IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 12, December 2025

DOI: 10.17148/IJARCCE.2025.141254

Lab Links-Intelligent Integration For Equitable Diagnostic Access

Abhishek A[1], Aditya S[2], Akshay Krishna K S[3], Darshan D Gowda[4], Abhilash L Bhat[5]

VII Sem, Dept. of Computer Science and Engineering, K. S. Institute of Technology[1]

VII Sem, Dept. of Computer Science and Engineering, K. S. Institute of Technology[2]

VII Sem, Dept. of Computer Science and Engineering, K. S. Institute of Technology[3]

VII Sem, Dept. of Computer Science and Engineering, K. S. Institute of Technology[4]

Assistant Professor, Dept. of Computer Science and Engineering, K. S. Institute of Technology[5]

Abstract: The process of medical diagnostics in developing regions remains fragmented, often requiring patients to manually locate laboratories, compare costs, and retrieve reports across multiple platforms. This paper presents Lab Link, an intelligent full-stack web application designed to simplify and digitize the end-to-end process of medical lab testing. The system allows users to securely create accounts, search for diagnostic tests, and discover nearby laboratories based on location and test availability. Each listed laboratory displays detailed information including test pricing, facilities, and user ratings, enabling data-driven selection. The application integrates Next.js for responsive client- side rendering, Cloudflare Workers for scalable backend computation, and PostgreSQL with Prisma ORM for optimized data management. JWT-based cookie authentication ensures secure and persistent sessions, while Zod validation enforces type-safe operations throughout the system. Test results are stored and retrieved in encrypted PDF format, preventing data loss and ensuring patient privacy.

I. INTRODUCTION

Accessing laboratory diagnostic services remains a fundamental yet often fragmented aspect of healthcare delivery, particularly in urban and semi-urban settings. Patients frequently encounter difficulties in locating suitable testing centers, comparing pricing, understanding test preparation guidelines, and storing or retrieving test results over time. Despite the rapid digitalization of other healthcare segments, diagnostic workflows—from test discovery to result management—remain largely manual, inefficient, and disconnected. The demand for a seamless, patient-centered platform to bridge this gap is evident. For instance, individuals undergoing routine or specialized medical tests must often visit multiple laboratories or rely on outdated information to make decisions. Furthermore, once tests are completed, results are typically delivered as physical documents or isolated digital files—both of which are prone to being misplaced, inaccessible, or insecure. As digital health records gain traction, there is a growing need to integrate diagnostic data into user-centric, reliable web platforms. The platform allows users to register, securely log in, and grant location access. Based on the test they search for, the system dynamically queries a database of labs to list facilities nearby that offer the desired diagnostic service, along with pricing, test descriptions, and preparation instructions. Additionally, patients' reports are uploaded in PDF format and persistently stored in their accounts for anytime access— minimizing the risk of data loss

II. LITERATURE REVIEW

In recent years, a range of digital health platforms have emerged to assist patients in navigating healthcare services such as appointment booking, teleconsultation, and medication tracking. However, comparatively fewer systems have been developed to streamline the experience of medical diagnostic testing — especially in terms of service discovery, cost comparison, and long-term results management. Studies have explored various digital approaches to healthcare service delivery, but most tend to focus either on hospital-level information systems or electronic health records (EHRs) used by clinicians, rather than tools designed explicitly for patient-initiated diagnostic service interactions.. Patel et al. (2021) presented a mobile application that allowed users to locate nearby pathology labs, yet the system was limited to static listings and lacked real-time filtering or location intelligence. Another study by Sharma and Kulkarni (2022) proposed a web portal for test result storage, but the platform offered no integration of lab service discovery or context-aware pre-test guidance. These systems often overlooked critical elements such as dynamic location-based listings, cost transparency, or user-accessible medical record retention in secure formats like PDFs. Additionally, prior solutions have tended to focus heavily on either frontend usability or backend data storage. A key missing component in earlier systems is the provision of pre-test instruction guidance, which can be critical in ensuring accurate diagnostic outcomes. Tests such as blood glucose, lipid profiles, and hormonal assays require specific preparations — yet this information is often scattered across websites or communicated verbally, risking misinterpretation. As noted by Chaudhary and Mehta (2020), integrating medical literacy tools into patient interfaces significantly improves compliance and health outcomes, but such integrations remain rare in lab-related digital platforms In terms of report preservation and accessibility, several platforms have proposed cloud storage of medical data. However, issues surrounding user privacy, data portability, and persistent accessibility across devices have not been addressed in full. While cloud-native architectures can offer persistent storage, they must also ensure encryption, authentication, and secure document retrieval, especially for sensitive medical records To address these limitations, our work proposes a full-stack CRUD application that not only facilitates user registration and location-enabled test discovery but also offers comprehensive features such as test pricing, pre-test instructions, lab information, and secure PDF-based report storage. Unlike previous systems that only partially implement these functionalities, our approach is integrated, user-centric, and designed to be scalable. It builds on earlier research in health informatics and patient engagement by offering a practical, deployable platform that can enhance accessibility and data continuity in outpatient diagnostic services.

III. METHODOLOGY

This project adopts a full-stack software engineering approach to develop a CRUD-based medical lab testing platform. The system is engineered to bridge the gap between patients and diagnostic facilities, integrating secure user authentication, geolocation services, cloud storage, and conversational interfaces. The methodology is broken down into the following components : User Registration and Authentication : Technology Stack**:** Authentication is managed using Firebase Authentication or Auth0. These services offer secure, out-of-the-box login systems with multi-platform SDKs (Web, Android, iOS).User Roles**:** Two primary roles are defined: Patient**:** Can search for lab tests, view nearby labs, book tests, and access stored results Lab Admin: Can upload test results, update pricing, and manage facility information. Security Measures**:** Passwords are hashed and never stored in plain text.JWTs (JSON Web Tokens) are used for maintaining authenticated sessions. Route guards are implemented to ensure role-based content visibility Location Access and Lab Matching: Matching users with the correct service is driven by real-time geolocation. Geolocation API: The browser's native navigator. geolocation API is used to retrieve the user's current latitude and longitude coordinates. Lab Data Model: Each lab record in the database contains fields for :Name, Address, Coordinates List of Available Tests, Pricing, Facility Rating, Contact Info. Search Functionality **:**On search input (e.g., "blood sugar"), the system filters labs offering that test using indexed queries. It calculates proximity using the Haversine formula or Firestore's GeoPoint queries. Map Rendering**:** Google Maps API displays pins for each matching lab, showing travel distance, estimated time, and contact options. Lab Test Metadata and Guidelines System Test Catalog: A dynamic repository of medical tests is maintained with the following attributes: such Test Name, Test Code, Description, Pre-test Instructions, Normal Ranges, Turnaround Time When a user selects a test, the UI presents do's and don'ts (e.g., "Fast for 12 hours before this test"). . PDF-Based Results Storage and Retrieval Lab Admins upload test result PDFs via a protected interface. Each file is linked to a patient's UID and test ID and timestamped. high quality, and suitable for statistical learning. WhatsApp Integration via Business API Once the data has been pre- processed, we proceed to the training phase by using typical machine learning algorithms: Backend Services and CRUD Operations We use various classification models like Random Forest, Logistic Regression, and Gradient Boosted Trees in our research. Containerization and Cloud Hosting We utilize cross-validation alongside grid and randomized search methods to refine model parameters and enhance predictive performance. Performance Evaluation: Throughout our prototype building process These instructions are retrieved dynamically and presented clearly when a user selects a test for booking. Lab administrators can update this information through an internal dashboard to keep the database current These documents are stored in a cloud storage service such as Firebase Storage or AWS S3, while their metadata (including test name, patient ID, and timestamp) is recorded in a NoSQL database like Firestore or MongoDB.

III. Containerization (With Docker): To keep environments consistent and avoid deployment issues, we use Docker to isolate each pipeline stage: Container Isolation: Every working unit i.e., data pre-processing, model training, and inference—is isolated in a separate Docker container. Dependency Management: Docker images include all dependencies, including the Python version and libraries, thus making it easier to switch between local development

ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

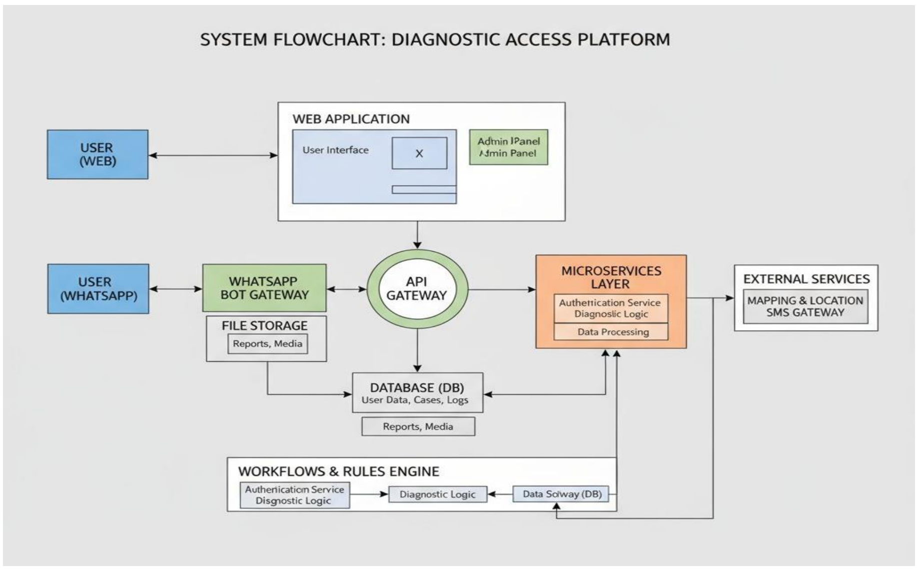
Impact Factor 8.471 ⁝⁝ Peer-reviewed & Refereed journal ⁝⁝ Vol. 14, Issue 12, December 2025

DOI: 10.17148/IJARCCE.2025.141254

and cloud deployment. Portability: The identical container image utilized for development locally can be run on cloud platforms such as AWS straight away without requiring any modification. This makes it reproducible and reduces environment-specific compatibility issues. IV. Cloud Deployment using AWS EC2 and S3: AWS hosts the trained models and associated artifacts for deployment to enable scalable and high-availability inference services: Artifact Storage: AWS S3 is used to store trained models, datasets, and logs.Model Serving: Docker containers run onAmazon Web Services Elastic Compute Cloud instances to handle inference requests through a RESTful API. Scalability: Cloud infrastructure allows for elastic scalability based on request load and system robustness. Thisdeployment method enables real-time predictions and is capable of hosting multiple simultaneous users efficiently. V.CI/CD and Automation (With GitHub Actions): Automation is a critical component that enables continuous integration and delivery of machine learning models. Source Control: All code and configurations reside in a single central GitHub repository.Trigger Mechanisms: Each time updates are pushed (i.e., new model code, schema changes, or updated datasets), GitHub Actions will trigger workflows automatically. VI. Automated Workflow: Restart Docker containers with the new code.Re-train models on the new data.Push the new Docker images to AWS and then redeploy with minimal downtime. The delivery pipeline helps so that the production environment

IV. SYSTEM ARCHITECTURE

Our lab testing platform integrates user authentication, geolocation-based search, cloud storage, and multi-platform communication into a unified and scalable web-based solution. It comprises five structured stages, supporting the full SDLC lifecycle from local experimentation to production-grade inference and observability. 1. Local Development: It is possible to build prototype data pipelines and train initial models by running web application. 2. CI/CD Orchestration: The setup uses GitHub Actions to implement automated processes for testing along with Docker builds and deployment activities on the main branch. 3. Containerization: All training and inference processes should be placed in individual Docker containers to preserve identical environments throughout different development stages. 4. Cloud Execution: The solution deploys AWS EC2 containers while utilizing S3 for both data and model storage which supports large-scale training and real-time inference operations. 5. Monitoring & Logging: AWS Cloud Watch provides monitoring features which track logs and system health along with model performance data while also sending alerts for anomalous events and failed processes.



V. CONCLUSION

We developed a comprehensive web-based platform that streamlines the process of diagnostic lab testing through secure user authentication, geolocation-based test center discovery, test-specific advisory, and digital report management. By tightly integrating location services, CRUD operations, and cloud-based data handling, our system offers a seamless and patient-friendly experience for individuals seeking lab testing services. The system architecture demonstrates how traditional CRUD applications can be extended to meet domain-specific needs, particularly in the healthcare sector, where accessibility, accuracy, and information retention are vital The solution is scalable and adaptable, allowing future integration with hospital information systems, wearable health devices, or real-time test result APIs. Additional enhancements—such as lab appointment booking, health insurance integration, and AI-driven

DOI: 10.17148/IJARCCE.2025.141254

health advisory—could further increase the system's utility. This project demonstrates how well-designed CRUD systems, when combined with geolocation, automation, and cloud capabilities, can significantly improve operation.

REFERENCES

[1]. M. Satyanarayanan, "The Emergence of Edge Computing," Computer, vol. 50, no. 1, pp. 30–39, 2017.
[2]. Cloudflare, "Cloudflare Workers: Serverless Applications at the Edge," Cloudflare Documentation,2023.
[3]. A. Sheth, "Serverless Computing and Edge AI: The Future of Scalable Web Systems," IEEE Internet Computing, vol. 25, no. 2, pp. 63–70, 2021.
[4]. Prisma Data, "Prisma ORM Documentation," 2024.
[5]. PostgreSQL Global Development Group, "PostgreSQL 16 Documentation," 2024.
[6]. C. D. Manning, P. Raghavan, and H. Schütze, Introduction to Information Retrieval. Cambridge University Press, 2008.
[7]. D. Crockford, "JSON Web Token (JWT): RFC 7519," IETF RFC, 2015.
[8]. Zod Library, "Zod: TypeScript-first SchemaValidation," npm Documentation, 2024.
[9]. R. C. Martin, Clean Architecture: A Craftsman'sGuide to Software Structure and Design. Prentice Hall,2018.
[10]. Next.js, "Next.js Framework Documentation,"Vercel, 2024.