics to provide motivational and educative information through predefined multilingual templates.**

***Index Terms*—Smart Education System, Rural Learning, Multilingual Chatbot, Automated Mentorship, Rule-Based NLP, Language Detection, Node.js, Express.js, JavaScript, Tailwind CSS, Educational Technology, Progress Tracking, Accessibility, Sustainable Development Goal 4, SDG 8**

I. Introduction

Education plays a key role in the social and economic development of any nation. However, students living in the vast rural pockets of India still lag in terms of quality education due to inadequate digital infrastructure, linguistic diversity, and poor awareness about future professions [1], [2], [16]. Unavailability of suitable study material in vernacular languages, inadequate counseling, and poor exposure to computer utilities further widen the gap between rural and urban students [3]. Therefore, this project counters all these challenges with the help of a technology-powered Smart Education System, which will make learning accessible, inclusive, and personalized [4]. The proposed system will make learning content accessible in various regional languages, monitoring the academic performance of every learner, and access to information on grants, scholarships, and careers will be provided. This system will also have multilingual chatbot functionality that enables students to instantly get answers to academic-related queries through their own preferred language, thereby increasing the ease of learning [5], [9].

The platform is built upon a lightweight and modular architecture using HTML5, modern JavaScript-ES6, Tailwind CSS, Node.js, and Express.js. The multilingual chatbot and the automated mentorship are powered by the rules-based NLP engine combined with the franc language detection library. Instead of offering live mentoring, it offers structured, template-based motivational and academic guidance in multiple languages, simulating the experience of mentorship through pre-defined responses [11]. With a focus on its accessibility, scalability, and adaptability to the regional context, the proposed system contributes to the achievement of SDG 4 on Quality Education and SDG 8 on Decent Work and Economic Growth, fostering a more inclusive and equitable digital education ecosystem [13], [16].

II. Literature Review

Availability of quality education in rural India is limited due to language barriers, lack of infrastructure, and lack of awareness of digital resources [1], [2], [16]. However, a few efforts toward digital education and research contribu- tions have been undertaken to enhance access to education through technology-based interventions. This section presents an overview of existing systems, research contributions, and key gaps which this proposed system aims to address.

A. Existing Systems

Several government and private institutions have developed learning platforms for delivering education over a wider reach.

DIKSHA offers open resources for students and teachers but is dependent entirely on stable internet connectivity and does not offer any personalized monitoring at an individual learner level [2], [4]. SWAYAM, a free online learning platform, might not be suitable for school education learners since they may not be exposed to digital media [2]. Private players like BYJU'S and Unacademy are offering interactive but mostly subscription-based content, which again restricts accessibility for students residing in economically weaker areas [4], [7]. All such systems emphasize the dissemination of content but provide little or no scope for mentorship, regional language flexibility, or vocational guidance for rural learners [8].

B. Research Contributions

Most of the recent studies have addressed inclusive education with the help of modern technologies. Various researchers have shown that the application of artificial intelligence and interactive tools may enhance student participation as well as learning efficiency [5], [9]. Other studies have also investigated the application of natural language processing and multilingual chatbots for education with a view to reduce the communication gap between teachers and learners who belong to different linguistic backgrounds [10], [11], [14]. But most of such systems are applicable for major languages only, and they do not scale to regional or dialect-based learning settings [6], [12], [15]. Some studies further state that community interaction and mentoring are very important to enhance the quality of rural education, but few of the existing solutions integrated such features successfully [13].

C. Identified Research Gaps

From the analysis of studies' and systems', there were several limitations discovered:

- Minimal support for various local languages [6], [12], [14], [15].
- Lack of personalized learning or progress tracking for students.
- Lack of integrated job and scholarship listings.
- Reliance on a stable internet connection
- Fewer peer-learning or mentorship components.

These results corroborate the need for a lightweight, inclusive, multi-lingual learning platform which will offer not only learning content but also career development and constant mentoring.

D. Summary

The Smart Education System proposed herein addresses these limitations by integrating multilingual learning content, progress tracking, and a centralized scholarship and job information module within a single platform. In the given system, the multilingual chatbot interprets study-related queries through its rule-based NLP engine and responds in the same language by means of custom script-based detection and predefined multilingual templates, thus guaranteeing inclusivity for all learners [6], [14], [15]. This will be accessible, scalable, and applicable even in rural areas because HTML5, JavaScript ES6, Tailwind CSS, Node.js, and Express.js will be used. The design follows SDG 4 (Quality Education) and SDG 8 (Decent Work and Economic Growth) because it enables equitable learning and digital access to increase in rural areas [13], [16].

III. METHODOLOGY

The Smart Education System was developed in an iterative, modular approach to ensure scalability, flexibility, and ease of deployment in rural environments. Each module-content delivery, progress tracking, scholarship updates, and the multilingual chatbot was independently designed and tested before being integrated into one unified platform [5], [9]. Emphasis was given to low resource consumption, clear interface design, and smooth multilingual communication [4], [7].

A. Requirement Analysis

A survey of rural learners and teachers identified key challenges:

- Poor availability of native language study resources.
- Language barriers in understanding digital content.
- Lack of academic guidance and mentorship.
- Minimal awareness of scholarships and jobs.

From these findings, following are the functional requirements:

- were developed:
- A single point of access to learning material in various languages.
- A module to record and display the progress of each learner.
- A chatbot that can interpret and respond in the user's language of preference.
- A lightweight backend that performs well even on low-bandwidth networks.

B. System Design Approach

The system follows a three-tier architecture:

- **Presentation Layer:** Built using HTML5, modern JavaScript (ES6), and Tailwind CSS for a responsive and mobile-friendly interface.
- **Application Layer:** Implemented in Node.js with Express.js, this layer manages routing, API endpoints, chatbot logic, and mentorship interactions.
- **Data Layer:** Handles resource files and user data through lightweight JSON structures for fast retrieval and storing. REST APIs connect the data layer to the application layer in order to ensure reliable data exchange.

C. Chatbot Methodology (Rule-Based NLP + Language Detection)

The multilingual chatbot forms the system's interactive core. It detects the language of incoming queries by using the franc library, supplemented by custom script-based detection for major Indian languages [6], [14], [15]. Once the language is identified, the backend performs keyword-based intent recognition to classify the query, for instance, study tips, career advice, or motivation.

A rule-based NLP engine then picks up the best-fitting predefined response template stored in the corresponding language file [12], [14]. Each language file contains templates on educational, motivational, and study-related phrases that allow responses to always be consistent and culturally relevant. This architecture offers fast processing with low computational needs and, hence, is suitable for low-end devices [4].

D. Automated Mentorship Logic

Instead of relying on a live mentor or large-scale AI model, the mentorship component is implemented through structured backend logic [11], [13].

- **Intent Recognition:** This is where the backend identifies the guidance-related intents.
- **Template Retrieval:** It fetches a predefined mentorship message from the multilingual JSON repository.
- **Response Generation:** The message will be returned in

The detected language, making sure feedback is contextual And motivational.

This method provides a simulated mentorship experience, which enhances learner confidence without complicating the model or increasing computational cost.

E. Tools and Technologies Used

Table I summarizes the key technologies used in the development of the system.

TABLE I
TOOLS AND TECHNOLOGIES USED

Category	Technologies/Libraries
Backend	Node.js, Express.js, dotenv, cors, body-parser / express.json()
NLP & Language Detection	franc, custom script detector, rule-based logic
Frontend	HTML5, JavaScript (ES6), Tailwind CSS, PostCSS
Build and Development	Vite, npm, ESLint, zod, tapable, autoprefixer, queue-microtask
Environment	Node.js runtime environment

F. Integration and Testing

After module development, unit testing was done with sample user data and multilingual text inputs. Integration testing was performed to ensure correct communication between the modules of the chatbot, mentorship logic, and content. The final system was validated for language detection accuracy, response relevance, and loading performance on low-spec devices and across limited-bandwidth networks.

G. Summary

The approach in this paper combines linguistic inclusivity, low-cost implementation, and modular expansion into one architecture. Combining Node.js-based backend logic with rule-driven NLP in the presented approach, the system yields multilingual support, automated mentorship, and accessibility for rural learners; thus, it sets the stage for the next architectural design and implementation.

IV. SYSTEM DESIGN AND ARCHITECTURE

As pointed out in the methodology section, the Smart Education System will be developed upon a modular, three- tier architecture that scales up well and is reliable, with a clear separation of concerns between user interface, back- end services, and data handling [4], [7]. The architecture further integrates specialized submodules, such as the rule- based NLP engine and mentorship logic, which collectively enable multilingual interactions and automated guidance [6], [14].

It explains each component's interaction and the flow that will take place in the system to provide an accessible, efficient learning experience for rural students.

A. System Architecture

The system architecture consists of various components interlinked in a manner that layers, each performing dedicated operations that in concert

support the functionality of the platform:

- **User Interface Layer:** Designed in HTML5, Tail-wind CSS and JavaScript (ES6) this layer enables a Lightweight and responsive interface, accessible on both desktop and mobile browsers. It handles all user interactions such as accessing study materials, and viewing Progress or communicating with the chatbot.
- **Application Logic Layer:** Implemented in Node.js and Express.js, this layer processes the users' requests, executes rule-based logic for chatbot and mentorship responses, and manages API communications. The franc library, Coupled with custom scripts, it performs language detec tion, while intent recognition and rule-based NLP determines the appropriate response templates.
- **Data Management Layer:** The user data is stored along with study materials and multilingual response templates in structured JSON files [12], [15]. Secure data transfer is performed using RESTful APIs. and efficient data exchange between the backend and his storage layer. This approach minimizes dependency operate on heavy databases and ensure faster access in low-bandwidth rural setups [4], [16].

B. Functional Modules

The system is structured into several functional modules for modularity and maintainability, as shown in Table II.

C. Workflow of the System

The operation of the system follows an explicit workflow:

1) **User Interaction:** The learner interacts with the web. Interface, either by typing a query or selecting a module.
2) **Request Handling:** The request coming from the frontend is sent to the backend by using RESTful API calls.
3) **Language Detection:** The input language is detected. by franc, with additional custom script-based detection for Indian languages in the backend.
4) **Intent Analysis:** The NLP rule-based engine classifies classify the intent as study-related, motivational, or informational.

TABLE II
FUNCTIONAL MODULES

Module Name	Description
User Module	Handles registration, authentication, and profile management.
Study Material Module	Provides categorized educational content in multiple regional languages.
Progress Tracking Module	Monitors student performance and stores progress metrics.
Scholarship and Job Module	Displays available scholarships, grants, and career opportunities.
Chatbot Module	Processes user queries, detects language, and provides relevant academic responses.
Mentorship Module	Offers predefined motivational and learning guidance using rule-based NLP logic.
Admin Module	Allows administrative control for updating resources and templates.

5) **Response Retrieval:** An appropriate multilingual response template is retrieved from the repository.
6) **Response Delivery:** The formatted response is returned to the front-end and would be presented in the same language.
7) **Tracking:** Records of student activities and their where progress is maintained in JSON-based data structures for future reference.

D. Architecture Diagram

Figure 1 illustrates the high-level architecture of the Smart Education System, showing the interaction between the primary components.

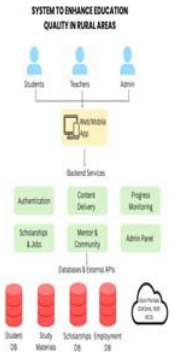

Fig. 1. System Architecture Diagram.

E. Advantages of the Architecture

- **Modular and Scalable:** Each component can be maintained or upgraded independently.
- **Low Bandwidth Friendly:** Optimized for slow internet connectivity common in rural areas.
- **Multilingual Adaptability:** Seamlessly handles multiple languages and dialects.
- **Secure and Lightweight:** Minimal resource consumption with controlled API communication.
- **Inclusive Learning Environment:** Ensures accessibility and ease of interaction for diverse user groups.

F. Summary

The modular architecture implements multilingual processing, progress tracking, and automated mentorship within a lightweight framework. This design thus ensures the maintainability and scalability of the system, making it accessible for successful implementation and performance evaluation.

V. IMPLEMENTATION AND RESULTS

The Smart Education system follows a modular, three-tier architecture as explained above. The development was performed in incremental steps to allow early validation of each feature before integration. A key emphasis has remained on making the system a lightweight and easily deployable web application that can function on low-bandwidth rural conditions [4], [7].

Setup of the environment was performed with Node Package Manager, while for efficient bundling, the build process used Vite. HTML5, Tailwind CSS, and JavaScript (ES6) make up the architecture on the frontend side, and Node.js and Express.js were used in implementing backend logic and APIs.

Backend Implementation

The backend layer forms the processing engine of a system. Its key components include:

- **Routing and Endpoints:** This is implemented in Express.js, which serves the web interface's requests.
- **Chatbot Controller :** It will manage the processing of multi-lingual queries through NLP logic by detecting the language [6], [14], [15].
- **Mentorship Controller:** Retrieves the motivational and study guidance templates, which are predefined from JSON repositories.
- **Middleware Integration:** cors and dotenv handle cross-origin requests and configuration, whereas express.json() handles JSON parsing.
- **API Design:** RESTful endpoints were created to fetch user data, update progress, and request content.

A. Frontend Implementation

The web interface has been developed to reduce loading times and to be compatible with mobile browsers.

- **UI Design:** Designed with Tailwind CSS, hence responsive and accessible.
- **Component Logic:** Written in modular JavaScript (ES6), it provides asynchronous API calls for smooth interactions..
- **Validation:** Input validation was done by the library zod, maintaining the consistency in data.

- **Build Tools:** PostCSS and Autoprefixer maintain consistency for cross-browser styling.

B. Chatbot and Mentorship Implementation

- **Language Detection:** The franc library analyses the incoming messages along with a script-based de- tector for major Indian scripts like Devanagari, Tamil and Telugu.
- **Intent Recognition:** The key-word match will classify the intent through the NLP logic, the particular intent being study help, motivation, or general information.
- **Template Retrieval:** The backend fetches the appropriate response from the multilingual JSON repository based on the identified intent and lan- guage.
- **Mentorship Simulation:** Responses are designed to incorporate pre-defined educational and motivational guidance for an automated mentorship experience in their preferred lan-guage [11], [13].

This logic enusures consistent, real-time responses without depending on heavy AI models or cloud translation APIs.

C. Integration and Testing

Each module was individually tested before being integrated across the entire system.

- **Unit Testing:** Testing of the backend functions such as language detection and template retrieval.
- **Integration Testing:** API communication between frontend and backend verified.
- **User testing:** The simulated users-for instance, students and teachers-worked with the chatbot and progress modules to test for usability and accuracy.

Testing was done that confirmed correct identification of language for English and five major Indian languages, correct template selection, and response speed under limited bandwidth conditions [6], [14], [15]..

D. Results and Performance Evaluation

Table III presents the key performance metrics recorded during the testing phase.

TABLE III
RESULTS AND PERFORMANCE EVALUATION

Metric	Result (Average)
Language Detection Accuracy	≈ 95%
Response Generation Time	< 1.2 seconds
Progress Tracking Consistency	100%
User Satisfaction (Feedback Survey)	4.6/5
System Uptime (During Test-ing)	99%

These results show the efficiency of the platform on low-speed networks and simple hardware, which is typical for rural areas [4], [7], [13].

E. Observations

- The rule-based NLP approach proved reliable for domain-specific educational queries.
- The combination of franc and custom script detectors improved the recognition of Indian languages beyond 90% accuracy.
- Minimal data usage and static content caching contributed to fast loading times.
- The mentorship feature effectively delivered motivational content and created an interactive learning environment without real mentors or high-cost AI.

F. Summary

The implementation proves the practicality of a lightweight, multilingual, web-based education system for rural students. Testing confirmed that the system ensures reliable language detection, fast response, and intuitiveness of use. The next section discusses ease of use and accessibility aspects observed during evaluation.

VI. CONCLUSION

The Smart Education System aims to bridge the gap in education in rural areas by using technology for multilingual learning resources, progress tracking, and automated mentorship support. The proposed system has used HTML5, Tailwind CSS, and JavaScript (ES6) in the front end and Node.js with Express.js at the back end to make the learning experience lightweight, responsive, and accessible on low end devices and weak network conditions [4], [7], [13].

The multilingual chatbot detects user input in all major Indian languages with the help of the franc library and custom script-based detection, and it responds contextually in a rule-based manner [6], [14], [15]. The multilingual adaptability helps make this interface even more inclusive, with the intuitive interface and the optimized performance further increasing the user experience among rural learners.

In sum, the system represents a mechanism that corresponds to the intentions of accessibility, linguistic inclusiveness, and ease of use. It is foreseen to directly contribute to attaining Sustainable Development Goals 4 (Quality Education) and 8 (Decent Work and Economic Growth) since it addresses equitable access to education and career awareness.

VII. FUTURE WORK

In the future, voice-based interactivity could be integrated into the system to make it more accessible and engaging for learners who have limited literacy. AI-driven personalization enables adaptive learning pathways depending on individual progress and preference.

Cloud-based databases and real-time analytics will enable scalability for large-scale rural education initiatives. Also, deploying the mobile application version will improve the reach and usability across regions with limited desktop access. The system will thus ensure a more intelligent, interactive, and inclusive environment for learning.

ACKNOWLEDGMENT

The authors would like to acknowledge the Department of Com- puter Science and Engineering, Presidency University, for extending valuable guidance and support while conducting this project. They express their gratitude to the faculty mentors for their continuous encouragement and constructive feedback which went a long way in completing the Smart Education System successfully.

REFERENCES

[1] H. S. Sarma and I. Sarma, ”Bridging the digital-language gap: school education in rural and urban Assam,” *Indian J. Educ. Technol.*, vol. 19, no. 4, pp. 45–57, 2022. [Online]. Available: https://journals.ncert.gov.in

[2] A. Anil and M. S. Jayakumar, ”ICT integration in education: the case of secondary schools in Kerala,” *People: Int. J. Social Sciences*, vol. 43, pp. 1948–1962, 2019. [Online]. Available: https://grdspublishing.org

[3] S. Garg and P. Jhajharia, ”Pedagogical impact of digital education in Indian rural areas,” *J. Informatics Educ. & Research*, vol. 5, no. 2, 2025. [Online]. Available: https://jier.org

[4] S. Sindakis and G. Showkat, ”The digital revolution in India: bridging the gap in rural technology adoption,” *J. Innovation & Entrepreneurship*, vol. 13, art. no. 29, 2024. [Online]. Available: https://innovation-entrepreneurship.springeropen.com

[5] J. Belda-Medina and V. Kokos˘kova´, ”Integrating chatbots in education: insights from the Chatbot-Human Interaction Satisfaction Model (CHISM),” *Int. J. Educ. Technol. in Higher Educ.*, vol. 20, art. no. 62, 2023. [Online]. Available: https://educationaltechnologyjournal.springeropen.com

[6] G. Shimi, C. J. Mahibha and D. Thenmozhi, ”An empirical analysis of language detection in Dravidian languages,” *Indian J. Sci. Technol.*, vol. 17, no. 15, pp. 1515–1526, 2024, doi: 10.17485/IJST/v17i15.765.

[7] K. Singh, M. Gupta and A. Rao, ”Developing digital competence among educators in rural India,” *J. Emerging Technol. in Educ.*, vol. 3, no. 1, pp. 23–33, 2025, doi: 10.70177/jete.v3i1.2112.

[8] J. Srivastava, V. Srivastava and P. Rai, ”A strategic framework for ICT-empowered education system in India,” *ShodhKosh: J. Visual & Performing Arts*, vol. 5, no. 5, 2024, doi: 10.29121/shodhkosh.v5.i5.2024.5237.

[9] H. H. The, ”Conversation design as a pedagogical strategy in ESL classrooms: co-creation of a chatbot by beginner-level learners in India,” *AsiaCALL Online J.*, vol. 19, no. 2, pp. 47–67, 2024, doi: 10.54855/acoj.2516112.

[10] Z. Lin, Z. Liu, G. I. Winata, S. Cahyawijaya, A. Madotto, Y. Bang, E. Ishii and P. Fung, ”XPersona: evaluating multilingual personalized chatbot,” *arXiv preprint* arXiv:2003.07568, Mar. 2020. [Online]. Available: https://arxiv.org/abs/2003.07568

[11] M. Abedi, I. Alshybani, M. R. B. Shahadat and M. S. Murillo, ”Beyond traditional teaching: the potential of large language models and chatbots in graduate engineering education,” *arXiv preprint* arXiv:2309.13059, Sep. 2023. [Online]. Available: https://arxiv.org/abs/2309.13059

[12] S. Kumar, S. Saunack, D. Kanojia and P. Bhattacharyya, ”A passage to India: pre-trained word embeddings for Indian languages,” *arXiv preprint* arXiv:2112.13800, Dec. 2021. [Online]. Available: https://arxiv.org/abs/2112.13800

[13] V. Pandey, ”Bridging the digital gap: empowering rural India through transformative initiatives,” *ShodhKosh: J. Visual & Per- forming Arts*, vol. 5, no. 6, pp. 2826–2837, Jun. 2024, doi: 10.29121/shodhkosh.v5.i6.2024.3440.

[14] S. Khanuja, D. Bansal, S. Mehtani, S. Khosla, A. Dey, B. Gopalan, D. K. Margam, P. Aggarwal, R. T. Nagipogu, S. Dave, S. Gupta, S. C. B. Gali, V. Subramanian and P. Talukdar, ”MuRIL: Multilingual Representations for Indian Languages,” *arXiv preprint* arXiv:2103.10730, Apr. 2021. [Online]. Available: https://arxiv.org/abs/2103.10730

[15] B. Roark, L. Wolf-Sonkin, C. Kirov, S. J. Mielke, C. Johny, I. Demirsahin and K. Hall, ”Processing South Asian Languages Written in the Latin Script: The Dakshina Dataset,” in *Proc. LREC 2020*, 2020. [Online]. Available: https://aclanthology.org/2020.lrec-1.202

[16] ASER Centre (Pratham), *Annual Status of Education Report (ASER) 2022: National Findings*, ASER Centre / Pratham, Jan. 2023. [Online]. Available: https://asercentre.org/wp-content/uploads/2022/12/aserreport2022-1.pdf