Full Stack Development with MERN

Project Documentation format

1. Introduction

Project Title: HealthAI - Intelligent Healthcare Assistant

Team Members:

-Pola Sruthi
-Pogala Sandya
-Rachamreddy Suhasini
- Puduru Kalyani

2. Project Overview

Purpose:

Our goal is to build an AI-powered assistant using Hugging Face models and a MERN stack application to help users check symptoms, predict diseases, and get health advice.

Features:

- Symptom checker
- Disease prediction
- AI medical chatbot
- Health report summarizer
- Interactive dashboards

3. Architecture

Frontend:

React.js for building the user interface with components for chat, forms, and dashboards.

Backend:

Node.js with Express.js to serve REST APIs, manage authentication, and integrate with ML services.

Database:

MongoDB for storing user profiles, symptom logs, and predictions.

ML Integration:

Used Hugging Face transformer models in Python on Google Colab for disease prediction and chatbot functionalities.

4. Setup Instructions

Prerequisites:

- Node.js, npm
- MongoDB
- Python 3.x
- Google Colab account
- Hugging Face API token

Installation:

- Clone the repo
- Install dependencies in client and server folders
- Set up environment variables (e.g., MONGO_URI, JWT_SECRET)

5. Folder Structure

Client:

- src/
  - components/
  - pages/
  - services/

Server:

- routes/
- controllers/
- models/
- middleware/
- ml/ (optional folder for integrating Python scripts)

6. Running the Application

Frontend:

- Run `npm start` inside the client folder.

Backend:

- Run `npm start` inside the server folder.

Colab:

- Open the provided Colab notebook and run cells to start ML APIs.

7. API Documentation

- `POST /api/symptoms` - Send symptoms and receive predictions.
- `POST /api/chat` - Send questions to the chatbot.
- `GET /api/reports` - Retrieve user health reports.

8. Authentication

Implemented JWT-based authentication. Users receive a token on login, which is required to access protected routes.

9. User Interface
Includes:
- Chatbot page
- Symptom submission form
- Dashboard with visualizations
- Login and signup pages.

10. Using Hugging Face & Colab
We developed ML models using Hugging Face transformers on Google Colab. Key steps:
- Loaded models with AutoModel and AutoTokenizer.
- Ran inference on symptoms/questions.
- Hosted endpoints using Colab notebooks and shared with MERN backend via HTTP requests.
Example Python code:

```
from transformers import AutoModelForCausalLM, AutoTokenizer
model = AutoModelForCausalLM.from_pretrained("facebook/blenderbot-400M-distill")
tokenizer = AutoTokenizer.from_pretrained("facebook/blenderbot-400M-distill")
def medical_qa(query):
    inputs = tokenizer(query, return_tensors="pt")
    reply_ids = model.generate(**inputs)
    return tokenizer.decode(reply_ids[0], skip_special_tokens=True)
```

11. Testing
Frontend tests with React Testing Library; backend tests with Mocha/Chai; ML model outputs manually verified on sample data.

12. Screenshots or Demo
demo video link:
https://drive.google.com/file/d/1XewsnPycXuEtqvSkPyzMNaezCBwf5P1a/view?usp=sharing

13. Known Issues
- ML responses may have latency when running on free Colab tiers.
- Some rare symptoms might not produce accurate predictions.

14. Future Enhancements
- Integrate real-time video calls with doctors
- Add multi-language support to chatbot
- Move ML models to dedicated cloud hosting for lower latency