**Report on Medical Symptom Checker (with RAG)**

**Introduction**

The Medical Symptom Checker is a web-based application that utilizes Retrieval-Augmented Generation (RAG) to analyze user-provided symptoms and retrieve relevant medical information. This system supports three diseases: Diabetes, Thyroid Disorders, and Typhoid Fever. It enables users to input their symptoms and select a disease category, and the system responds with relevant medical insights.

**Objective**

The primary objective of this project is to create an intelligent medical query system that:

- Uses retrieval-based methods to fetch relevant medical data.

- Utilizes LLM (Large Language Models) for generating meaningful responses.

- Provides users with a preliminary assessment of their symptoms while advising them to seek professional medical consultation.

**Technologies Used**

The system integrates several AI and web development technologies, including:

- Python for backend development.

- Flask as the web framework to create an API-based interface.

- FAISS (Facebook AI Similarity Search) for efficient nearest-neighbor search.

- Sentence Transformers for embedding textual medical records.

- Hugging Face Transformers to use the "mistralai/Mistral-7B-Instruct-v0.1" model for text generation.

- Pandas for handling structured medical datasets.

**Dataset Description**

The system uses three datasets stored as CSV files:

- diabetes.csv – Contains information about symptoms and indicators of Diabetes.

- Thyroid\_Diff.csv – Includes medical data related to Thyroid disorders.

- Typhoid.csv – Holds symptom-related information about Typhoid Fever.

**Working Mechanism**

The workflow of the system is as follows:

1. Data Loading

- Three datasets are loaded into Pandas DataFrames.

- Each dataset is preprocessed to create text-based representations of medical records.

2. FAISS Indexing

- Each dataset is converted into text representations, embedded using Sentence Transformers, and stored in FAISS indices.

- FAISS allows for efficient similarity search, ensuring relevant records are retrieved quickly.

3. User Query Processing

- The user enters symptoms and selects a disease category (Diabetes, Thyroid, or Typhoid).

- The system embeds the query and searches for the most relevant records using FAISS.

4. Retrieval-Augmented Generation (RAG)

- Retrieved medical records are passed into the Mistral-7B-Instruct language model.

- The model generates an AI-based response using a structured prompt.

5. API-Based Interaction

- The Flask API allows users to input symptoms and receive AI-generated responses.

- The output consists of the AI response and a disclaimer advising users to consult a medical professional.

**Flask API Endpoints**

- `/ask` (POST Request) – Accepts user input and disease selection, returns AI-generated responses.

- Example Request:

```json

{

"query": "I feel extremely thirsty and urinate frequently.",

"disease": "diabetes"

}

```

- Example Response:

```json

{

"answer": "Based on the provided symptoms, you may be at risk for diabetes. It is advisable to consult a doctor for further evaluation.",

"disclaimer": "This is an AI-generated response. Please consult a medical professional for confirmation."

}

```

**Conclusion**

This Medical Symptom Checker provides an AI-powered approach to preliminary medical assessments. By combining retrieval-based search (FAISS) with generative AI (Mistral-7B-Instruct), it delivers meaningful insights based on structured medical records. However, it is not a substitute for professional medical advice, and users are strongly encouraged to consult a healthcare professional for accurate diagnosis and treatment.