International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 11 | Nov - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

# Design and Development of a Smart Online College Admission Management System with OCR-Based Document Verification

**Prof. Dipali S. Nale[1], Pranjal Sonawane[2], Prajwal Gaikwad[3], Srushti Shirke[4], Shruti Thorat[5]**

[1] Assistant Professor, Dept of Computer Engineering, SVPM's COE, Malegaon Bk. (Pune), India.
Email: dipaliraut14@gmail.com
[2] Dept of Computer Engineering, SVPM's COE, Malegaon Bk. (Pune), India.
Email: pranjalsonawane2365@gmail.com
[2,3,4,5] Department of Computer Engineering, SVPM's COE, Malegaon Bk. (Pune), India.

***Abstract*—The traditional college admission process involves several manual activities, including application form submission, document collection, eligibility verification, fee calculation, and admission status management. These activities can be time consuming and may lead to errors, delays, and difficulties in tracking applications. To address these challenges, this research proposes a Smart College Admission Management Software Application with OCR Based Document Verification.**

**The proposed software application provides a centralized digital environment through which students can register, submit ad-mission applications, upload required documents, select courses, and track their application status. Optical Character Recognition (OCR) technology is incorporated to extract relevant information from uploaded documents and support the document verification process. An eligibility checking mechanism assists admission staff in validating student information according to predefined admission criteria. The application also includes administrative and staff modules for application management, document review, fee related operations, and scholarship information.**

**The software is developed using React with Vite for the frontend, Python FastAPI for backend services, and MySQL/MariaDB for database management. The proposed application aims to reduce manual paperwork, improve information management, and provide greater transparency throughout the admission process. Its effectiveness will be evaluated through functional testing, document extraction accuracy, and application processing performance.**

****Keywords:** College Admission Management, Software Application, Optical Character Recognition, Document Verification, Admission Automation, Eligibility Verification.**

## I. INTRODUCTION

The admission process is an important administrative activity in educational institutions. Traditionally, colleges manage admission applications, student information, document collection, eligibility verification, and fee-related activities through manual procedures or separate software applications. These methods can create difficulties in maintaining accurate records, coordinating admission activities, and providing timely application updates to students. With the advancement of web technologies, online admission management systems provide an opportunity to simplify and centralize these activities. However, document verification remains an important part of the admission process because admission staff must examine student certificates and other supporting documents before processing applications.

This research presents the design and development of a Smart Online College Admission Management System with OCR-Based Document Verification. The application is developed using React with Vite for the frontend, Python FastAPI for backend services, and MySQL/MariaDB for database management.

The system provides facilities for student registration, online admission applications, course selection, document uploading, application tracking, fee management, and scholarship-related information. It also provides separate interfaces for students, admission staff, and administrators according to their responsibilities.

Optical Character Recognition (OCR) technology is integrated to extract textual information from uploaded documents. The extracted information can assist the admission staff during document verification and reduce the need for repeated manual data entry. The final verification and admission decisions remain subject to the institution's defined rules and authorized staff review.

The proposed system aims to provide a centralized, accessible, and organized platform for managing college admission activities while improving document-processing workflows and application transparency.

## II. PROBLEM STATEMENT

Traditional college admission procedures involve several manual activities, such as collecting student information, receiving documents, checking eligibility, calculating fees, and communicating application status. Managing these activities manually can result in paperwork, repetitive data entry, delayed document verification, record-management difficulties, and limited transparency for applicants.

Furthermore, manually reading information from multiple student documents requires considerable staff effort. There is a need for an integrated system that supports online admission applications, OCR-assisted document information extraction, eligibility checking, fee management, and application tracking through a centralized platform.

Therefore, the proposed research focuses on developing a smart online college admission management application that integrates these functionalities to support an organized and efficient admission process.

## III. OBJECTIVES

1) To develop a centralized online college admission management application.
2) To provide secure student registration and login facilities.
3) To enable students to submit admission applications and upload required documents online.
4) To integrate OCR technology for extracting relevant information from uploaded documents.
5) To assist admission staff in verifying documents and checking student eligibility.
6) To provide fee calculation and scholarship related information.
7) To develop separate dashboards for students, admission staff, and administrators.
8) To provide application status tracking and admission related notifications.
9) To reduce manual paperwork and repetitive data-entry activities.
10) To maintain organized student and admission records using a relational database.

## IV. PROPOSED SYSTEM

The proposed system is a web-based college admission management application that integrates student services and administrative admission operations into a centralized platform.

Students can create accounts, log in, fill out admission applications, select courses, upload supporting documents, and track their application progress. The OCR module extracts relevant textual information from uploaded documents and makes it available for verification.

The backend, developed using Python Fast API, processes application requests and communicates with the MySQL/MariaDB database. The React and Vite frontend provides an interactive interface for students and authorized staff members.

Admission staff can review uploaded documents, verify extracted information, and update application statuses. Administrators can manage admission-related information and oversee application processing.

The system also incorporates fee-management and scholarship-related facilities to help organize financial information associated with the admission process.

The proposed system is intended to improve the organization, accessibility, and transparency of college admission activities.

## V. LITERATURE REVIEW

Existing research demonstrates the importance of systems, automated admission workflows, and OCR-based document processing in educational and administrative applications.

### 5.1 Student Information Management System

Mardaniyan (2019) presented a web-based Student Information Management System that centralized student records and administrative activities. The study discussed database management, role-based access control, student information management, and reporting. It provides a foundation for developing centralized educational management applications.

### 5.2 Online Student Admission and Registration Management

Monic and Hussein (2026) proposed an Online Student Admission and Registration Management System to address manual admission procedures. Their system supports online registration, document uploading, application tracking, and administrative processing. The study highlights the importance of digital platforms in improving admission workflow management.

### 5.3 Integrated Academic and Admission Management

Hidayah presented an integrated web-based academic management and student admission monitoring system using the Django framework and MySQL database. The research combined academic information, payment administration, and student admission processes into a centralized application.

### 5.4 OCR-Based Document Processing

Hegghammer (2022) conducted a benchmarking study of OCR technologies, including Tesseract, Amazon Textract, and Google Document AI. The research highlights that OCR performance can vary depending on the document and recognition technology. This supports the need to evaluate OCR extraction quality when processing student documents.

## VI. RESEARCH GAP

From the identified research areas, the following research gap is considered:

- Admission information and document processing may be handled through separate activities.
- Manual document verification can increase staff workload.
- Student information may require repeated data entry.
- Application-status tracking may not be integrated with all admission operations.
- OCR-based text extraction needs to be connected with admission document review and eligibility checking.
- The proposed application aims to integrate these operations into a single admission management environment.

## VII. SYSTEM ARCHITECTURE

The application follows a client-server architecture executed locally. The React interface communicates with FastAPI endpoints over HTTP. The backend performs authentication, validates requests, stores application and document metadata through SQLAlchemy, saves uploaded files in a local upload directory, and invokes OCR for supported image formats. SQLite is the default database configured by the supplied project.

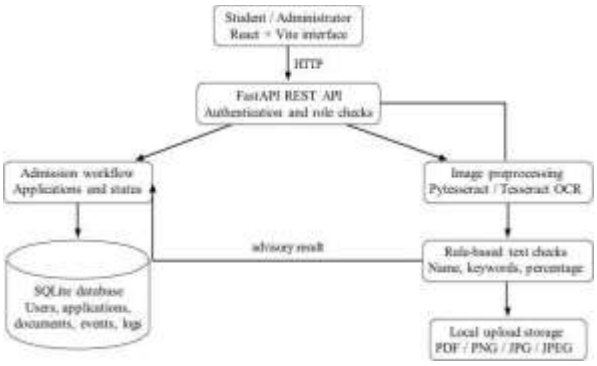



Fig. 1. High-level architecture of the implemented local software prototype.

*A. Student Module*

A student can create an account using a name, email address, mobile number, and password, sign in, submit an admission application, upload supporting documents, and view the application status, fee summary, and recorded status time-line. The application form captures course, department, year of study, academic year, category, previous percentage, and backlog count.

*B. Administrator Module*

The administrator workspace provides an application summary, search and status filtering, applicant and document details, document review actions, application status updates, fee total and fee-paid fields, and an audit-log viewer. Administrators can mark documents as verified, rejected, or requiring resubmission and can rerun the advisory text checks.

*C. Data Management and Security Controls*

The backend stores user, admission, document, status-event, and audit-log records. Passwords are stored as PBKDF2-HMAC-SHA256 derived hashes using a per-password salt and 310,000 iterations. Authenticated requests use signed JSON Web Tokens with an eight-hour expiry. Role checks restrict administrative endpoints to administrator accounts. Upload handling checks the file extension, size limit, and basic file signature before storing the file under a randomly generated name. These are prototype-level safeguards and do not replace a production security assessment.

## VIII. METHODOLOGY

**A. Student Registration and Authentication**

Students provide the registration information required by the application. The backend processes the request and stores the relevant account data in the database. Authentication controls access to student functions.

**B. Admission Application Submission**

After login, students complete the admission form, provide the requested academic and personal details, select the relevant course, and submit the application. The application stores the submitted information for subsequent review.

**C. Document Upload and OCR**

Students upload the admission documents supported by the application. The OCR component processes the supported input and extracts text. The extracted information is presented to the relevant application workflow for staff review or further processing, according to the implemented functionality.

**D. Staff Review and Status Management**

Autho rised staff inspect application details and uploaded documents, review OCR output, and update application status through the available staff functions. The application maintains the resulting records in the database.

**E. Database Operations**

MySQL/MariaDB is used to persist student details, admission applications, document metadata, and other supported admission related information. Backend services communicate with the database to retrieve and update records.

## IX. IMPLEMENTATION DETAILS

The frontend uses React 18 and Vite 5. The backend uses Python, FastAPI, Pydantic request validation, SQLAlchemy ORM, PyJWT, Pillow, and Pytesseract. The default persistence layer is SQLite, and uploaded files are stored locally. The project can be started as two local development processes: the FastAPI server and the Vite development server. The interface is accessed through a local browser address; this describes the local execution arrangement and does not imply public internet hosting. The implemented API groups include authentication, ad-mission creation and retrieval, document upload and down-load, administrator document re-verification, document review, admission-status updates, administrator summary, and audit-log retrieval. The admission eligibility note currently uses demo rules: a non-zero backlog triggers a manual-review note, and a previous percentage below 35 triggers a demo-threshold note. Final eligibility requires staff review and institution-approved rules.

**TABLE I**
**BACKEND TEST RESULTS FROM THE SUPPLIED PROJECT**

| Test case | Result |
|---|---|
| Matching marksheet text and declared percentage | Pass |
| Percentage mismatch detection | Pass |
| Student-name mismatch detection | Pass |
| Fuzzy name matching with OCR-like spelling noise | Pass |
| Unreadable / empty OCR text handling | Pass |
| Percentage extraction range filtering | Pass |
| API workflow: registration, login, application, upload vali-dation, role restriction, and status locking | Pass |
| **Total** | **7/7 passed** |

repository does not include a labelled image dataset, groundtruth transcriptions, OCR character error rate (CER), word error rate (WER), processing-time measurements, or user acceptance results. Therefore, no numerical OCR accuracy or admission efficiency improvement is claimed.

## X. FUTURE WORK

The current version is a local minimum viable prototype. OCR is limited to image formats and depends on the local Tesseract installation. The verification logic is keyword and field matching rather than a trained machine learning model. It does not validate certificate signatures, QR codes, issuing authority records, security marks, or image tampering. Eligibility rules are illustrative and must be replaced with approved institutional rules. Fee fields are maintained through administrator actions; there is no live payment gateway. Email/mobile OTP, automated SMS/email notifications, scholarship processing, chatbot support, merit list generation, and seat allocation are not implemented in the supplied code and should not be represented as completed functionality. Future work should include PDF OCR, a labelled dataset of permitted dummy or consented documents, CER/WER evaluation, multilingual OCR testing, more robust layout aware field extraction, and a documented human review protocol. Document authenticity would require independent evidence such as trusted issuer verification, digital signatures, QR validation, or a dedicated tamper detection method. Further work should also include database migrations, secure document storage, encryption, malware scanning, rate limiting, audit policy review, backup and retention controls, and integration testing with institution approved fee and eligibility rules.

## XI. CONCLUSION

This paper presented a local college admission management software prototype that combines student application handling, document uploads, application status tracking, administrator review, and OCR-assisted text consistency checks. The implementation uses React/Vite, FastAPI, SQLAlchemy, SQLite by default, Pillow, and Pytesseract with a local Tesseract engine. The backend test suite passed all seven available tests, providing initial functional evidence for the implemented API and rule based verification logic. The system is not a production certified platform and does not prove document authenticity. Its principal value is a transparent workflow in which OCR output and rule-based checks assist an administrator while keeping final decisions under human control.

## ACKNOWLEDGMENT

The authors acknowledge the guidance provided during the project by Prof. D. S. Nale, Department of Computer Engineering, SVPM College of Engineering, Malegaon BK, Baramati. The authors should review, verify, and take responsibility for every technical statement and reference in this manuscript. If this AI assisted draft is used for a publication submission, the authors must follow the target venue's current policy on disclosure of generative AI assistance.

## REFERENCES

[1] R. Jayadevan, S. R. Kolhe, P. M. Patil, and U. Pal, "Offline recognition of Devanagari script: A survey," IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 41, no. 6, pp. 782–796, Nov. 2011, doi: 10.1109/TSMCC.2010.2095841.

[2] K. Ntirogiannis, B. Gatos, and I. Pratikakis, "Performance evaluation methodology for historical document image binarization," IEEE Transactions on Image Processing, vol. 22, no. 2, pp. 595–609, Feb. 2013, doi: 10.1109/TIP.2012.2219550.

[3] M. A. Souibgui and Y. Kessentini, "DE-GAN: A conditional generative adversarial network for document enhancement," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 44, no. 3, pp. 1180–1191, Mar. 2022, doi:10.1109/TPAMI.2020.3022406.

[4] S. Karthikeyan, A. G. Seco de Herrera, F. Doctor, and A. Mirza, "An OCR postcorrection approach using deep learning for processing medical reports," IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no.5, pp.2574–2581, May 2022, doi:10.1109/TCSVT.2021.3087641.

[5] J. Memon, M. Sami, R. A. Khan, and M. Uddin, "Handwritten optical character recognition (OCR): A comprehensive systematic literature review (SLR)," IEEE Access, vol. 8, pp. 142642–142668, Jul. 2020, doi: 10.1109/ACCESS.2020.3012542.

[6] X. Situ, T. Liu, H. Yao, C. Qin, and X. Zhang, "Document-image perceptual hashing for content authentication," IEEE Transactions on Big Data, vol. 11, no. 6, pp. 3474–3487, Dec. 2025, doi:10.1109/TBDATA.2025.3596854.