AI-Powered Mental Health Tracker – Documentation

Abstract

Mental health issues are becoming increasingly prevalent, yet access to timely and personalized mental health support remains a challenge. The **AI-Powered Mental Health Tracker** is a mobile application designed to analyze user emotions using **Natural Language Processing (NLP)** and **Voice Emotion Analysis**. By utilizing AI-driven sentiment analysis and voice tone detection, the app provides **personalized mental health recommendations**, helping users manage their emotional well-being effectively. This solution not only enables self-awareness but also encourages proactive mental health care by offering relevant mindfulness exercises, therapy suggestions, and mood tracking features.

1. Project Overview

The **AI-Powered Mental Health Tracker** is a mobile application designed to analyze user emotions using text and voice inputs. The app utilizes **Natural Language Processing (NLP)** and **Voice Emotion Analysis** to provide personalized mental health recommendations, exercises, and therapy suggestions.

Key Features:

- **Text-Based Sentiment Analysis** (Using BERT/GPT models)
- **Voice Emotion Detection** (Using Librosa & SpeechRecognition)
- **Personalized Recommendations**
- **Mood Progress Tracking**
- **React Native Mobile Application**

---

2. Tech Stack

Backend (Flask + AI Models)

- Python (Flask/FastAPI)
- Hugging Face Transformers (BERT for sentiment analysis)
- Librosa & SpeechRecognition (Voice emotion analysis)
- PostgreSQL/MongoDB (For storing user data)

Frontend (React Native)

- React Native (for cross-platform mobile UI)
- Axios (API communication)
- React Navigation (App navigation)

---

3. System Architecture

User Input (Text/Voice)

↓

Frontend (React Native)

↓

Backend API (Flask)

↓

NLP Model (BERT) for Sentiment Analysis

Voice Emotion Detection (Librosa + SpeechRecognition)

↓

Personalized AI-Based Recommendations

↓

Response Sent Back to Frontend

↓

User Receives Mood Analysis & Suggestions

---

4. Implementation

Backend: Flask + AI Models

◆ **app.py (Flask API)**

```
from flask import Flask, request, jsonify
from sentiment_model import analyze_sentiment
from voice_analysis import analyze_voice_emotion
from recommendations import get_recommendations

app = Flask(__name__)

@app.route("/analyze-text", methods=["POST"])
def analyze_text():
    data = request.json
    text = data.get("text", "")
    sentiment = analyze_sentiment(text)
    recommendations = get_recommendations(sentiment)
    return jsonify({"sentiment": sentiment, "recommendations": recommendations})

@app.route("/analyze-audio", methods=["POST"])
def analyze_audio():
    file = request.files["audio"]
    file.save("temp_audio.wav")
    emotion = analyze_voice_emotion("temp_audio.wav")
    recommendations = get_recommendations(emotion)
    return jsonify({"emotion": emotion, "recommendations": recommendations})

if __name__ == "__main__":
    app.run(debug=True)
```

- **sentiment_model.py (BERT Sentiment Analysis)**

```
from transformers import pipeline

sentiment_pipeline = pipeline("sentiment-analysis")

def analyze_sentiment(text):
    result = sentiment_pipeline(text)
    return result[0]["label"]
```

- **voice_analysis.py (Voice Emotion Detection)**

```
import librosa
import numpy as np
import speech_recognition as sr
```

```
def extract_audio_features(file_path):
    y, sr = librosa.load(file_path)
    mfcc = np.mean(librosa.feature.mfcc(y=y, sr=sr, n_mfcc=13).T, axis=0)
    return mfcc


def analyze_voice_emotion(file_path):
    recognizer = sr.Recognizer()
    with sr.AudioFile(file_path) as source:
        audio = recognizer.record(source)
    try:
        text = recognizer.recognize_google(audio)
        return "Sad" if "depressed" in text.lower() else "Happy"
    except:
        return "Neutral"
```

- **recommendations.py (Personalized AI Suggestions)**

```
def get_recommendations(mood):
    suggestions = {
        "POSITIVE": ["Try meditation", "Go for a walk", "Listen to relaxing music"],
        "NEGATIVE": ["Consider talking to a therapist", "Practice deep breathing"],
        "NEUTRAL": ["Journaling can help", "Try light exercise"],
        "Happy": ["Keep up the good work!", "Do something creative"],
        "Sad": ["Talk to a friend", "Listen to uplifting music"],
        "Neutral": ["Maintain a gratitude journal"]
    }
    return suggestions.get(mood, ["Stay mindful"])
```

Frontend: React Native

- **App.js (Main Component)**

```
import React from 'react';
import { NavigationContainer } from '@react-navigation/native';
import { createStackNavigator } from '@react-navigation/stack';
import HomeScreen from './screens/HomeScreen';
import ResultScreen from './screens/ResultScreen';

const Stack = createStackNavigator();

export default function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator>
        <Stack.Screen name="Home" component={HomeScreen} />
        <Stack.Screen name="Results" component={ResultScreen} />
      </Stack.Navigator>
    </NavigationContainer>
  );
}
```

◆ **HomeScreen.js (User Input - Text & Audio)**

```
import React, { useState } from "react";
import { View, TextInput, Button, Text } from "react-native";
import axios from "axios";

export default function HomeScreen({ navigation }) {
  const [text, setText] = useState("");
  const [recording, setRecording] = useState(false);

  const analyzeText = async () => {
    const response = await axios.post("http://localhost:5000/analyze-text", { text });
    navigation.navigate("Results", response.data);
```

```
 };

 const analyzeAudio = async () => {
  // Audio recording logic (to be integrated with react-native-voice)
 };

 return (
  <View style={{ padding: 20 }}>
   <Text>Enter your thoughts:</Text>
   <TextInput
    placeholder="How are you feeling?"
    value={text}
    onChangeText={setText}
    style={{ borderBottomWidth: 1, marginBottom: 10 }}
   />
   <Button title="Analyze Text" onPress={analyzeText} />
   <Button title={recording ? "Stop Recording" : "Record Voice"} onPress={analyzeAudio} />
  </View>
 );
}
```

---

5. Future Enhancements

- **Real-Time Chatbot Support** using OpenAI's GPT.
- **Data Visualization** (Mood Trends Over Time)
- **Therapist API Integration** (Recommend Nearby Mental Health Experts)
- **Gamification & Rewards** for tracking emotions consistently.
- **Integration with Wearable Devices** for stress level monitoring.
- **Privacy-Focused Edge AI** for local processing.

---

6. Contributors

- **Jyothi Laxmi**- Lead Developer

For queries, reach out to jyothilaxmipoosaala@gmail.com

---

7. License

This project is licensed under the MIT License - see the LICENSE file for details.