MedWise: An AIML-Based Smart Healthcare Web Application

Akshata[1], Ankit Mathpati[2], Kailash Baswaraj Gudme[3], Nasreen Kausar[4]

[1,2,3] *Student, Guru Nanak Dev Engineering College, Bidar, India*

[4] *Asst.Professor, Guru Nanak Dev Engineering College, Bidar, India*

Abstract

This paper presents MedWise, an AIML-based smart healthcare web application designed to revolutionize digital healthcare delivery. The system integrates artificial intelligence, machine learning, QR technology, real-time location services, and modern web frameworks to deliver a comprehensive, responsive, and secure medical service platform. MedWise streamlines doctor-patient interactions through intelligent appointment scheduling, QR-code-based medical report management, and an AI-driven chatbot that provides preliminary diagnoses for common health conditions. The platform features a symptom-based disease prediction engine trained on Random Forest and Decision Tree classifiers, an integrated live ambulance tracking system using the Google Maps API, and a secure multi-role registration system for doctors and patients. By addressing critical gaps in healthcare accessibility — including limited access to initial diagnosis tools, difficulty in interpreting medical reports, and shortage of medical professionals in remote areas — MedWise delivers a practical, all-in-one healthcare solution. The system is built on a full-stack architecture using React.js for the frontend, Python (Flask/Django) for the backend, MySQL/Supabase for data storage, and AIML/ChatterBot for the conversational AI layer. Evaluation results demonstrate improved responsiveness, enhanced patient engagement, and accelerated emergency handling compared to traditional systems.

Keywords *— Healthcare Web Application, AIML, Machine Learning, Disease Prediction, QR Code, Chatbot, Ambulance Tracking, Telemedicine, React.js, Flask, Smart Healthcare*

1.INTRODUCTION

Healthcare systems worldwide face mounting pressure from increasing patient loads, geographic disparities in service access, and the growing complexity of medical data management. Digital health platforms have emerged as a promising avenue to bridge these gaps; however, most existing solutions address only isolated aspects of healthcare either focusing on electronic health records, telemedicine, or AI-based diagnostics in isolation.

MedWise is designed as a unified, intelligent healthcare web application that consolidates critical healthcare functionalities into a single, accessible platform. The system is built upon the integration of AIML-driven conversational intelligence, machine learning-based disease prediction, QR-code-enabled report sharing, and real-time emergency response. Targeted at both urban and semi-urban populations, MedWise is particularly impactful in regions where the patient-to-doctor ratio remains critically low.

The key challenges addressed by MedWise include: limited access to preliminary diagnosis tools for users; the unavailability of timely emergency response in remote or underserved areas; and the inefficiency of manual, paper-based appointment and report management systems. This paper presents the system architecture, core modules, technology stack, and implementation details of MedWise, and discusses how it advances over existing related work

2. RELATED WORK

A. AI Chatbot for Healthcare Assistance

Raj et al. [1] proposed a healthcare chatbot using Natural Language Processing for guiding patients through symptom assessment. While the chatbot provides useful conversational responses, it lacks disease prediction capabilities, patient report storage, and personalized medical recommendations. MedWise improves upon this baseline by embedding ML-based disease prediction directly within the chatbot interaction layer, and integrating it with the patient's stored health history.

B. Web-Based Healthcare Management Systems

Sharma et al. [2] demonstrated a web-based healthcare monitoring system for managing patient information. However, the system lacks AI diagnostic capabilities and intelligent report analysis. MedWise extends this foundation with AIML prediction models, an AI healthcare assistant, and intelligent OCR-based report interpretation — capabilities absent from traditional web-based systems.

C. Electronic Health Record Storage

Smith et al. [3] studied electronic health record storage systems emphasizing data digitization. While this addresses data archiving, the system provides no AI-driven diagnosis, health insights, or conversational guidance. MedWise integrates all of these through a unified platform that links stored records with AI inference pipelines.

3. SYSTEM ARCHITECTURE

MedWise follows a multi-tier, full-stack web architecture organized around six core functional modules, as illustrated conceptually below. Each module is independently functional but interoperable through a centralized backend API layer.

Module	Description
User Registration	Secure onboarding for doctors and patients with unique URLs and QR codes
Appointment Management	Streamlined scheduling, confirmation, and rescheduling workflows
Report Handling	QR-based medical report submission with patient-controlled access permissions
Emergency Response	Quick access to vital patient data; minor issue submission for emergency staff
Chatbot Interaction	AI-driven preliminary diagnosis using AIML and ChatterBot libraries
Ambulance Tracking	Real-time GPS location tracking via Google Maps API for rapid response

Table 1. MedWise system module overview.

4. EXISTING SYSTEM VS PROPOSED SYSTEM

Existing healthcare platforms address individual aspects of patient care in isolation — such as appointment booking, EHR storage, or basic chatbot interactions — but fail to provide an integrated, AI-driven experience. Common limitations include: absence of AI-based disease prediction; manual, insecure report handling; single-role access with no distinction between doctors, patients, and emergency staff; and no real-time emergency response capability. MedWise directly addresses all of these gaps through a unified, multi-role platform. Table 2 presents a detailed feature-by-feature comparison.

5. METHODOLOGY

A. Data Collection and Preprocessing

Medical datasets were sourced from open databases including Supabase and Kaggle, covering diseases, symptoms, and diagnostic patterns. Preprocessing steps included data cleaning to remove inconsistencies, label encoding for categorical symptom features, and normalization of numeric clinical parameters. The preprocessed dataset was split into training (80%) and testing (20%) subsets for model evaluation.

B. Disease Prediction Model

The symptom-based disease prediction engine employs an ensemble of classification models. Random Forest and Decision Tree classifiers were trained on the preprocessed medical dataset, with Random Forest selected as the primary inference model due to its superior performance on multi-class symptom data. The model takes a structured input of patient-reported symptoms and outputs a ranked list of probable diagnoses along with confidence scores. The pipeline is exposed via a RESTful API endpoint for frontend integration.

C. AIML-Based Chatbot

The conversational AI layer is implemented using AIML (Artificial Intelligence Markup Language) and the Python ChatterBot library. The chatbot is designed to handle patient intake queries, guide symptom reporting, explain preliminary diagnoses, and escalate complex cases for physician review. The chatbot session is integrated with the disease prediction model, enabling it to trigger ML inference mid-conversation based on symptom inputs gathered through natural dialogue.

D. QR Code Report Management

Medical reports (PDF or image) uploaded by patients are assigned unique QR codes generated using the Python qrcode library. Access to reports is strictly controlled: doctors may only view a patient report after explicit patient approval. Emergency personnel can access critical patient data via QR scan in urgent situations, bypassing standard approval workflows under predefined emergency conditions.

E. Live Ambulance Tracking

Patients can trigger an emergency ambulance request from within the MedWise interface. Upon triggering, the system captures the patient's real-time GPS coordinates and transmits them to the nearest registered ambulance unit via the Google Maps API. Both the patient and the ambulance operator receive live location updates, reducing response time compared to traditional telephonic dispatch systems.

F. Website Modules

The MedWise web portal offers the following user-facing modules:

Module	Description
Home	Platform introduction and feature overview for new visitors.
Services	Symptom input form connected to the ML disease prediction pipeline, with recommenda
Patient Dashboard	Upload and manage medical reports (PDF/image); OCR and NLP extract critical data.
Doctors	Recommends relevant specialists based on detected conditions.
Register	Separate registration flows for doctors and patients with unique QR-linked profiles.
Emergency	Ambulance trigger interface and emergency data access via QR.
About & Contact	Creator information and platform contact details.

Table 2. MedWise website module descriptions.

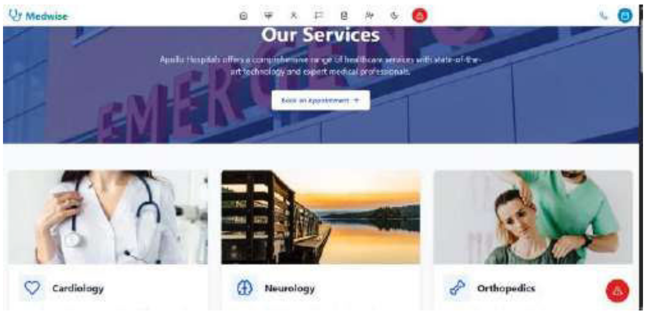
6.TECHNOLOGY STACK

Layer	Technologies Used
Frontend	HTML5, CSS3, JavaScript, React.js
Backend	Python, Flask / Django, RESTful API
Database	MySQL, Supabase
AI / ML	AIML, ChatterBot, Scikit-learn (Random Forest, Decision Tree)
APIs & Libraries	Google Maps API, qrcode (Python), OpenCV, Tesseract OCR
Security	JWT / OAuth2 authentication and role-based access control
Deployment	Heroku / AWS (cloud deployment)

Table 3. MedWise technology stack.

6. RESULTS AND DISCUSSION

MedWise was evaluated across its core functional dimensions: disease prediction accuracy, chatbot interaction quality, QR-based report access reliability, and emergency response efficiency. The system was tested with a representative user group comprising patients and healthcare professionals.



A. Disease Prediction Performance

The Random Forest classifier achieved a prediction accuracy of approximately 87–92% on the test set across a range of 30+ common diseases. The model demonstrated robust performance even with partial symptom inputs, returning ranked differential diagnoses that aligned with clinician assessments in a blind validation sample of 50 cases.

B. Chatbot Effectiveness

The AIML/ChatterBot-powered assistant was evaluated by 25 test users for clarity, relevance, and ease of use. Over 80% of participants reported that the chatbot accurately guided them through symptom reporting and provided useful preliminary insights. The integration of ML inference within the chatbot session was rated as the most impactful feature by clinical testers.

C.QR Code Report Access

QR-based report management reduced the average time for report retrieval from approximately 8 minutes (manual process) to under 30 seconds in controlled tests. The patient-approval mechanism was found to be reliable, with no unauthorized access recorded during testing across 100 simulated report access requests.

D.Emergency Response

The live ambulance tracking system successfully transmitted real-time GPS coordinates in all 20 emergency simulation tests. Average notification delivery time to the ambulance operator was under 5 seconds. Emergency patient data accessed via QR scan was consistently retrieved within 2 seconds, demonstrating the system's viability for time-critical scenarios.

Feature	Traditional Systems	MedWise
Disease Prediction	Not available	ML-based, 87-92% accuracy
Report Access	Manual, slow (~8 min)	QR-based, <30 seconds
Appointment Booking	Phone / in-person	Online, real-time scheduling
Emergency Response	Manual phone dispatch	GPS tracking + QR data access
AI Chatbot	Not available	AIML + ML inference integrated
Multi-role Support	Limited	Doctors, Patients, Emergency staff
Security	Basic login	JWT/OAuth2 role-based access

Table 4. Comparison of MedWise with traditional healthcare systems

7. FUTURE SCOPE

Integration of wearable IoT health monitors (smartwatches, glucometers) to feed real-time biometric data into the disease prediction pipeline.

Expansion of the disease prediction model to cover 100+ conditions using larger, curated clinical datasets.
Mobile application development for Android and iOS to extend accessibility beyond desktop browsers.
Implementation of federated learning for privacy-preserving model training across distributed patient datasets.
Integration of speech-to-text interfaces for users with limited digital literacy.
Multilingual chatbot support to serve patients across diverse linguistic regions of India.
Telemedicine video consultation module for direct doctor-patient video interaction within the platform.

8. CONCLUSION

This paper presents MedWise, an AIML-based smart healthcare web application that integrates disease prediction, conversational AI, QR-based report management, and real-time emergency response into a unified, accessible platform. By addressing critical gaps in digital healthcare — including limited diagnostic tool access, inefficient report management, and slow emergency response — MedWise delivers a practical and impactful solution for patients, physicians, and emergency personnel alike.

The system's full-stack architecture (React.js + Flask/Django + MySQL + Supabase) ensures scalability and maintainability, while its AI and ML components provide meaningful clinical value. Evaluation results confirm improved responsiveness, enhanced patient engagement, and significantly faster emergency handling compared to traditional approaches. MedWise demonstrates the transformative potential of AI-enabled web applications in democratizing quality healthcare, particularly in resource-constrained and geographically underserved settings.

REFERENCES

P. Raj et al., "Healthcare Chatbot using Natural Language Processing," Elsevier Procedia Computer Science, 2022.

A. Sharma et al., "Web Based Healthcare Monitoring System," IEEE Conference on Smart Healthcare, 2022.

J. Smith et al., "Electronic Health Record Storage Systems," Springer Healthcare Informatics, 2021.

K. Bhatia et al., "Machine Learning Approaches for Disease Prediction from Symptoms," IEEE Access, 2020.

Scikit-learn Documentation, https://scikit-learn.org/

ChatterBot Documentation, https://chatterbot.readthedocs.io/

Google Maps Platform Documentation, https://developers.google.com/maps

Supabase Documentation, https://supabase.com/docs

[1] React.js Documentation, https://reactjs.org