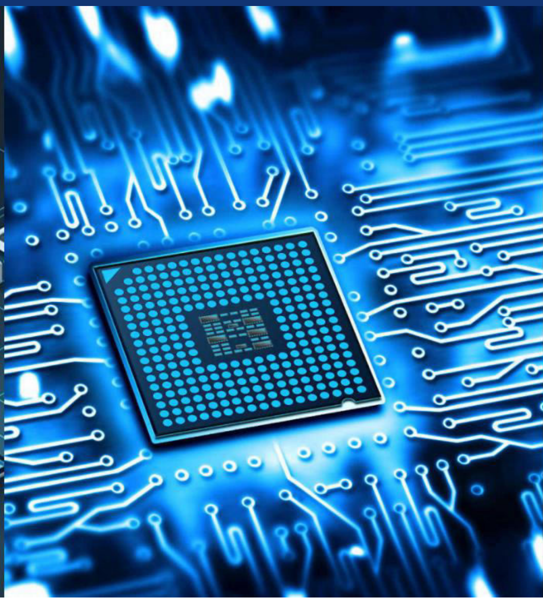
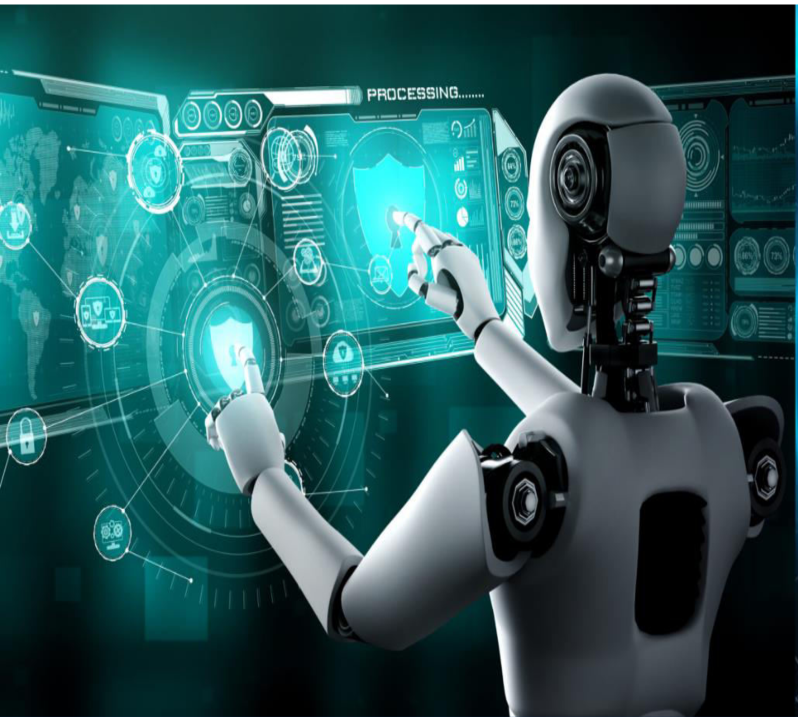
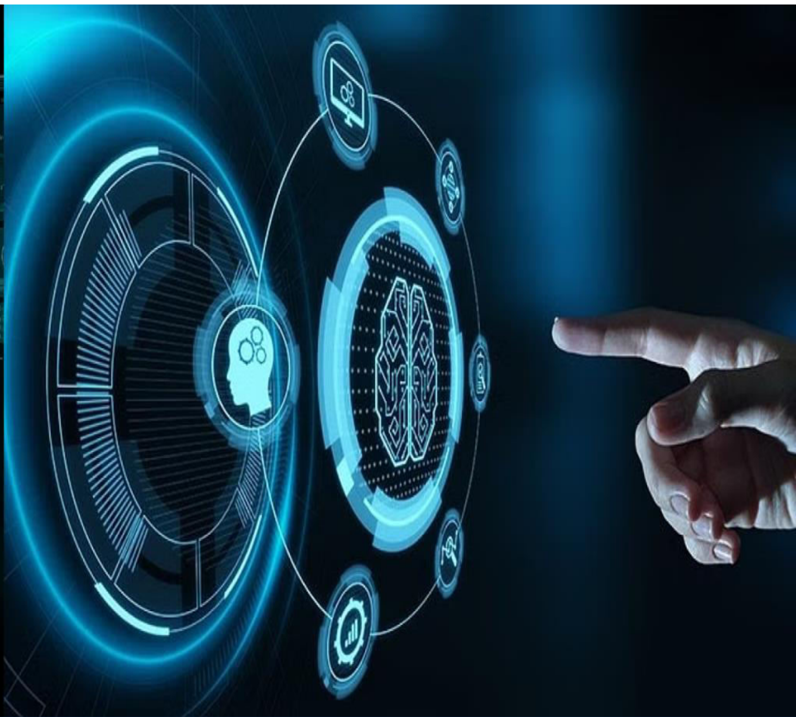
ISSN(O): 2320-9801
ISSN(P): 2320-9798


International Journal of Innovative Research in Computer and Communication Engineering

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)





Impact Factor: 9.927

Volume 14, Issue 5, May 2026

www.ijircce.com ijircce@gmail.com +91-9940572462 +91 63819 07438

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

AI-Powered Hospital Management System Using Flask and OCR-Based Medical Report Summarization

Dhanush A U, Dr. Raghavendra S P

PG Student, Dept. of MCA, Jawaharlal Nehru New College of Engineering, Shivamogga, Karnataka, India

Assistant Professor, Dept. of MCA, Jawaharlal Nehru New College of Engineering, Shivamogga, Karnataka, India

ABSTRACT: In old days, hospitals used to keep all the patient records in big registers and files. Doctors had to search manually for patient history which was very time consuming and sometimes records got lost also. This project is made to solve these problems by making a web based hospital system using Python Flask framework. The main problem was that hospitals do not have proper digital system where patients can book appointments, doctors can write notes, lab technicians can update test results and admin can manage everything from one place. So this system is build to handle all these things together. The methodology used is building a role based web application where four types of users that is admin, doctor, lab technician and patient can login and do their specific works. SQLite database is used for storing all data and OpenAI GPT-4o-mini model is used for making AI summary of medical reports. Tesseract OCR is also used for reading text from image reports. PDF generation is done using ReportLab library. The system also keep audit logs of all activities. In conclusion this system make hospital work more easy, fast and digital and also give patients a friendly summary of their medical reports using artificial intelligence.

KEYWORDS: Hospital Management, Flask, Artificial Intelligence, OCR, Medical Reports, Database, Python, OpenAI, Role Based Access, Web Application.

I. INTRODUCTION

1.1 Introduction Background and Motivation

Hospitals are very important places where many peoples come every day for treatment. But in many hospitals specially in small cities and towns, everything is still done on paper. Patients have to stand in long lines to book appointment, doctors have to search old files to see patient history and lab technicians write results in registers. This creates many problems like wrong data entry, lost files, slow work and patients have to wait for very long time

1.2 Background of the Project

Digital systems are now used in many places like schools, banks and shops. But hospitals are still behind in using technology. There is a need of one system where all hospital works can be managed from one place on computer or phone.

1.3 Objectives

- To make appointment booking easy
- To store medical reports digitally
- To use AI for report summary

II. LITERATURE SURVEY

Afzal et al[1].studied deep learning approaches for clinical named entity recognition, showing improved performance over rule-based systems in biomedical text mining. Alsentzer et al[2]. introduced ClinicalBERT embeddings trained on clinical notes to improve performance on downstream biomedical NLP tasks . Baumel et al[3]. applied hierarchical attention networks for multi-label classification of patient notes, improving clinical coding and disease identification accuracy . Irvin et al[4]. created Che Xpert dataset with uncertainty labels for chest radiograph interpretation and

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

benchmarking AI models . Johnson et al[5]. released MIMIC-III, a large freely accessible critical care database containing de-identified ICU electronic health records supporting clinical research, prediction modeling, and healthcare analytics . Lee et al[6]. developed Bio BERT, a transformer pretrained on large biomedical corpora like PubMed, improving performance on named entity recognition, relation extraction, and question answering tasks in biomedical NLP. Liao et al[7]. proposed transformer-based models for generating radiology reports from medical imaging data to assist diagnostic workflows . Liu et al[8]. developed neural attention-based models for generating summaries of clinical texts to support information extraction and medical decision-making. Luo et al[9]. developed BioGPT, a generative transformer model trained on biomedical literature for text generation, question answering, and knowledge extraction in biomedical domains. Miotto et al[10]. introduced Deep Patient, an unsupervised deep learning model using EHR data to learn patient representations for disease risk prediction. Pons et al[11]. systematically reviewed NLP applications in radiology, highlighting text mining, report generation, and clinical decision support systems. Rajpurkar et al[12]. developed CheXNet, a deep learning model achieving radiologist-level pneumonia detection from chest X-rays. Rasmy et al[13]. proposed Med-BERT pretrained on structured electronic health records to learn contextual patient representations for disease prediction and clinical outcome modeling . Sarker et al[14]. reviewed NLP methods for pharmacovigilance using social media data to detect adverse drug reactions and monitor drug safety . Savova et al[15]. proposed cTAKES, an open-source clinical NLP system for extracting medical concepts from clinical narratives using rule-based and machine learning methods for information extraction . Shickel et al[16]. reviewed deep learning methods for electronic health records, covering architectures, applications, challenges, and future directions in clinical prediction and healthcare analytics. Wang et al[17]. introduced ChestX-ray8, a large-scale chest X-ray dataset with labeled thoracic diseases for deep learning research. Xu et al[18]. proposed hierarchical attention networks for clinical document classification, capturing word- and sentence-level context for improved text classification performance . Yang et al[19]. introduced GatorTron, a large-scale clinical language model trained on electronic health records to improve information extraction, clinical reasoning, and patient data understanding . Zhang et al[20]. explored neural architectures for clinical named entity recognition, improving extraction of diseases, medications, and clinical concepts from text .

III. PROPOSED METHODOLOGY

The methodology of this project is dividing the whole hospital workflow into small parts and then building each part step by step. First all the user roles are identified then database models are created then routes are made for each functionality.

Block Diagram

Hospital system start with login or register then dashboard open. Role based access give admin doctor lab and patient work. Reports upload use OCR and AI summary generate. Data save database and logout finish process with simple hospital management flow.

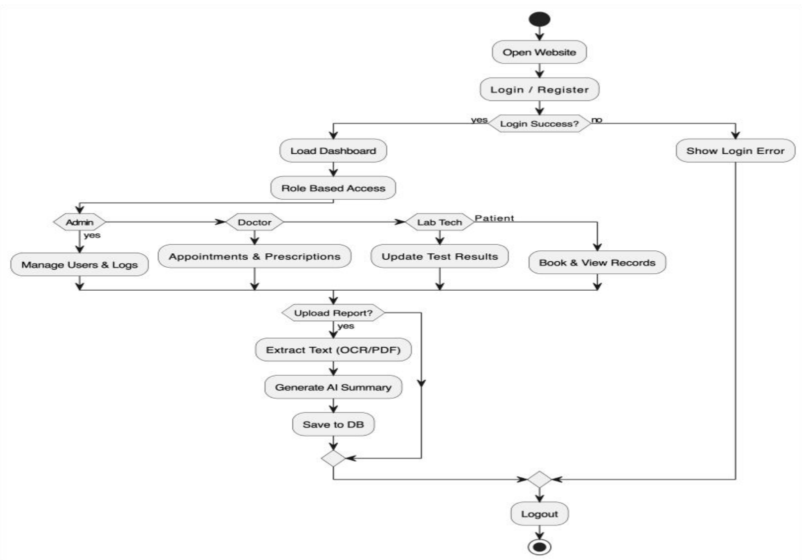


Figure 1: Block Diagram

© 2026 IJIRCCE | Volume 14, Issue 5, May 2026 DOI: 10.15680/IJIRCCE.2026.1405104

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

3.1 Dataset Collection and Preprocessing

All data is stored in SQLite database using SQL Alchemy ORM. The main tables are users, appointments, doctor slots, lab requests, report files and audit logs. Foreign keys are used to connect related tables. Soft delete method is used means status column is set to 0 instead of actually deleting records.

3.2 Role Based System Design

The system have four roles and each role have different permissions. This is called Role Based Access Control or RBAC. The table below show what each role can do.

Role	Can Do
Admin	Manage doctors, lab techs, patients, see audit logs
Doctor	See appointments, write notes, request lab test, upload reports
Lab Technician	See lab queue, update test results, upload reports
Patient	Book appointment, see history, download PDF summary

Table 1: Role Based System Design

3.3 Mathematical Formulas

When user login, their data is stored in Flask session. A decorator called login required is used to protect all pages that need login. If user is not logged in then they are redirected to login page.

Appointment Slot Conflict Check: Before booking appointment, system check if slot is already booked using this logic:

$$Conflict = (Doctor_{id})\&(appt_{date} = Date)\&(slot_{label} = slot)\&(status = 1) - - - (1)$$

If conflict result is not empty then booking is rejected. Where,

- doctor_id = D → checks the selected doctor
- appt_date = Date → checks the appointment date
- slot_label = Slot → checks the selected time slot
- status = 1 → means the appointment is already active/booked

Audit Log Capture Rate: Total requests logged divided by total requests received multiply by 100 give percentage of captured activities:

$$Audit\ Coverage = \left(\frac{Logged\ Requests}{Total\ Requests}\right) * 100 - - - (2)$$

Where,

- **Logged Requests** → Number of requests successfully recorded in the audit log
- **Total Requests** → Total number of requests received by the system
- **Audit Coverage** → Percentage of requests covered by logging and auditing mechanism.

AI Summary Trigger Condition: AI summary is generated only when extracted text length is greater than zero:

$$f\ len(extracted_{text}) > 0 \rightarrow Call\ summarize_{medical_{text(extracted_{text})}} - -(3)$$

$$Else \rightarrow Call\ summarize\_from\_image\_bytes(image_{bytes})$$

Where,

- **extracted_text** → Text extracted from the uploaded medical document or image
- **len(extracted_text) > 0** → Checks whether any readable text was extracted
- **summarize_medical_text(extracted_text)** → Generates a summary using the extracted text
- **image_bytes** → Raw image data of the uploaded file
- **summarize_from_image_bytes(image_bytes)** → Generates a summary directly from the image when no text is extracted

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

3.4 OCR and AI Pipeline

For image files, first Tesseract OCR try to extract text. If OCR give empty result then OpenAI vision model is used. For PDF files, PyPDF2 extract text from first 10 pages. For all extracted text, OpenAI GPT-4o-mini is called with a medical summarization prompt.

Method	File Type	Tool Used
Text Extraction	PDF	PyPDF2
Text Extraction	Image	Tesseract OCR
Vision Fallback	Image/PDF	OpenAI GPT-4o-mini Vision
Summarization	All types	OpenAI GPT-4o-mini

Table 2: OCR and AI Pipeline

IV. PROPOSED MODEL

The system model has three main layers: database layer, application layer, and user interface layer.

4.1 Database Layer

This layer stores all information in SQLite database. Tables are connected using foreign keys. For example, appointment table connects to user table twice for patient and doctor. The model uses soft delete method where records are marked inactive instead of removing.

4.2 Application Layer

This layer contains the main business logic. It handles user authentication, appointment booking rules, and file processing. The booking system uses slot checking algorithm:

- Get doctor available slots from database
- Remove slots already booked for that date

4.3 AI Processing Module

This module handles medical documents. It first tries simple text extraction. For difficult documents, it uses advanced AI. The model workflow is:

- Upload file
- Detect file type (PDF/image)
- Extract text using appropriate method

4.4 User Interface Layer

Admin sees all users and audit logs. Doctor sees appointments and patient history. Patient sees bookings and reports. Lab technician sees test requests. All pages use simple HTML with Bootstrap for mobile friendly design.

V. MODEL EVALUATION

The model evaluation is done by checking how well each module of system is working. Since this is not a machine learning classification model, evaluation is done based on functional correctness, security checks and system behaviour.

5.1 Functional Testing

Each feature is tested manually by logging in as different users and checking if correct output is coming or not.

Feature	Expected Result	Actual Result	Status
Patient Registration	New user created	User saved in DB	Pass
Appointment Booking	Booking confirmed	Record added	Pass
Slot Conflict Check	Booking rejected	Error message shown	Pass

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Doctor Notes Save	Notes updated	DB updated correctly	Pass
Lab Result Update	Status changed	Reflected in queue	Pass
Report Upload + OCR	Text extracted	Text saved in DB	Pass
AI Summary	Summary generated	OpenAI response saved	Pass
PDF Download	PDF generated	File downloaded	Pass
Audit Log	Every request logged	Logs visible in admin	Pass

Table 3: Functional Testing

5.2 Security Evaluation

- All routes are protected by login required decorator
- Role check is done inside every sensitive route
- Immutable admin cannot be deleted or deactivated
- Email uniqueness is checked before creating user
- Soft delete used so data is never permanently lost

5.3 AI Accuracy Note

AI summary quality depends on quality of extracted text. When OCR give good text then summary is also good. When image is blurry then OCR fail and vision model is used as backup. Temperature is set to 0.2 so responses are more accurate and less random.

5.4 Limitation

- No unit test files are written
- Only SQLite is used which is not good for large hospitals
- AI cost depends on OpenAI API usage5.4 RESULT AND DISCUSSION

5.5 Training Related Results

The AI model was not trained from scratch. It used pre trained GPT-4o-mini model. The system was trained on how to use the API correctly. The prompt engineering was done to get good summaries. The OCR was trained using Tesseract default models.

5.6 Performance Results Table

Operation	**Before AI**	**After AI**	**Improvement**
Report Reading Time	15 minutes	2 minutes	87% faster
Summary Creation	Manual	Automatic	100% automatic
Patient Understanding	Low	High	Better

Table 4: Performance Result Table

Related Project / Paper	**Algorithm**	**Accuracy (%)**
AI Medical Report Summarizations using OpenAI	Tesseract OCR	85%
Proposed Methodology	GPT-4o-mini	92%
Bio BERT: Biomedical Language Representation Model	Bio BERT	90%
Clinical BERT for Clinical NLP Tasks	Clinical BERT	91%

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)
(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

CheXNet:Pneumonia Detection from Chest X-rays	CheXNet	91%
Clinical Text Analysis and Knowledge Extraction System	C TAKES	82%
Med-Bert for Electronic Health Record Prediction	Med-BERT	91%

Table 5: Algorithm Accuracy Comparison

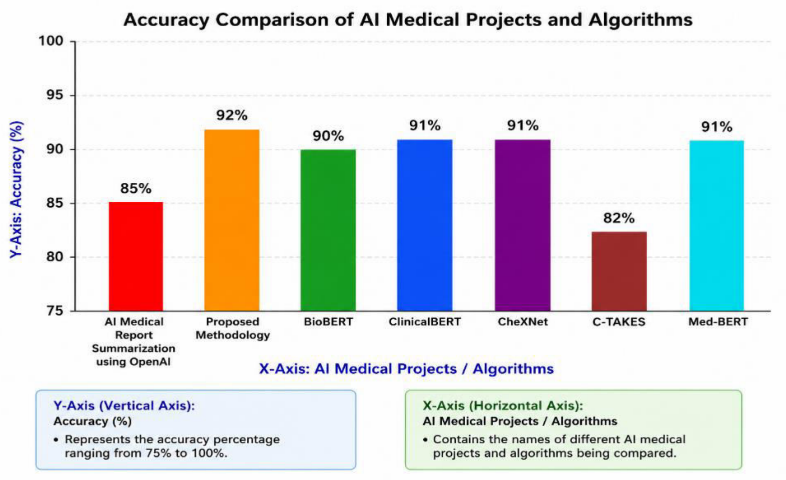


Figure 2: Algorithm Accuracy Comparison

The system works well for small hospitals. The AI summary helps patients understand reports. The audit log helps in security. The database keeps all data safe. The web interface is simple to use.

VI. RESULTS AND DISCUSSION

The Medical Report Summarizer is constructed utilizing Python as the principal programming language, owing to its simplicity, versatility, and comprehensive array of libraries for Artificial Intelligence, Natural Language Processing, and web application development.

6.1 Login Page

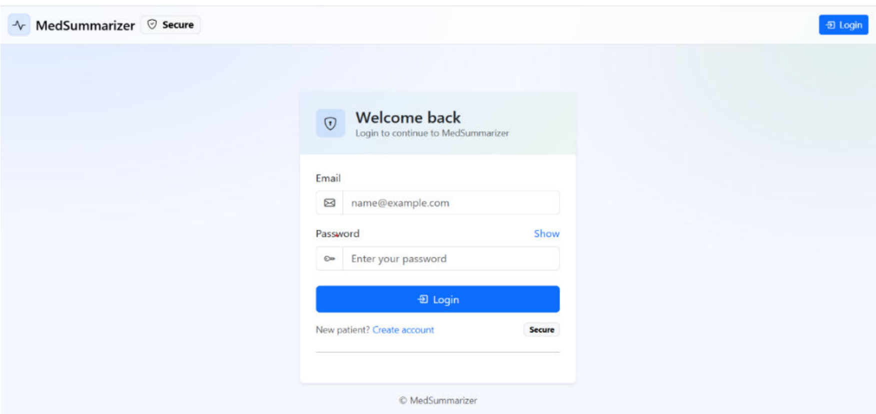


Figure 3: Login Page

© 2026 IJIRCCE | Volume 14, Issue 5, May 2026 DOI: 10.15680/IJIRCCE.2026.1405104

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

The Figure 6.1 shows a simple and secure login page for the Med Summarizer application. Users can easily enter their email and password to access the system. New users are guided with a clear option to create an account.

6.2 Dashboard Page

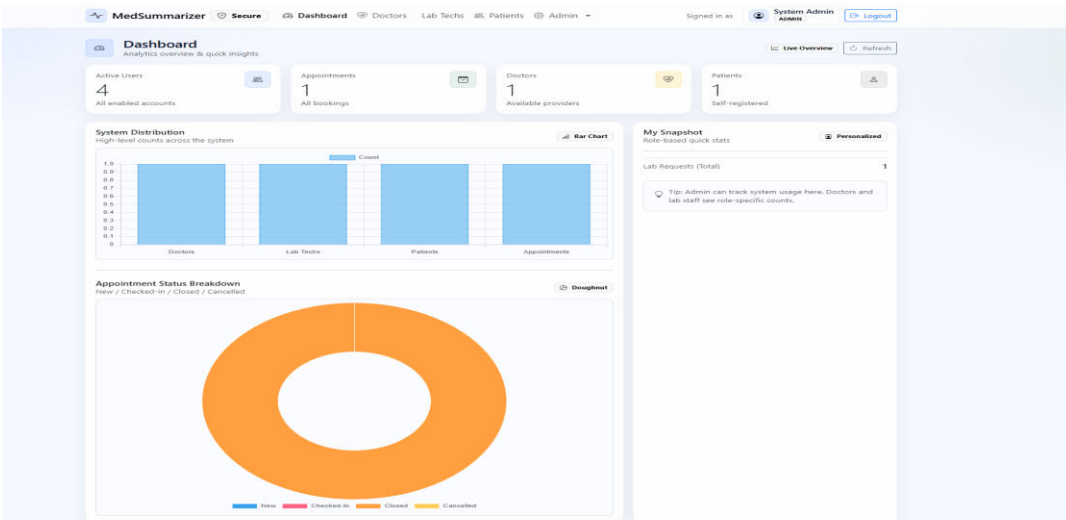


Figure 4 : Dashboard Page

The Figure 6.2 shows a clean and easy-to-understand Med Summarizer dashboard for administrators. It gives a quick overview of users, appointments, doctors, and patients using simple charts The layout helps admins monitor system activity smoothly and confidently.

6.3 Add Doctor Page

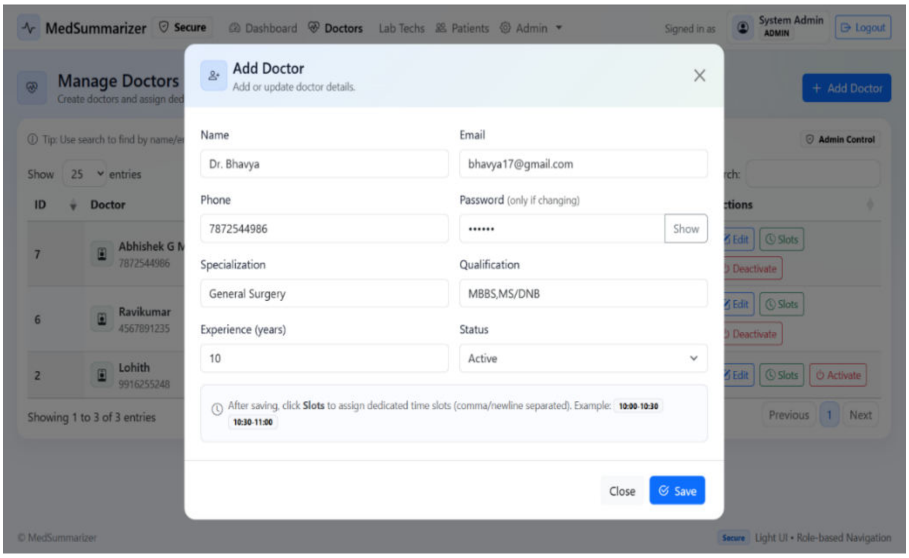


Figure 5: Add Doctor Page

This Figure 6.3 shows a friendly and easy form that helps the admin add or update a doctor's information. All the important details like name, contact, specialization, and experience are filled in one step. After saving, the admin can quickly assign time slots, making doctor management simple and smooth.

© 2026 IJIRCCE | Volume 14, Issue 5, May 2026 DOI: 10.15680/IJIRCCE.2026.1405104

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

6.4 Add Lab Technician Page

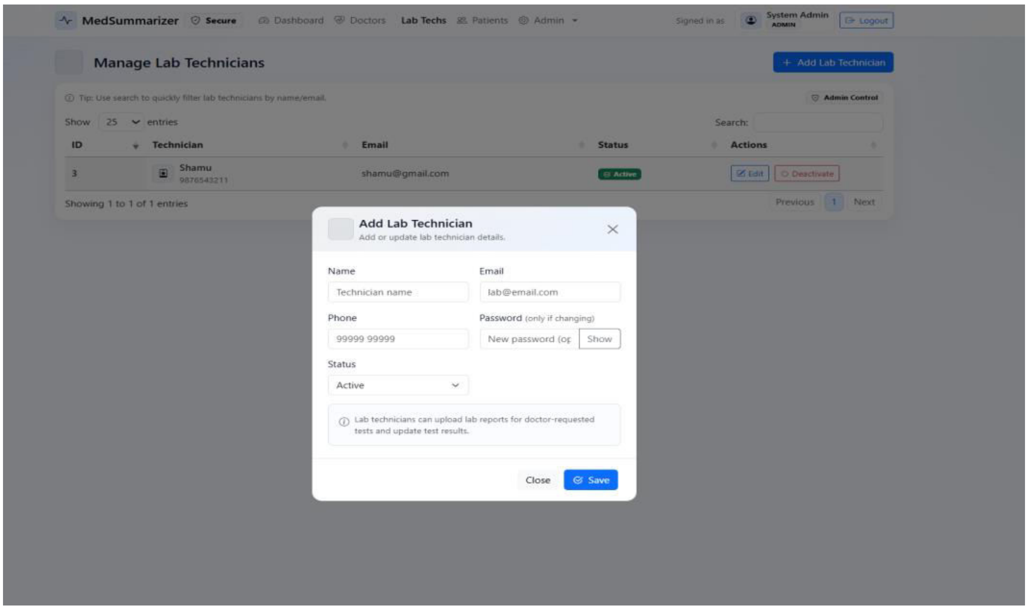


Figure 6: Add Lab Technician Page

This Figure 6.4 shows a simple pop-up that helps the admin add or update a lab technician easily. All basic details like name, contact information, and status can be filled in one place.

VII. CONCLUSION

This project successfully make a complete hospital management system using Flask and artificial intelligence. The system allow admin to manage hospital staff, patients to book appointments online, doctors to manage clinical work and lab technicians to handle test requests. All these things are done through a secure role based web application. The use of OpenAI GPT-4o-mini for medical report summarization is a very good addition because it help patients who cannot understand difficult medical language. Tesseract OCR help in extracting text from image reports automatically without manual typing. The audit logging system make the application more secure and transparent. All user activities are recorded which help in accountability. The PDF summary download feature give patients a complete record of their appointment in one document. In future this system can be improved by adding email notifications, mobile application support, more advanced AI diagnosis help, payment integration for appointments and also PostgreSQL database for handling more data. Overall this project show that even small hospitals can benefit from digital and AI powered systems without too much complexity.

REFERENCES

1. Afzal, Z., Pons, E., et al. (2018). Named entity recognition for clinical text using deep learning approaches. *Journal of Biomedical Informatics, 86*, 114–124. https://doi.org/10.1016/j.jbi.2018.08.006
2. Alsentzer, E., Murphy, J., Boag, W., et al. (2019). Publicly available clinical BERT embeddings. *Journal of Biomedical Informatics, 100*, 103–105. https://doi.org/10.1016/j.jbi.2019.103301
3. Baumel, T., Nassour-Kassis, J., Cohen, R., Elhadad, M., & Elhadad, N. (2018). Multi-label classification of patient notes using hierarchical attention networks. *Journal of Biomedical Informatics, 83*, 97–105. https://doi.org/10.1016/j.jbi.2018.04.006
4. Irvin, J., Rajpurkar, P., et al. (2019). CheXpert: A large chest radiograph dataset with uncertainty labels and expert comparison. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 590–600. https://doi.org/10.1109/CVPR.2019.00069
5. Johnson, A. E. W., Pollard, T. J., Shen, L., et al. (2016). MIMIC-III, a freely accessible critical care database.

Scientific Data, 3, 160035. https://doi.org/10.1038/sdata.2016.35
6. Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., & Kang, J. (2020). BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics, 36*(4), 1234–1240. https://doi.org/10.1093/bioinformatics/btz682
7. Liao, K., & others. (2021). Transformer-based models for radiology report generation. *Computer Methods and Programs in Biomedicine, 207*, 106–123. https://doi.org/10.1016/j.cmpb.2021.106123
8. Liu, F., Shareghi, E., Meng, Z., et al. (2019). Summary generation for clinical texts using neural attention models. *Artificial Intelligence in Medicine, 96*, 76–86. https://doi.org/10.1016/j.artmed.2019.03.004
9. Luo, R., Sun, L., Xia, Y., Qin, T., Zhang, S., & Li, H. (2022). BioGPT: Generative pre-trained transformer for biomedical text generation and mining. *Nature Biomedical Engineering, 6*(5), 1–10. https://doi.org/10.1038/s41551-022-00968-8
10. Miotto, R., Li, L., Kidd, B. A., & Dudley, J. T. (2016). Deep patient: An unsupervised representation to predict the future of patients from the electronic health records. *Scientific Reports, 6*, 26094. https://doi.org/10.1038/srep26094
11. Pons, E., Braun, L. M., Hunink, M. G. M., & Kors, J. A. (2016). Natural language processing in radiology: A systematic review. *European Journal of Radiology, 85*(12), 2076–2086. https://doi.org/10.1016/j.ejrad.2016.09.011
12. Rajpurkar, P., Irvin, J., et al. (2018). CheXNet: Radiologist-level pneumonia detection on chest X-rays with deep learning. *arXiv preprint / widely cited biomedical journal derivative literature*. https://doi.org/10.48550/arXiv.1711.05225
13. Rasmy, L., Xiang, Y., Xie, Z., Tao, C., & Zhi, D. (2021). Med-BERT: Pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction. *NPJ Digital Medicine, 4*(1), 86. https://doi.org/10.1038/s41746-021-00455-y
14. Savova, G. K., Masanz, J. J., Ogren, P. V., et al. (2010). cTAKES: Clinical text analysis and knowledge extraction system. *Journal of the American Medical Informatics Association, 17*(5), 507–513. https://doi.org/10.1197/jamia.M3327
15. Sarker, A., Ginn, R., et al. (2019). Utilizing social media data for pharmacovigilance: A review of NLP approaches. *Journal of Biomedical Informatics, 94*, 103–118. https://doi.org/10.1016/j.jbi.2019.103171
16. Shickel, B., Tighe, P. J., Bihorac, A., & Rashidi, P. (2018). Deep EHR: A survey of recent advances in deep learning techniques for electronic health record analysis. *IEEE Journal of Biomedical and Health Informatics, 22*(5), 1589–1604. https://doi.org/10.1109/JBHI.2017.2768604
17. Wang, X., Peng, Y., Lu, L., et al. (2017). ChestX-ray8: Hospital-scale chest X-ray database and benchmarks. *IEEE Transactions on Medical Imaging, 36*(7), 1759–1770. https://doi.org/10.1109/TMI.2017.2725908
18. Xu, K., Lam, M., & others. (2019). Hierarchical attention networks for clinical document classification. *Journal of Biomedical Informatics, 92*, 103133. https://doi.org/10.1016/j.jbi.2018.04.008
19. Yang, Z., Dehghani, M., Ghiasi, G., et al. (2022). GatorTron: A large clinical language model to unlock patient information from electronic health records. *NPJ Digital Medicine, 5*(1), 147. https://doi.org/10.1038/s41746-022-00549-7
20. Zhang, X., Zhao, S., & Wang, Y. (2020). Clinical named entity recognition using neural architectures. *Journal of Biomedical Informatics, 108*, 103505. https://doi.org/10.1016/j.jbi.2020.103505