🛠️ Development Documentation

Project Title: AI-Powered Healthcare Assistant (MedScan-AI)

# 🧩 1. Project Overview

The AI-Powered Healthcare Assistant is a web-based platform that enables users to upload medical reports (PDFs, images, etc.) and receive AI-analyzed diagnostic summaries, risk predictions, and medicine recommendations. It bridges the gap between technical medical reports and patient understanding using OCR, NLP, and AI technologies like Gemini or OpenAI.

# 🎯 2. Project Objectives

* Simplify medical report language using AI.
* Detect risks and abnormalities from lab results.
* Provide personalized health recommendations.
* Ensure data security and privacy compliance (HIPAA/GDPR).
* Enable access to healthcare knowledge for remote users.

## Feature Matrix

|  |  |  |
| --- | --- | --- |
| Feature | Description | Technology Stack |
| 📤 Medical Report Upload | Upload PDFs/images for analysis | React, HTML5 |
| 🧠 AI-Based Analysis | Gemini/OpenAI generates insights | Node.js, @google/generative-ai, OpenAI |
| 🔎 OCR Extraction | Extracts text from scanned reports | pdf-parse, Tesseract |
| 🩺 NLP Interpretation | Understands medical terms/symptoms | NLP (SpaCy, BioBERT), GPT |
| 💊 Medicine Suggestions | Based on diagnosis, suggests treatment | Gemini, Mayo Clinic API |
| 🧾 History Dashboard | Store & view past report results | MongoDB Atlas |
| 🗺️ Hospital Finder | Recommend nearby doctors & clinics | Google Maps API |
| 🧬 Telemedicine | Book virtual consultations | Socket.io, Zoom SDK |
| 🔐 Security & Compliance | Secure user data (HIPAA, GDPR) | AES-256, JWT, OAuth 2.0 |

# 🧱 4. System Architecture

Frontend (React.js + Tailwind CSS)  
• Upload report  
• View analysis  
• Chatbot UI (optional)  
• Dashboard  
  
Backend (Node.js + Express)  
• API endpoints for OCR, AI analysis, recommendations  
• MongoDB for storing reports/user data  
• Gemini/OpenAI prompt processor  
  
Database (MongoDB Atlas)  
• User profiles  
• Uploaded report metadata  
• Result logs  
  
External APIs  
• OpenAI / Gemini (LLM)  
• Google Vision API (OCR alt)  
• Mayo Clinic / WebMD API (med info)  
• Google Maps API (location services)

# 🧪 5. Testing Plan

Unit Testing - Mocha, Chai  
Integration Testing - Postman, Swagger  
Performance Testing - JMeter  
Accuracy Validation - PubMed, Expert Review

# 📦 6. Deployment Plan

Frontend: Vercel or Netlify  
Backend: Render or Railway  
Database: MongoDB Atlas  
Environment Variables: .env stored securely

# 🔑 7. Environment Variables (.env)

MONGO\_URI=your\_mongodb\_uri  
GEMINI\_API\_KEY=your\_gemini\_key  
JWT\_SECRET=your\_auth\_key

# 🔧 8. Tools & Libraries

OCR: Tesseract, pdf-parse  
NLP: SpaCy, BioBERT, NLTK  
AI: OpenAI / Gemini  
Frontend: React, Tailwind CSS  
Backend: Node.js, Express  
Auth: JWT, OAuth 2.0  
Storage: AWS S3 (optional), MongoDB  
Deployment: Vercel, Render, Docker (optional)

# 🚧 9. Challenges & Mitigation

• Variable report formats → Train models with diverse samples  
• Data privacy concerns → End-to-end encryption, HIPAA compliance  
• False positives in diagnosis → Manual validation, feedback loop  
• Multi-language input → Use multilingual NLP models  
• Abbreviations → Medical terminology dictionary integration

# 🔮 10. Future Enhancements

• Integration with smartwatches (Fitbit, Apple Health)  
• Blockchain for secure health record sharing  
• Drug interaction checker  
• Voice assistant integration (Alexa, Google Assistant)  
• Explainable AI (XAI) with reasoning logs

## 📁 11. Folder Structure (Actual)

medscan-ai/

├── backend/

│ ├── controllers/

│ │ ├── reportController.js // OCR + AI

│ │ ├── medicineController.js // Recommender

│ │ ├── hospitalController.js // Hospital/Blood bank

│ │ └── userController.js // Auth + dashboard

│ ├── routes/

│ │ ├── reportRoutes.js

│ │ ├── medicineRoutes.js

│ │ ├── hospitalRoutes.js

│ │ └── userRoutes.js

│ ├── services/

│ │ ├── geminiService.js

│ │ └── externalAPIService.js

│ ├── models/

│ │ ├── User.js

│ │ ├── Report.js

│ │ └── Medicine.js

│ ├── utils/

│ │ └── pdfExtractor.js

│ ├── app.js

│ ├── server.js

│ └── .env

│

├── frontend/

│ ├── src/

│ │ ├── components/

│ │ │ ├── UploadForm.js

│ │ │ ├── MedicineList.js

│ │ │ ├── HospitalMap.js

│ │ │ └── Dashboard.js

│ │ ├── pages/

│ │ │ ├── Home.js

│ │ │ ├── Analyze.js

│ │ │ ├── Login.js

│ │ │ └── Profile.js

│ │ ├── api/

│ │ │ ├── analyzeAPI.js

│ │ │ ├── medicineAPI.js

│ │ │ ├── hospitalAPI.js

│ │ │ └── authAPI.js

│ │ ├── App.js

│ │ └── index.js

│ └── tailwind.config.js

│

├── .gitignore

├── README.md

└── package.json

## 👥 Team-Based Feature Division (For 4–5 Members)

### 🔹 ****SHIVEK****

**Working Branch:** feature/ocr-analysis  
**Module:** Medical Report Analysis (OCR + AI)

**Responsibilities:**

* PDF & image upload
* OCR text extraction (pdf-parse, Tesseract, Google Vision API)
* Send text to Gemini/OpenAI API
* AI response parsing and display

**Tech Stack:**

* Node.js, Express, pdf-parse, Tesseract, @google/generative-ai
* React (upload form)
* API Route: POST /api/analyze/pdf

**Deliverables:**

* Clean OCR + AI analysis pipeline
* Frontend upload form + result view

### 🔹 ****RAJU****

**Working Branch:** feature/medicine-recommender  
**Module:** Medicine Recommendation System

**Responsibilities:**

* Interpret AI results for relevant medicines
* Use external API (WebMD, DrugBank) or internal DB
* Display medicine name, dose, side effects (optional)
* Store medicine suggestions in user history

**Tech Stack:**

* Node.js (API handler)
* MongoDB (medicine DB or logs)
* React (medicine display page)

**Deliverables:**

* Dynamic medicine suggestion UI
* Secure backend logic

### 🔹 ****HUSSIAN****

**Working Branch:** feature/hospital-locator  
**Module:** Nearby Hospital & Blood Bank Locator

**Responsibilities:**

* Location-based hospital and blood bank suggestions
* Integrate Google Maps API or Mapbox
* Show contact info, rating, availability (if API supports)

**Tech Stack:**

* React + Google Maps API
* HTML5 Geolocation API
* Optional DB of local hospitals/blood banks

**Deliverables:**

* Hospital locator with search
* Blood bank availability UI

### 🔹 ****SONAM****

**Working Branch:** feature/user-auth-dashboard  
**Module:** User Dashboard + History + Auth

**Responsibilities:**

* JWT-based login/signup system
* User profile dashboard
* Display previous reports, analysis, recommendations

**Tech Stack:**

* Node.js + JWT Auth
* MongoDB (User & report history)
* React + Tailwind UI

**Deliverables:**

* Auth flow: login, logout, protected routes
* Dashboard page with charts/history

### 🔹 ****Team Member 5 (Optional)****

**Working Branch:** feature/admin-chatbot  
**Module:** Admin Panel + Email/Chatbot Integration

**Responsibilities:**

* Admin view of all users/reports
* Email summary system (NodeMailer)
* AI chatbot interface (optional feature)

**Tech Stack:**

* Node.js (Admin routes + mailer)
* React Admin Panel
* AI: Gemini conversational endpoint

**Deliverables:**

* Working admin panel
* Emailing system or basic chatbot

## 📋 Suggested GitHub Branch Naming Convention

bash

CopyEdit

main/

├── feature/ocr-analysis

├── feature/medicine-recommender

├── feature/hospital-locator

├── feature/user-auth-dashboard

├── feature/admin-chatbot