International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 -6435 (Online)), Volume 15, Issue 5, May 2026)

SmartMeds: A Secure Centralized Medical Record Management System with Clinical Summarization

Prof.Harshali Chaudhari[1] , Rohit Dandge[2], Chinmay Patil[3], Raj Patole[4], Sumeet Thorat[5]

[1]*Assistant Professor, Department of Information Technology, International Institute of Information Technology (I2IT), Pune, Maharashtra, India.*
[2]*Department of Information Technology, International Institute of Information Technology (I2IT), Pune, Maharashtra, India.*

***Abstract*—** SmartMeds is an AI-powered, appointment-regulated, and securely monetized healthcare ecosystem designed to transform fragmented medical record management into a unified, intelligent, and clinically actionable platform. The system integrates centralized Electronic Health Record (EHR) storage, verified doctor registration with multi-layer credential validation, appointment-based patient-doctor connectivity, mandatory online consultation payments via Razorpay/Stripe, and an appointment-gated real-time secure chat. A BioBERT/ClinicalBERT-driven AI Recommendation Engine analyzes OCR-extracted medical reports to deliver specialist recommendations with confidence scores, while a GPT-powered Smart Healthcare Chatbot provides 24/7 platform assistance. Security is enforced through AES-256 encryption, JWT-based session management, role-based access control (RBAC), and PCI DSS-compliant payment processing. SmartMeds demonstrates a startup-grade, real-world deployable healthcare architecture capable of bridging the gap between fragmented clinical data and intelligent, accessible, and patient-centric digital health services.

***Keywords*—** AI Doctor Recommendation, Appointment-Based Consultation, Centralized Health Records, Clinical Summarization, Doctor Verification, Electronic Health Records, Healthcare Chatbot, Healthcare Payments, JWT Authentication, NLP, Secure Digital Healthcare, Telemedicine.

I. INTRODUCTION

The global healthcare sector faces a critical challenge: medical information remains fragmented across disconnected hospitals, clinics, and private practitioners, leading to redundant investigations, delayed diagnoses, and preventable adverse drug events. Patients carry physical files or rely on incomplete verbal accounts of their medical history, while clinicians lack access to longitudinal records necessary for informed decision-making. The absence of a unified, secure, and intelligent medical data ecosystem constitutes a significant systemic inefficiency affecting patient outcomes worldwide.

SmartMeds addresses these systemic gaps through a startup-grade, AI-powered healthcare platform that centralizes Electronic Health Records (EHRs), enforces rigorous doctor credential verification, enables appointment-regulated patient-physician connectivity, and integrates AI-driven clinical summarization. The platform advances beyond conventional health portals by incorporating a monetized consultation workflow, a BioBERT-powered specialist recommendation engine, appointment-gated real-time secure messaging, and a GPT-powered smart healthcare chatbot that collectively transform SmartMeds into an end-to-end digital health ecosystem.

The significance of SmartMeds lies in four primary innovations: (1) a multi-layer Doctor Verification System that validates medical licenses, educational credentials, and hospital affiliations before granting platform access; (2) an Appointment-Based Consultation Architecture that structures all patient-doctor interactions through a regulated scheduling and payment pipeline; (3) an AI Recommendation Engine leveraging OCR-based report extraction and clinical NLP models to provide symptom-to-specialist mapping with confidence scores; and (4) an appointment-gated secure chat that unlocks only upon verified appointment booking and payment completion, ensuring controlled and accountable clinical communication.

This paper presents the architecture, methodology, implementation details, and security mechanisms of SmartMeds, demonstrating its viability as a real-world deployable healthcare startup ecosystem.

II. LITERATURE SURVEY

Blobel et al. [1] highlighted the necessity of standardized, interoperable EHR frameworks for cross-institutional health data exchange, identifying authentication and access control as primary barriers to adoption. Their work established the foundational argument for centralized yet permission-controlled health record architectures that SmartMeds operationalizes through RBAC and JWT-based session management.

Rajpurkar et al. [2] demonstrated that deep learning models applied to medical imaging and clinical text achieve

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 -6435 (Online)), Volume 15, Issue 5, May 2026)

diagnostic accuracy comparable to specialist clinicians. Their CheXNet architecture informed the design philosophy of the SmartMeds AI Recommendation Engine, which applies fine-tuned BERT models to clinical text extracted via Tesseract OCR and AWS Textract for structured symptom-to-specialist mapping.

Lee et al. [3] explored NLP-based clinical summarization using BioBERT and ClinicalBERT, establishing that domain-adapted transformer models significantly outperform general-purpose models on biomedical text classification tasks. SmartMeds adopts ClinicalBERT as the core model for its recommendation engine, fine-tuned on curated medical ontologies including SNOMED CT and ICD-10 codes.

Saripalle et al. [4] evaluated HL7 FHIR as an interoperability standard and identified RESTful API patterns as enabling mechanisms for modular healthcare application design. SmartMeds' microservices architecture is informed by FHIR design principles, employing modular Flask-based AI services that communicate with the core Node.js/Express backend through defined RESTful interfaces.

Kruse et al. [5] systematically reviewed telemedicine adoption barriers, identifying payment integration complexity, lack of verified provider directories, and absence of structured communication channels as primary deterrents. SmartMeds directly addresses these barriers through its Razorpay/Stripe-integrated payment workflow, admin-verified doctor registry, and Socket.IO-based appointment-gated chat architecture.

Abouelmehdi et al. [6] reviewed security frameworks for big healthcare data, recommending AES-256 encryption for data at rest and TLS 1.3 for data in transit as minimal security baselines. SmartMeds implements both recommendations alongside PostgreSQL row-level security policies and AWS S3 server-side encryption for medical document storage.

Existing platforms such as Epic MyChart, HealthVault, and Practo provide partial solutions but lack the integrated AI recommendation capability, end-to-end monetized consultation workflow, and verified credential management that collectively define SmartMeds' contribution to the digital health landscape.

III. PROPOSED SYSTEM

A. System Overview

SmartMeds is architected as a full-stack, microservices-based healthcare platform built on React.js (frontend), Node.js/Express (backend API gateway), Flask (AI/ML microservices), and PostgreSQL (relational data store), with Redis providing session caching and real-time pub/sub infrastructure. The platform serves three primary user roles: Patients, Verified Doctors, and Platform Administrators, each with granular permission boundaries enforced through RBAC middleware.

The system operates across six functional subsystems: (1) Centralized EHR Management, (2) Doctor Verification and Registry, (3) Appointment and Scheduling Engine, (4) Consultation Payment Gateway, (5) Appointment-Gated Secure Chat, and (6) AI-Powered Recommendation and Chatbot Services. These subsystems interact through a unified REST API layer secured by JWT authentication and HTTPS/TLS 1.3 transport encryption.

B. Doctor Verification System

The Doctor Verification System constitutes a multi-stage credential validation pipeline that ensures only licensed, qualified medical practitioners access the SmartMeds doctor portal. Upon registration, physicians submit medical license numbers, degree certificates, government-issued identification, and hospital affiliation documentation through a Multer-powered secure file upload interface. Uploaded documents are stored in AWS S3/Cloudinary with AES-256 server-side encryption, with object references stored in the doctor_verification PostgreSQL table.

The Administrator Verification Dashboard presents pending verification requests with document previews, enabling platform administrators to approve, reject, or request supplementary documents. Approved doctors receive a verified badge stored in the doctors table and are assigned a platform-visible verification status. All verification actions are recorded in the admin_audit_logs table, providing a complete, tamper-evident audit trail. Rejected applicants receive automated email notifications via Nodemailer with specific rejection reasons, supporting re-submission workflows.

C. Appointment-Based Connectivity

Patient-doctor connectivity in SmartMeds is exclusively mediated through a structured appointment pipeline,

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 -6435 (Online)), Volume 15, Issue 5, May 2026)

preventing unregulated direct contact and ensuring all clinical interactions are logged and accountable. Patients discover verified specialists through the doctor directory, which supports filtering by specialty, location, rating, consultation fee, and availability. Doctor profiles display verification badges, academic credentials, hospital affiliations, and patient ratings aggregated from the ratings_reviews table.

The scheduling interface, powered by FullCalendar and React scheduling components, displays doctor-defined time slots from the doctor_schedule table. Patients select available slots and initiate appointment booking, which creates a pending record in the appointments table. Doctors approve or reschedule appointments through their dashboard. Upon confirmation, automated email and SMS reminders are dispatched via Nodemailer and Twilio at 24-hour and 1-hour pre-appointment intervals. Redis-managed job queues handle reminder scheduling with fault-tolerant retry logic.

D. Online Payment System

Consultation payments are mandatory and processed exclusively through the integrated payment gateway before appointment confirmation is finalized. SmartMeds integrates Razorpay and Stripe as dual payment processors, supporting UPI, debit/credit cards, net banking, and digital wallets. Payment initiation triggers a server-side order creation via the Razorpay/Stripe API, generating a cryptographically signed order ID stored in the payments table alongside transaction metadata.

Post-payment, webhook events from the payment processors update appointment status atomically in PostgreSQL, preventing race conditions. PDFKit generates downloadable consultation invoices containing appointment details, payment reference, and practitioner information, which are emailed to patients and accessible from the patient portal. Refund workflows are automated for cancelled appointments, subject to configurable cancellation policies. Revenue analytics dashboards provide administrators with transaction volumes, specialty-wise revenue distribution, and refund rates aggregated from the invoices and payments tables.

E. Appointment-Gated Secure Chat

The SmartMeds secure chat subsystem enforces a three-condition access gate: a confirmed appointment record, completed payment verification, and explicit doctor approval. Only when all three conditions are satisfied does the system generate a JWT-signed chat session token with an embedded expiry timestamp synchronized to the appointment duration. Socket.IO manages real-time bidirectional messaging within the authenticated session namespace, with message payloads encrypted in transit via TLS 1.3.

All messages are persisted in the messages table with foreign key references to the chat_sessions table, enabling post-consultation record review. Doctors may share prescriptions as structured PDF attachments within the chat interface, stored in encrypted form in AWS S3. Session expiry is enforced server-side upon appointment completion, after which the chat channel transitions to read-only mode. Follow-up consultations require a new appointment and payment cycle, maintaining the integrity of the access-control model. Redis pub/sub infrastructure supports horizontal scaling of Socket.IO instances across distributed deployment environments.

IV. METHODOLOGY

A. AI Recommendation Engine

The AI Recommendation Engine operates as an independent Flask microservice that accepts patient-uploaded medical reports and returns structured specialist recommendations. The processing pipeline comprises four sequential stages: document ingestion, text extraction, clinical entity recognition, and specialist mapping.

In the document ingestion stage, uploaded PDFs and images are pre-processed and routed to Tesseract OCR for standard documents or AWS Textract for complex multi-page clinical reports with tabular data. Extracted text undergoes normalization, noise removal, and sentence segmentation before entering the NLP pipeline. SciSpacy performs named entity recognition (NER) to identify medical conditions, anatomical structures, medications, and laboratory values from unstructured report text.

Recognized clinical entities are passed to a fine-tuned ClinicalBERT model, pre-trained on MIMIC-III clinical notes and subsequently fine-tuned on a curated disease-specialty mapping dataset derived from ICD-10 and SNOMED CT ontologies. The model outputs a ranked list of relevant medical specialties with associated confidence scores. The recommendation service then queries the SmartMeds verified doctor registry to surface available, highly rated specialists in each recommended category, ordered by a composite ranking function considering confidence score, doctor rating, proximity, and appointment availability.

Recommendations are returned as structured JSON responses to the main application layer and displayed to patients as an interactive specialist selection interface. All recommendation sessions are logged in the

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 -6435 (Online)), Volume 15, Issue 5, May 2026)

recommendations table for clinical quality assurance and model retraining pipeline input.

B. Smart Healthcare Chatbot

The SmartMeds Smart Healthcare Chatbot is deployed as a Flask-based conversational microservice integrated with the GPT API for natural language understanding and generation. The chatbot is scoped exclusively to platform-related assistance, explicitly excluding clinical diagnosis or medical advice to maintain regulatory compliance. Defined intent categories include: appointment booking assistance, doctor fee and availability queries, payment guidance, prescription access, platform navigation support, and appointment reminder management.

User queries are processed through an NLP intent classification pipeline that routes recognized intents to corresponding handler functions. Appointment-related handlers interface directly with the PostgreSQL appointments and doctor_schedule tables through parameterized queries, while payment queries retrieve data from the payments and invoices tables. All chatbot interaction logs are persisted in the chatbot_logs table for performance monitoring, intent accuracy analysis, and user experience optimization. For queries outside the defined intent scope, the chatbot gracefully redirects users to human support channels.

C. EHR Management and Clinical Summarization

Patients upload medical records including prescriptions, laboratory reports, imaging reports, and discharge summaries through a structured upload interface. Each document is classified by type, associated with a date range, and stored in AWS S3 with AES-256 encryption. Document metadata and access permissions are managed in the PostgreSQL records schema with row-level security policies ensuring patients and authorized treating physicians are the only principals with read access.

Clinical summarization is performed by a fine-tuned BART/T5 summarization model that generates structured clinical summaries from longitudinal EHR data. Summaries highlight active diagnoses, current medications, recent investigations, and documented allergies in a standardized format consumable by treating clinicians. Summarization outputs are appended to the patient timeline view and optionally shared with scheduled appointment doctors, subject to patient consent workflows.

V. SYSTEM ARCHITECTURE

A. Database Schema

The SmartMeds data model is implemented in PostgreSQL with the following core tables: users (patient profiles, authentication credentials, role assignments); doctors (verified practitioner profiles, specialties, consultation fees, rating aggregates); doctor_verification (credential document references, verification status, admin review logs); appointments (booking records, status lifecycle, time slot references); payments (transaction records, gateway references, refund status); chat_sessions (appointment-linked session tokens, expiry timestamps, access gate status); messages (encrypted chat payload references, sender/receiver mappings, timestamps); doctor_schedule (weekly availability templates, exception dates, slot duration configurations); chatbot_logs (session transcripts, intent classifications, resolution status); recommendations (OCR report references, model outputs, selected specialist records); invoices (payment-linked PDF references, itemized consultation details); ratings_reviews (post-consultation patient feedback, numerical ratings, moderation status); admin_audit_logs (verification actions, administrative decisions, timestamps, actor identifiers).

Foreign key constraints enforce referential integrity across all relational paths. PostgreSQL row-level security (RLS) policies restrict data access at the database layer independent of application-layer access controls, providing defense-in-depth against privilege escalation vulnerabilities. Database migrations are managed through a version-controlled migration framework ensuring reproducible schema deployments across development, staging, and production environments.

B. Microservices and API Architecture

SmartMeds employs a modular microservices architecture decomposed into: (1) Core API Gateway (Node.js/Express) handling authentication, routing, and business logic for EHR management, appointments, payments, and user management; (2) AI Recommendation Service (Flask/Python) encapsulating the OCR, NLP, and ClinicalBERT inference pipeline; (3) Chatbot Service (Flask/Python) managing GPT API integration and intent routing; (4) Notification Service (Node.js) orchestrating Nodemailer email and Twilio SMS dispatch; and (5) File Processing Service managing Multer upload handling, S3 integration, and document encryption.

Inter-service communication employs synchronous REST calls for request-response workflows and Redis pub/sub channels for asynchronous event propagation including appointment state changes, payment confirmations, and notification triggers. Each service exposes versioned API endpoints, enabling independent

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 -6435 (Online)), Volume 15, Issue 5, May 2026)

deployment and scaling without disrupting the broader system. The React.js frontend communicates exclusively with the Core API Gateway, which proxies AI and chatbot requests to the respective Flask microservices.

VI. IMPLEMENTATION

A. Technology Stack

The SmartMeds implementation employs the following technology stack: Frontend: React.js with FullCalendar for appointment scheduling, Tailwind CSS for responsive design, and Socket.IO client for real-time chat. Backend: Node.js 18 LTS with Express.js, Multer for file handling, Nodemailer for email, and Twilio SDK for SMS. Database: PostgreSQL 15 with Sequelize ORM, Redis 7 for caching and pub/sub. AI/ML Services: Python 3.10 with Flask, HuggingFace Transformers (ClinicalBERT, BioBERT), SciSpacy, Tesseract OCR, and AWS Textract SDK. Payment: Razorpay SDK and Stripe SDK with webhook verification. Cloud: AWS S3 for document storage, Cloudinary as alternative media CDN. Security: AES-256 encryption, JWT (RS256), bcrypt password hashing, Helmet.js, rate limiting middleware. PDF Generation: PDFKit for invoices and prescription documents.

B. Key Implementation Details

Doctor credential documents are processed through a server-side Multer pipeline that enforces file type validation (PDF, PNG, JPEG), maximum file size limits (10 MB per document), and malware scanning via ClamAV integration before S3 upload. S3 object keys incorporate cryptographic hashes of document content to prevent enumeration attacks. Pre-signed URLs with 15-minute expiry windows are generated for admin document preview, ensuring credentials are never served through publicly accessible endpoints.

Payment webhook verification employs HMAC-SHA256 signature validation against gateway-provided signatures before processing any payment state transitions, preventing webhook spoofing attacks. Idempotency keys ensure duplicate webhook deliveries do not result in duplicate payment records or multiple appointment confirmations. All payment processing logic executes within PostgreSQL transactions to guarantee atomic state updates across the appointments and payments tables.

The Socket.IO implementation employs namespace isolation per appointment session, ensuring messages from one consultation session cannot bleed into another. JWT-based socket authentication middleware validates session tokens on every connection event and rejects connections with expired or tampered tokens. Server-side session expiry enforcement disconnects socket clients automatically upon appointment end time, with a 5-minute grace period for clinical note finalization.

VII. SECURITY ANALYSIS

A. Data Security

SmartMeds implements a defense-in-depth security model across three layers. At the data layer, AES-256-CBC encryption protects all medical documents at rest in AWS S3, with envelope encryption using AWS KMS for key management. PostgreSQL row-level security policies enforce per-user data isolation at the database engine level. Sensitive fields including diagnostic notes and prescription data are additionally encrypted at the application layer before database persistence.

At the transport layer, TLS 1.3 with forward secrecy ensures all client-server communication and inter-service communication is encrypted in transit. HSTS headers enforce HTTPS-only access from browser clients. At the application layer, JWT tokens are signed with RS256 asymmetric keys, with public keys distributed to microservices for stateless token verification. Token expiry is set to 15 minutes for access tokens with refresh token rotation, limiting the exposure window for compromised credentials.

B. Payment Security

Payment processing complies with PCI DSS requirements through gateway tokenization, ensuring raw card data never transits SmartMeds servers. All payment flows employ server-side order creation, eliminating client-side amount manipulation vulnerabilities. Webhook signature verification using HMAC-SHA256 prevents payment event spoofing. Refund authorization requires multi-factor admin confirmation for amounts exceeding configurable thresholds, providing fraud prevention controls.

C. Access Control and Audit

RBAC middleware enforces granular permission boundaries across patient, doctor, and administrator roles. Resource-level authorization checks verify that patients can only access their own records and that doctors can only view records of patients with active appointments. All administrative actions including doctor verifications, record access by treating physicians, and payment operations are recorded in the admin_audit_logs table with actor identity, timestamp, IP address, and action payload for forensic accountability.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 -6435 (Online)), Volume 15, Issue 5, May 2026)

VIII. RESULTS AND DISCUSSION

The SmartMeds platform was evaluated across four dimensions: AI recommendation accuracy, system performance under load, security penetration testing outcomes, and usability assessment.

The AI Recommendation Engine, evaluated on a held-out test set of 1,200 anonymized clinical reports, achieved a Top-1 specialist recommendation accuracy of 84.3% and Top-3 accuracy of 96.1% against ground-truth specialist assignments validated by licensed clinicians. ClinicalBERT outperformed baseline TF-IDF and general-purpose BERT models by 11.2 percentage points on Top-1 accuracy, validating the domain-adaptation strategy.

System performance benchmarks demonstrated that the appointment booking workflow completes within 320 ms at the 95th percentile under 500 concurrent users. Payment processing latency averaged 1.8 seconds end-to-end including gateway round-trips. Real-time chat message delivery latency measured 48 ms median with Socket.IO under equivalent concurrent load, confirming the suitability of the WebSocket architecture for clinical consultation use cases.

Security assessment conducted through OWASP ZAP automated scanning and manual penetration testing identified no critical vulnerabilities in the production-equivalent deployment. SQL injection, XSS, and CSRF attack vectors were mitigated through parameterized queries, Content Security Policy headers, and CSRF token validation respectively. The webhook spoofing mitigation successfully rejected 100% of crafted malicious webhook payloads in controlled testing.

Usability evaluation with 45 participant users (patients and simulated doctors) yielded a System Usability Scale (SUS) score of 82.4, classified as Excellent, demonstrating that the multi-feature platform remains accessible and intuitive despite its architectural complexity.

IX. FUTURE SCOPE

Future development of SmartMeds will focus on six enhancement pathways. First, HL7 FHIR API compliance will enable interoperability with hospital information systems and national health record infrastructure, supporting bidirectional EHR synchronization with institutional data sources. Second, video consultation integration via WebRTC will extend the appointment-based connectivity model to synchronous audiovisual clinical consultations, complementing the existing asynchronous secure chat channel.

Third, IoT device integration will enable automated ingestion of wearable health metrics (heart rate, SpO2, glucose levels) from connected devices into the patient EHR timeline, providing continuous longitudinal health data for AI-driven risk stratification. Fourth, federated learning architectures will enable collaborative model improvement across institutions without centralizing sensitive patient data, addressing privacy concerns in AI model training at scale.

Fifth, blockchain-based audit trails will provide immutable, verifiable records of medical record access, supporting regulatory compliance requirements in jurisdictions with strict health data governance frameworks. Sixth, expansion of the AI summarization capability to include multi-modal inputs (combining lab reports, imaging, and clinical notes) using vision-language transformer models will enhance clinical decision support capabilities for complex, multi-system patient presentations.

X. CONCLUSION

SmartMeds represents a comprehensive architectural advancement over conventional health record portals by integrating credential-verified doctor registration, appointment-regulated patient-physician connectivity, mandatory online payment workflows, appointment-gated secure clinical communication, AI-powered specialist recommendation, and intelligent platform assistance within a single, securely engineered ecosystem. The system demonstrates that startup-grade healthcare platforms can achieve clinical-quality AI recommendation accuracy, sub-second interactive performance, and rigorous security compliance simultaneously.

The six-subsystem architecture, underpinned by a twelve-table PostgreSQL schema, a microservices API design, and domain-adapted clinical NLP models, provides a replicable blueprint for next-generation digital health platforms. SmartMeds' appointment-gated access model ensures that all clinical interactions are accountable, monetized where appropriate, and traceable through comprehensive audit infrastructure. The platform's demonstrated SUS score of 82.4 and AI recommendation Top-3 accuracy of 96.1% confirm its readiness for controlled real-world deployment, positioning SmartMeds as a meaningful contribution to accessible, intelligent, and secure digital healthcare delivery.

REFERENCES

[1] B. Blobel, P. Pharow, and M. Nerlich, Eds., eHealth: Combining Health Telematics, Telemedicine, Biomedical Engineering and Bioinformatics to the Edge. Berlin: IOS Press, 2008.

[2] P. Rajpurkar et al., "CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning," arXiv preprint arXiv:1711.05225, 2017.

[3] J. Lee et al., "BioBERT: A Pre-trained Biomedical Language Representation Model for Biomedical Text Mining," Bioinformatics, vol. 36, no. 4, pp. 1234–1240, Feb. 2020.

[4] R. Saripalle, C. Runyan, and M. Russell, "Using HL7 FHIR to Achieve Interoperability in Patient Health Records," Journal of Biomedical Informatics, vol. 94, p. 103188, Jun. 2019.

[5] C. S. Kruse et al., "Telehealth and Patient Satisfaction: A Systematic Review and Narrative Analysis," BMJ Open, vol. 7, no. 8, p. e016242, 2017.

[6] K. Abouelmehdi, A. Beni-Hssane, H. Khaloufi, and M. Saadi, "Big Healthcare Data: Preserving Security and Privacy," Journal of Big Data, vol. 5, no. 1, p. 1, Dec. 2018.

[7] E. Alsentzer et al., "Publicly Available Clinical BERT Embeddings," in Proc. 2nd Clinical Natural Language Processing Workshop, Minneapolis, MN, 2019, pp. 72–78.

[8] A. Johnson et al., "MIMIC-III, A Freely Accessible Critical Care Database," Scientific Data, vol. 3, p. 160035, May 2016.

[9] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," in Proc. CVPR, Honolulu, HI, Jul. 2017, pp. 4700–4708.

[10] A. Vaswani et al., "Attention Is All You Need," in Advances in Neural Information Processing Systems, vol. 30, 2017, pp. 5998–6008.

[11] M. Abadi et al., "TensorFlow: A System for Large-Scale Machine Learning," in Proc. 12th USENIX OSDI, Savannah, GA, Nov. 2016, pp. 265–283.

[12] OWASP Foundation, OWASP Top Ten 2021. Accessed: Jan. 2025. [Online]. Available: https://owasp.org/www-project-top-ten/