TED UNIVERSITY

SENG 492_O Senior Project

Team ForeSightAI

MedicalAI

Detailed Design Report

Spring 2025

Submission Date: 15.03.2025

Team Members:

Batuhan Mert ÖZTÜRK, 15308021268, Software Engineering

Ayman MA. HAMDAN, 10582175188, Software Engineering

Rufat NAGHIYEV, 99588787168, Software Engineering

Supervisor:

Asst. Prof. Dr. Venera ADANOVA

Jury Members:

Prof. Dr. Tansel DÖKEROĞLU

PhD. Ali BERKOL

PhD. Fırat AKBA

TED UNIVERSITY

İçindekiler

1. Introduction

1.1 Problem Definition

In modern healthcare, hospitals face significant overcrowding due to non-critical patients seeking medical attention. This leads to inefficiencies in resource allocation, longer wait times, and increased burden on healthcare professionals. Many patients visit hospitals for minor illnesses that could be managed remotely, reducing the need for in-person consultations. Conversely, critical patients may experience delays in receiving timely medical attention due to hospital congestion.

MedicalAI aims to address this issue by developing an AI-powered triaging system that assists patients in determining whether they need an in-person consultation. By leveraging Large Language Models (LLMs), the system analyzes patient symptoms, provides preliminary diagnostic insights, and recommends appropriate healthcare actions. Additionally, MedicalAI generates reports that doctors can review, enabling them to validate AI-generated recommendations efficiently. This solution optimizes hospital workflows, reduces unnecessary patient visits, and ensures that critical cases receive prioritized attention while maintaining compliance with healthcare regulations.

1.2 Purpose

The purpose of MedicalAI is to create an AI-powered healthcare assistant that enhances patient-doctor interactions by providing preliminary symptom analysis, diagnostic insights, and triaging recommendations. By utilizing Large Language Models (LLMs), the system aims to reduce unnecessary hospital visits, optimize doctors' time, and prioritize critical cases. This solution supports both patients and healthcare professionals by offering a mobile application for patients to report symptoms and a web-based platform for doctors to review AI-generated reports. MedicalAI is designed to function as a decision-support tool rather than an autonomous diagnostic system, ensuring that final medical decisions remain under the control of healthcare professionals.

1.3 Scope

MedicalAI encompasses a mobile application for patients and a web-based platform for doctors, integrated with a Large Language Model (LLM) for symptom analysis and diagnostic report generation. The system allows patients to input their symptoms via the mobile app, where the LLM processes the data to provide preliminary recommendations.

Doctors can then access and review these AI-generated reports through the web platform, modifying or validating the recommendations as needed.

The project ensures secure data handling, compliance with healthcare regulations, and seamless communication between patients and doctors. It aims to improve efficiency in medical triaging and diagnosis while reducing unnecessary hospital visits and prioritizing critical cases.

1.4 Overview

MedicalAI is an AI-powered healthcare assistant designed to enhance medical triaging and optimize patient-doctor interactions. The system integrates a Large Language Model (LLM) to analyze patient symptoms, generate preliminary diagnostic reports, and assist doctors in decision-making.

The solution consists of:

- A mobile application for patients to input symptoms, upload medical documents and receive preliminary medical guidance.
- A web-based platform for doctors to review AI-generated reports, validate diagnoses, and provide final medical recommendations.

MedicalAI prioritizes security, compliance with healthcare regulations (e.g., GDPR), and explainability, ensuring that doctors retain full control over medical decisions. By reducing unnecessary hospital visits and automating initial diagnosis steps, the system aims to improve healthcare efficiency and patient outcomes.

1.5 Definitions, Acronyms, and Abbreviations

- AI (Artificial Intelligence): The use of machine learning algorithms and computational models to mimic human intelligence in decision-making.
- LLM (Large Language Model): An advanced AI model trained on vast datasets to understand and generate human-like text, used in MedicalAI for symptom analysis and diagnosis support.
- RAG (Retrieval-Augmented Generation): A technique that enhances AI-generated responses by retrieving relevant medical documents or knowledge bases.
- Multi-Agent System: A system in which several AI agents cooperate to complete a challenging task. The multi-agent system used in MedicalAI is made up of

specialized agents that each play a different function in improving medical decision-making.

- GDPR (General Data Protection Regulation): A European Union regulation that governs data privacy and protection, ensuring patient data security in MedicalAI.
- HIPAA (Health Insurance Portability and Accountability Act): A U.S. regulation ensuring the confidentiality and security of healthcare information.
- EHR (Electronic Health Records): Digital medical records that store patient history, diagnoses, and treatment plans, integrated with MedicalAI for reference.
- UI (User Interface): The visual components of the mobile and web applications that allow users to interact with MedicalAI.
- API (Application Programming Interface): A set of protocols and tools that allow different software components to communicate, facilitating MedicalAI's integration with external systems.
- Cloud Computing: Remote servers used to store and process MedicalAI data securely, enabling scalability and real-time processing.
- Triage: The process of prioritizing patients based on the severity of their condition, a key function of MedicalAI.

1.6 References

1. MedicalAI Project Proposal – Provides an overview of the project, including its goals, objectives, and expected impact.
2. MedicalAI Project Specification Report – Defines the functional and non-functional requirements, constraints, and system features.
3. MedicalAI Project Analysis Report – Analyzes the problem domain, current challenges, and proposed AI-based solutions.
4. Health-LLM Personalized Retrieval-Augmented Disease Prediction System. Link
5. IEEE Code of Ethics – Ensures ethical AI implementation in healthcare. Link
6. GDPR Official Guidelines – Reference for data privacy and security compliance. Link
7. HIPAA Compliance Rules – U.S. regulations on patient data protection. Link
8. Relevant Research Papers on AI in Healthcare – Studies on the effectiveness of LLMs in medical diagnosis and decision support.

2. System Overview

MedicalAI is an AI-powered healthcare assistant designed to optimize patient triaging, reduce unnecessary hospital visits, and enhance doctor efficiency. The system integrates Large Language Models (LLMs) with Retrieval-Augmented Generation (RAG) to analyze patient symptoms, generate preliminary diagnostic reports, and assist doctors in decision-making.

By offering a mobile application for patients and a web-based platform for doctors, MedicalAI facilitates seamless communication while ensuring data security, compliance, and explainability.

2.1 System Objectives

- **Improve Healthcare Efficiency**: Reduce hospital congestion by helping patients determine if they need in-person consultation.
- **Assist Doctors with Decision-Making**: Generate preliminary reports to provide doctors with structured, AI-analyzed patient data.
- **Ensure Patient Data Security**: Implement role-based access control to protect sensitive medical data.
- **Enhance Accessibility & Usability**: Provide a user-friendly mobile app for patients and an efficient web interface for doctors.

2.2 System Components & Interactions

1. Mobile Application for Patients

- Patients report symptoms in natural language via chat.
- AI asks follow-up questions for better understanding.
- AI generates a preliminary report with possible diagnoses and recommendations.
- Patients receive guidance on whether they need to visit a doctor.

2. Web Application for Doctors

- Doctors log in securely and view patient cases.

- AI-generated reports provide symptom analysis and potential diagnoses.
- Doctors can modify, approve, or reject AI-generated recommendations.
- Doctors can flag critical cases for immediate attention.
- Supports patient record storage, doctor notes, and follow-up management.

3. AI Processing Module (LLM & RAG & Agent-Based Analysis)

- Uses LLMs to analyze text-based symptom descriptions.
- Implements RAG to retrieve relevant medical knowledge for enhanced decision-making.
- Generates structured diagnostic reports for doctor review.
- Ensures explainability, allowing doctors to understand how AI reached its conclusions.

4. Secure Database & Cloud Infrastructure

- Stores patient history, chat logs, and reports securely.
- Implements end-to-end encryption & role-based access (GDPR & HIPAA compliance).

2.3 System Workflow (Data Flow & Interactions)

Step 1: Patient Symptom Input & AI Processing

- Patient submits symptoms via mobile app.
- AI-agents asks clarifying questions based on symptoms.
- AI processes data and generates a preliminary diagnostic report.

Step 2: Doctor Review & Validation

- Doctor logs into the web platform and reviews AI-generated reports.
- The report contains:
  - Patient symptoms
  - Potential diagnoses
  - Suggested next steps (e.g., visit required)
- Doctor can edit or approve the report and send it back to the patient.

3. Design Considerations

3.1Assumptions and Dependencies

- **AI Model Accuracy**: The effectiveness of the AI-driven triaging system relies on the accuracy and reliability of the Large Language Model (LLM). It is assumed that the model has been worked with multi-agent system and RAG on extensive medical datasets and will provide clinically relevant insights.
- **Database Availability**: The system depends on a structured database (PostgreSQL, MySQL) for patient records and a vector database (Pinecone/FAISS) for AI-driven searches. Downtime in these databases could impact service delivery.
- **Cloud Services**: MedicalAI is hosted on a secure cloud environment, ensuring scalability and availability. It assumes continuous uptime and compliance with security standards.
- **Doctor Validation**: The AI-generated reports are not autonomous decisions; they are subject to review and validation by medical professionals.

3.2 Design Constraints

- **Data Privacy & Legal Compliance**: The system must adhere to ensuring encryption, secure authentication, and restricted data access.
- **AI Explainability**: The model must provide transparent reasoning for its diagnoses to ensure doctors can understand and trust AI-generated insights.
- **Interoperability**: MedicalAI must integrate seamlessly with existing hospital Electronic Health Records (EHR) and API-based medical services.

3.3 Performance Constraints

- **Response Time**: The AI processing should return preliminary diagnostic reports within **5 seconds** for optimal user experience.
- **Scalability**: The system should support a high number of concurrent users without significant performance degradation.
- **Concurrent Requests**: The infrastructure should handle at least **100,000 API requests per minute** efficiently.

3.4 Reliability

- **Failover Mechanisms**: Redundant server instances and automatic failover configurations are implemented to prevent downtime.

3.5 Usability

- **User-Friendly Interface**: The mobile app for patients should have an intuitive chat-based interaction, while the web app for doctors should provide structured reports with editable fields.

3.6 Portability & Extensibility

- **Cloud-Based Deployment**: Designed for cloud hosting to ensure flexibility and easier maintenance.
- **Future Expansion**: The system should allow additional AI models, external data sources, and new healthcare features to be integrated

---

4. Data Design

4.1 Data Description

MedicalAI processes various types of medical data to assist doctors in diagnosing diseases and recommending treatments. The system primarily utilizes three data sources:

1. **Medical Records:**
   - Patient symptoms, past diagnoses, and treatment history.
   - Collected from user inputs and stored securely.
   - Used for AI-based disease prediction and medical report generation.
2. **UMLS Data (Unified Medical Language System):**
   - A vast database containing standardized medical terminologies.
   - Includes diseases, symptoms, treatments, and medical conditions.
   - Extracted from sources like **SNOMED CT, ICD-10, RxNorm** for structured medical knowledge.

3. **AI-Generated Reports:**
    - Created based on **symptom analysis and medical database lookups**.
    - Summarizes potential diseases, confidence levels, and suggested treatments.
    - Designed to assist doctors in reviewing and validating AI predictions.

Each of these data types plays a crucial role in **MedicalAI's decision-making process**, ensuring accurate and meaningful medical insights.

4.2 Data Dictionary

The MedicalAI database is structured to store and manage medical records, disease information, and AI-generated reports efficiently. Below is an overview of the primary database tables and their attributes.

4.2.1 Patients

Column	Type	Description
patient_id	UUID	Unique identifier for each patient.
name	String	Patient's full name.
age	Integer	Patient's age.
gender	Enum	Male or Female

4.2.2 Symptom Analysis

Column	Type	Description
record_id	UUID	Unique identifier for each record.
patient_id	UUID	Links to the patient table.
symptoms	JSON	List of reported symptoms.
predicted_disease	String	AI-predicted disease name.
confidence	Float	AI confidence level (0-1).

4.2.3 UMLS Data

Column	Type	Description
cui	String	Unique Concept Unique Identifier from UMLS.
name	String	Name of the disease, symptom, or treatment.
type	Enum	Disease, Symptom, Treatment.
definition	Text	Standardized medical description.

4.2.4 AI-Generated Reports

Column	Type	Description
report_id	UUID	Unique identifier for each report.
patient_id	UUID	Links to the patient table.
analysis_date	DateTime	Timestamp of the report.
report		

4.3 Vectorized Data for Semantic Search in MedicalAI

MedicalAI utilizes **vectorized data** to enable fast and accurate **semantic search** for disease prediction and medical insights. This approach allows the system to retrieve **the most relevant diseases, symptoms, and treatments** from large medical datasets, such as **UMLS, SNOMED CT, and RxNorm**.

Vectorized data refers to **textual medical information transformed into numerical embeddings**. These embeddings are stored in a **vector database (e.g., Pinecone, Weaviate, or Milvus)**, allowing for **fast similarity search** and **retrieval-augmented generation (RAG)** in AI-driven medical analysis.

MedicalAI generates vectorized data for:

Diseases & Symptoms → Semantic representation of illnesses and their related symptoms.
AI Reports & Doctor Notes → Embeddings of past AI-generated reports for personalized recommendations.
Medical Knowledge Base (UMLS & PubMed) → Vectorized references to medical literature.

Field	Type	Description
id	String(UUID)	Unique identifier for each vector entry.
embedding	Array (Float)	Numerical vector representation of the data.
metadata	JSON	Additional details (disease name, source, etc.).

5. System Architecture

5.1 Architectural Design

MedicalAI follows a **microservices-based architecture** that separates key components into independently deployable units.

High-Level System Architecture:

- **Frontend Layer:**
  - **Mobile App** (Android Kotlin) for patient interaction. Jetpack compose is used
  - **Web App** (Vue.js) for doctors
- **Backend Layer:**
  - **API Gateway** (FastAPI) for handling client requests
  - **Authentication Service** (OAuth/JWT for secure login)
  - **AI Processing Module** (LLM & RAG & Agent-based diagnosis generation)
  - **Data Storage** (PostgreSQL or MySQL for structured data, Pinecone or Qdrant for vector-based searches)
  - **Logging & Monitoring** (ELK Stack for tracking system health)

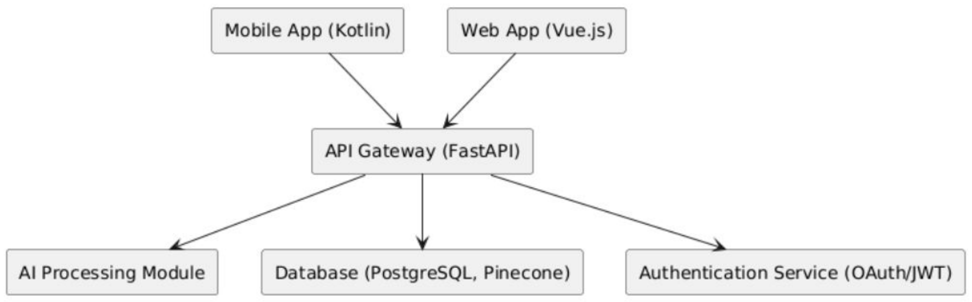


Figure 1

5.2 Description of Components

1. **Mobile App (Android Kotlin)** - Allows patients to submit symptoms and receive preliminary AI insights.
2. **Web Platform (Vue.js)** - Enables doctors to review and modify AI-generated reports.
3. **AI Processing Module** - Handles symptom analysis, retrieval-augmented generation (RAG), and medical report creation.
4. **Database Layer** - Stores structured patient records and vectorized medical knowledge for AI-based retrieval.
5. **Security & Compliance Module** - Implements end-to-end encryption and role-based access control.

5.3 Dynamic Behavior

Step 1: Symptom Submission

- Patient submits symptoms through chat-based UI.
- AI asks follow-up questions for clarification.
- AI generates an initial diagnostic report.

Step 2: Doctor Review

- Doctor logs into the web platform and views the AI-generated report.
- Doctor modifies/validates recommendations and finalizes the diagnosis.

Step 3: Patient Follow-Up

- Patients receive notifications for medical actions (e.g., tests, medication reminders).
- Doctors can track patient progress over time.

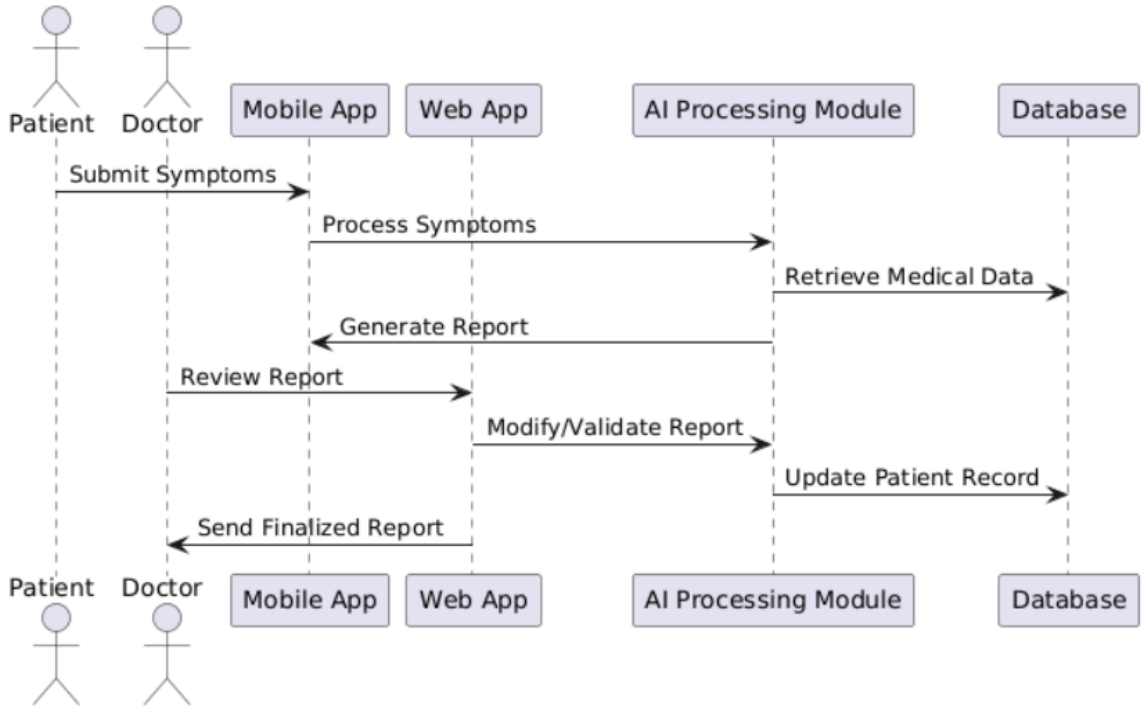


Figure 2

---

6. User Interface Design

In this section, we will describe the User Interface (UI) design of MedicalAI. The system is divided into two primary interfaces:

- **A Mobile Application for Patients** → Designed for symptom input, AI-generated disease reports, and treatment recommendations.
- **A Web Application for Doctors** → Optimized for managing patient data, reviewing AI reports, and making medical decisions.

6.1 Overview of User Interface

MedicalAI is structured to provide an **intuitive and efficient user experience** for both patients and doctors. The UI follows **a clean, minimalistic, and user-friendly design**, ensuring easy navigation and accessibility.

6.1.1 Mobile App for Patients

Mobile app which is written using **Jetpack Compose** metarial for Kotlin enables patients to enter symptoms, obtain medical insights, and receive reports created by AI. makes therapy recommendations in accordance with medical advice. shows past reports so that people can monitor their health.

6.1.2 Web Interface for Doctors

The web interface of MedicalAI which is written using **Vue.js** allows medical professionals to examine patient reports produced by AI, confirm diagnosis, and recommend therapies. enables illness and treatment searches through the integration of RxNorm and UMLS. uses a triage system driven by AI to prioritize urgent cases.

6.2 Screen Images

The screenshots we have provided below for MedicalAI are a mockup design. They are designed to guide us in the design phase of the applications.

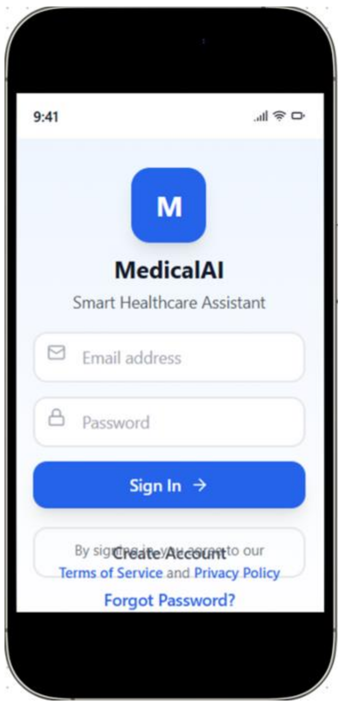

9:41
M
MedicalAI
Smart Healthcare Assistant
Email address
Password
Sign In
Create Account
By signing in, you agree to our
Terms of Service and Privacy Policy
Forgot Password?

Figure 3: MedicalAI Login Page

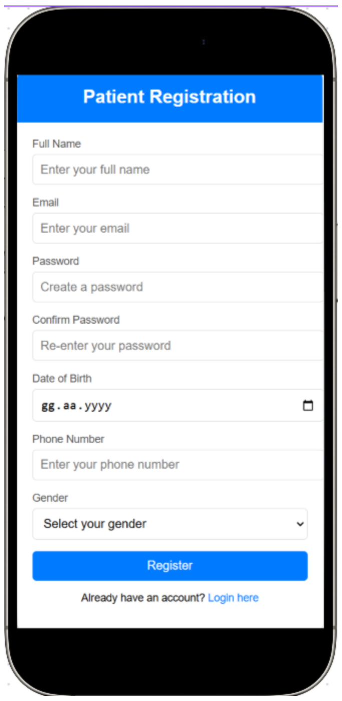


Figure 4: MedicalAI Sıgn In Page

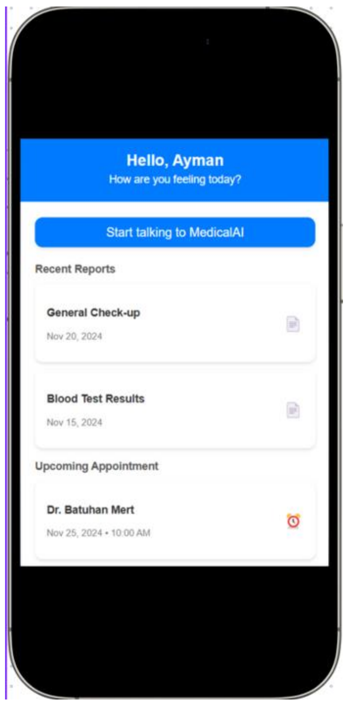
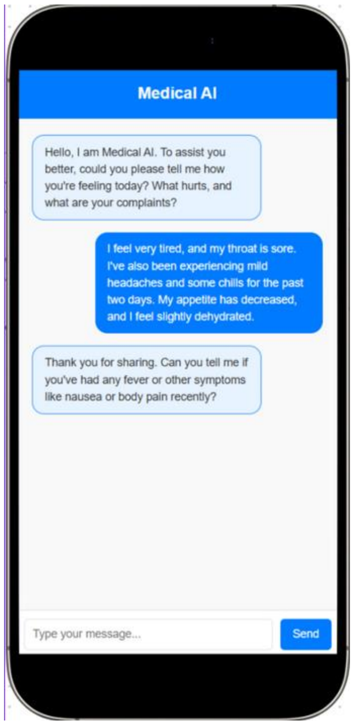


Figure 5: Home Page



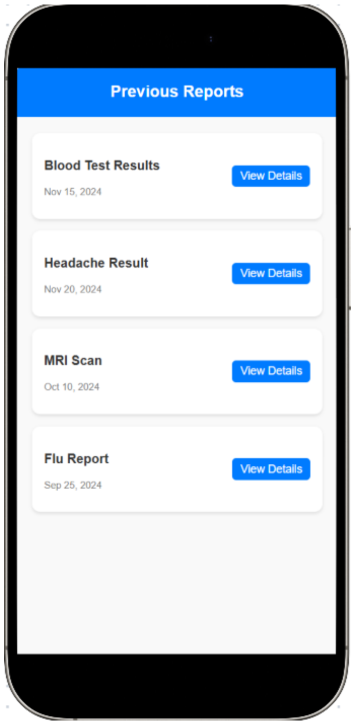
Medical AI

Hello, I am Medical AI. To assist you better, could you please tell me how you're feeling today? What hurts, and what are your complaints?

I feel very tired, and my throat is sore. I've also been experiencing mild headaches and some chills for the past two days. My appetite has decreased, and I feel slightly dehydrated.

Thank you for sharing. Can you tell me if you've had any fever or other symptoms like nausea or body pain recently?

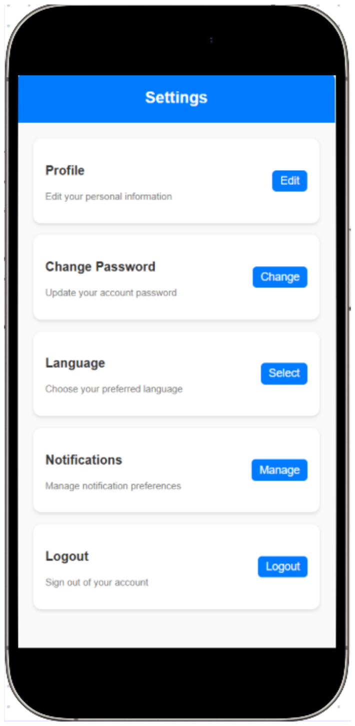
Figure 6: Chat interface with



Previous Reports

Blood Test Results	Nov 15, 2024	View Details
Headache Result	Nov 20, 2024	View Details
MRI Scan	Oct 10, 2024	View Details
Flu Report	Sep 25, 2024	View Details

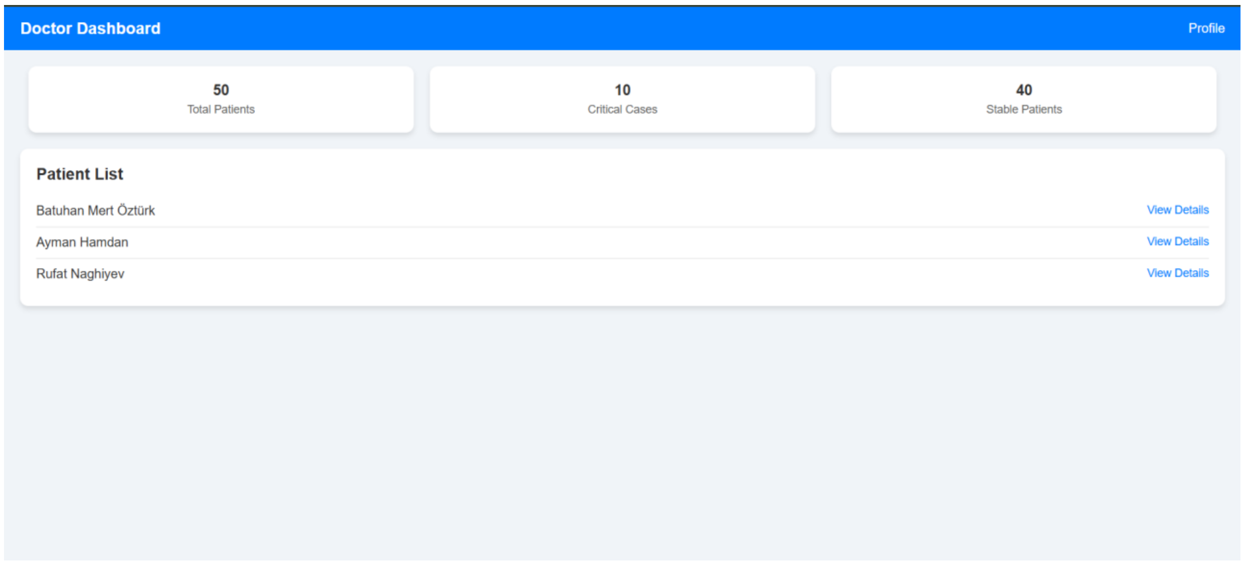
Figure 7: Reports



Settings

Profile	Edit your personal information	Edit
Change Password	Update your account password	Change
Language	Choose your preferred language	Select
Notifications	Manage notification preferences	Manage
Logout	Sign out of your account	Logout

Figure 8: Settings Page



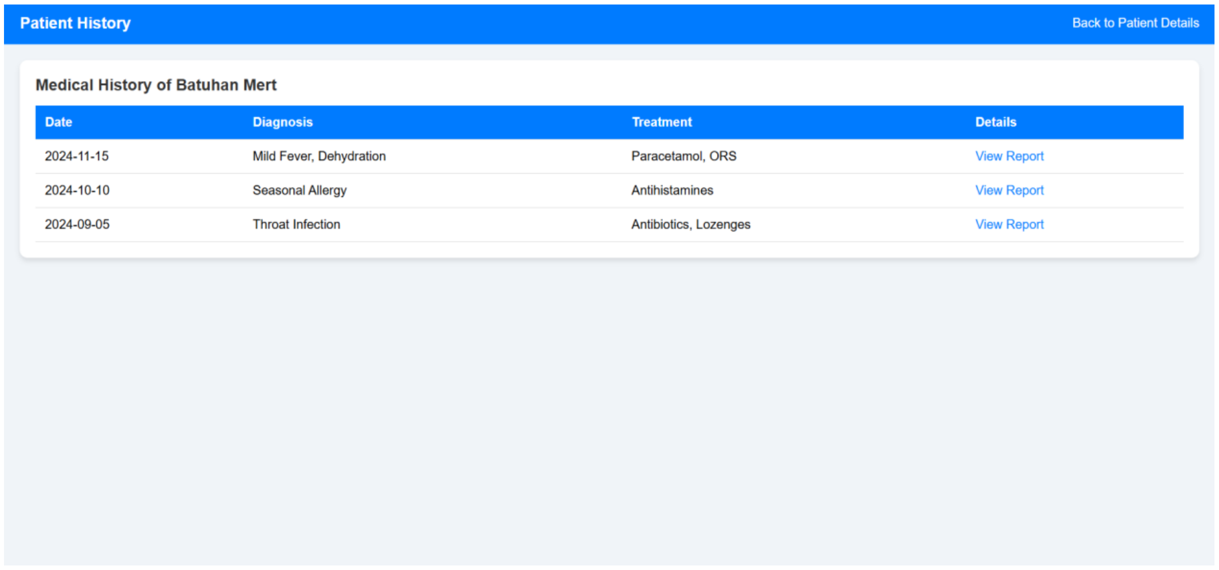
Doctor Dashboard Profile

50 Total Patients	10 Critical Cases	40 Stable Patients

Patient List

Batuhan Mert Öztürk	View Details
Ayman Hamdan	View Details
Rufat Naghiyev	View Details

Figure 9: Doctor Dashboard


Patient History

Back to Patient Details

Medical History of Batuhan Mert

Date	Diagnosis	Treatment	Details
2024-11-15	Mild Fever, Dehydration	Paracetamol, ORS	View Report
2024-10-10	Seasonal Allergy	Antihistamines	View Report
2024-09-05	Throat Infection	Antibiotics, Lozenges	View Report

Figure 10: History of Patient

7. Detailed Design

7.1 Key Components

1. LLM Processing

- Uses a **fine-tuned transformer model** to interpret patient symptoms and retrieve relevant medical knowledge.
- Implements **RAG (Retrieval-Augmented Generation)** to fetch supporting medical literature before generating responses.

2. Report Generation

- AI generates **structured medical reports** based on symptoms, previous history, and medical literature.
- Reports include **potential diagnoses, risk factors, and suggested next steps**.
- Doctors can edit reports before finalizing recommendations.

3. Authentication & Data Security

- User access and authorization levels should be defined but easily adjustable.
- Authentication methods should be adaptable and expandable when needed.
- System security should be continuously monitored and improved against new threats.

4. Doctor Interaction & Feedback

- Doctors can approve, modify, or reject AI-generated recommendations.
- Doctors can chat MedicalAI for decision-making

7.2 Constraints & Interactions

- **AI Limitations**: AI cannot diagnose diseases definitively; it serves as a decision-support tool.
- **Doctor Validation Process**: The system must enforce manual review by a medical professional before finalizing reports.

8. Libraries and Tools

8.1 Hardware

MedicalAI consists of a cloud-based backend, a mobile application for patients, and a web platform for doctors:

- **Cloud Backend:** Hosts AI processing, data storage, and authentication, ensuring secure and scalable operations. *Docker* is used for deployment
- **Mobile Application:** Runs on *Android Kotlin (Jetpack Compose),* allowing patients to input symptoms, and receive AI-generated recommendations.
- **Web Platform:** Designed for doctors to review AI-generated reports, validate diagnoses.

8.2 Software

MedicalAI utilizes a combination of backend, frontend, and AI technologies to ensure smooth operation and security:

- **Backend:** Built with Python (FastAPI) for handling API requests, user authentication, and AI processing.
- **AI Model:** Uses Large Language Models (LLMs) with Retrieval-Augmented Generation (RAG) for symptom analysis and diagnosis support.
- **Agent Framework:** Uses *CrewAI* agent framework to organize and communicate each agent in order to work together effectively.

- **Database:** PostgreSQL for structured data, Vector Database (e.g., Pinecone, FAISS) for AI-driven searches, and Firebase for real-time interactions.
- **Mobile Application:** Developed in **Android Kotlin (Jetpack Compose).**
- **Web Platform:** Built with *Vue.js* for a responsive interface, integrated with backend APIs.

9. Time Planning

We have created a time plan for the Medical AI project as shown in the Gantt chart below. We need to give a long time to collect, process and optimize the use of symptom analysis data for the agents that we will use in the project. Our goal is to comply with the time plan made for this comprehensive project.

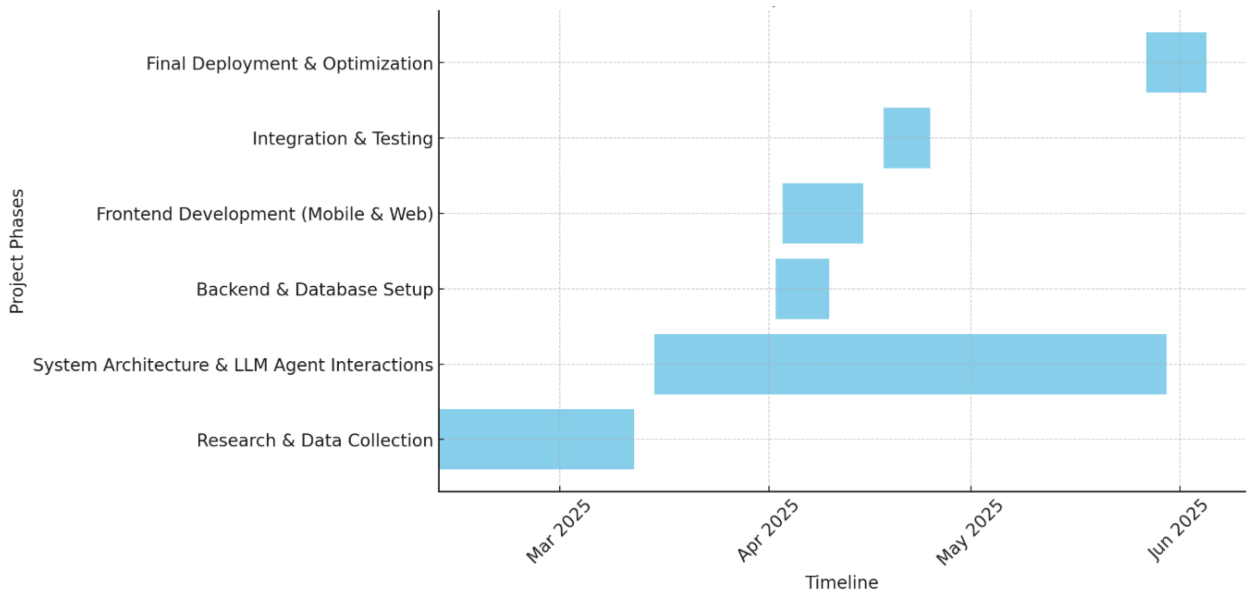


Figure 11: Gantt Chart for MedicalAI's Development Timeline

10. Conclusion

By combining vectorized medical information, retrieval-augmented generation (RAG), and AI-powered disease prediction, medical AI aims to transform medical decision-making. The system includes a web-based dashboard for physicians that offers AI-assisted diagnostic, patient management, and prescription capabilities, as well as a mobile application for patients that allows symptom entry, viewing of AI-generated reports, and therapy recommendations. MedicalAI ensures quick, precise, and comprehensible AI-driven healthcare insights by deploying a multi-agent AI system that effectively processes user inputs, retrieves pertinent medical knowledge, and produces comprehensive reports.

The integration of UMLS, SNOMED CT, and RxNorm allows for real-time knowledge retrieval, enhancing AI accuracy and reliability. The use of vector databases enables fast and precise semantic search, improving disease prediction and correlation analysis. These design decisions collectively aim to empower patients with AI-driven preliminary medical assessments, enhance doctor's efficiency by automating initial diagnostics, and advance AI's role in healthcare by demonstrating its potential in real-world medical applications.

In the future, MedicalAI can be further enhanced by extending its sources of medical knowledge to include academic papers, hospital databases, and real-time clinical trial data. Additionally, by improving models using patient history and contextual health data, tailored AI recommendations could be put into practice. Future versions might also incorporate image-based and speech-based diagnostics, enabling AI to interpret voice-based symptom descriptions and medical images for more thorough evaluations. MedicalAI has the potential to become a vital tool for patients and medical professionals alike, bridging the gap between AI-driven insights and human expertise, provided it keeps up with current developments in AI and healthcare.