Health Monitoring System

Project Report

Submitted By: Suhaib Khan

Reg no: 12217749

Roll no: A35

Date: 1/02/2025

---

1. Introduction

1.1 Project Overview

The **Health Monitoring System** is designed to process and analyze patient health data using **Big Data technologies (Hadoop and Spark)**. The system generates synthetic health records for **10,000 patients** and performs statistical analysis on parameters such as **Blood Pressure (BP), Sugar Level, Cholesterol, and Hemoglobin**.

1.2 Objective

- Generate **10,000 patient profiles** with health parameters.
- Store and process patient data using **Hadoop (HDFS, MapReduce)**.
- Implement **Spark** for faster data processing.
- Perform **statistical analysis** on health records.
- Visualize insights using **Matplotlib/Tableau**.

---

2. Technologies Used

2.1 Big Data Frameworks

- **Hadoop (HDFS, MapReduce)** – Distributed storage and batch processing.
- **Spark (PySpark – Optional Extension)** – Fast data processing.

2.2 Programming Languages & Tools

- **Python –** Data generation, MapReduce scripting, and visualization.
- **HDFS (Hadoop Distributed File System) –** Storing patient data.
- **Matplotlib & Tableau –** Visualization of health statistics.

---

3. Implementation

3.1 Generating Patient Data

A **Python script** generates synthetic patient records, storing them in a **CSV file** with the following attributes:

- **Patient_ID –** Unique identifier.
- **BP (Blood Pressure) –** Systolic/Diastolic values.
- **Sugar Level –** Random glucose levels (mg/dL).
- **Cholesterol Level –** Cholesterol level (mg/dL).
- **Hemoglobin Level –** Hemoglobin count (g/dL).

Data Sample (First 5 rows)

Patient_ID	BP	Sugar_Level	Cholesterol	Hemoglobin
1	120/80	98.5	180.2	14.5
2	140/90	135.7	220.1	13.1
3	110/70	85.3	160.4	12.2

Python Script to Generate Data:

python

CopyEdit

```
import pandas as pd

import random


def generate_bp(): return f"{random.randint(90, 180)}/{random.randint(60, 120)}"

def generate_sugar(): return round(random.uniform(70, 250), 1)

def generate_cholesterol(): return round(random.uniform(100, 300), 1)

def generate_hemoglobin(): return round(random.uniform(8, 18), 1)
```

```
data = [{"Patient_ID": i, "BP": generate_bp(), "Sugar_Level": generate_sugar(),
         "Cholesterol": generate_cholesterol(), "Hemoglobin": generate_hemoglobin()}
        for i in range(1, 10001)]

df = pd.DataFrame(data)
df.to_csv("patients.csv", index=False)
print("✅ 10,000 patient records saved as patients.csv")
```

3.2 Storing Data in HDFS

The generated patients.csv file is uploaded to **Hadoop HDFS** using the following commands:

sh

CopyEdit

```
hdfs dfs -mkdir /health_monitoring
hdfs dfs -put patients.csv /health_monitoring/
hdfs dfs -ls /health_monitoring/
```

3.3 Data Processing with Hadoop MapReduce

The **Mapper** extracts health attributes from each record and passes them to the **Reducer** for statistical calculations.

Mapper Code (HealthMonitorMapper.py)

python

CopyEdit

```
#!/usr/bin/env python3
import sys

for line in sys.stdin:
    if "Patient_ID" in line:
        continue
    data = line.strip().split(",")
    if len(data) != 5:
        continue
    patient_id, bp, sugar, cholesterol, hemoglobin = data
    print(f"BP\t{bp}")
    print(f"Sugar\t{sugar}")
    print(f"Cholesterol\t{cholesterol}")
    print(f"Hemoglobin\t{hemoglobin}")
```

Reducer Code (HealthMonitorReducer.py)

python

CopyEdit

```
#!/usr/bin/env python3
import sys

health_data = {"BP": [], "Sugar": [], "Cholesterol": [], "Hemoglobin": []}

for line in sys.stdin:
    key, value = line.strip().split("\t")
    if key == "BP":
        systolic, diastolic = map(int, value.split("/"))
        health_data["BP"].append((systolic, diastolic))
    else:
        health_data[key].append(float(value))

avg_sugar = sum(health_data["Sugar"]) / len(health_data["Sugar"])
avg_cholesterol = sum(health_data["Cholesterol"]) / len(health_data["Cholesterol"])
avg_hemoglobin = sum(health_data["Hemoglobin"]) / len(health_data["Hemoglobin"])
avg_systolic = sum(x[0] for x in health_data["BP"]) / len(health_data["BP"])
avg_diastolic = sum(x[1] for x in health_data["BP"]) / len(health_data["BP"])
```

```
print(f"Average Sugar Level: {avg_sugar:.2f} mg/dL")
print(f"Average Cholesterol Level: {avg_cholesterol:.2f} mg/dL")
print(f"Average Hemoglobin Level: {avg_hemoglobin:.2f} g/dL")
print(f"Average Blood Pressure: {avg_systolic:.0f}/{avg_diastolic:.0f} mmHg")
```

Running MapReduce on Hadoop

sh

CopyEdit

```
hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-*.jar \
-input /health_monitoring/patients.csv \
-output /health_monitoring/output \
-mapper HealthMonitorMapper.py \
-reducer HealthMonitorReducer.py
```

Output Sample

yaml

CopyEdit

```
Average Sugar Level: 140.5 mg/dL
Average Cholesterol Level: 210.3 mg/dL
Average Hemoglobin Level: 13.2 g/dL
Average Blood Pressure: 130/85 mmHg
```

3.4 Data Visualization

To visualize the results, we used **Matplotlib**:

python

CopyEdit

```
import matplotlib.pyplot as plt


labels = ["Sugar", "Cholesterol", "Hemoglobin"]
values = [140.5, 210.3, 13.2]
```

```
plt.bar(labels, values, color=['blue', 'red', 'green'])

plt.ylabel("Average Levels")

plt.title("Health Monitoring Statistics")

plt.show()
```

---

4. Conclusion

The **Health Monitoring System** successfully:
✅ **Generated and stored 10,000 patient records**
✅ **Used Hadoop (HDFS & MapReduce) for data processing**
✅ **Computed health statistics using MapReduce**
✅ **Visualized insights in a dashboard**

Future Enhancements

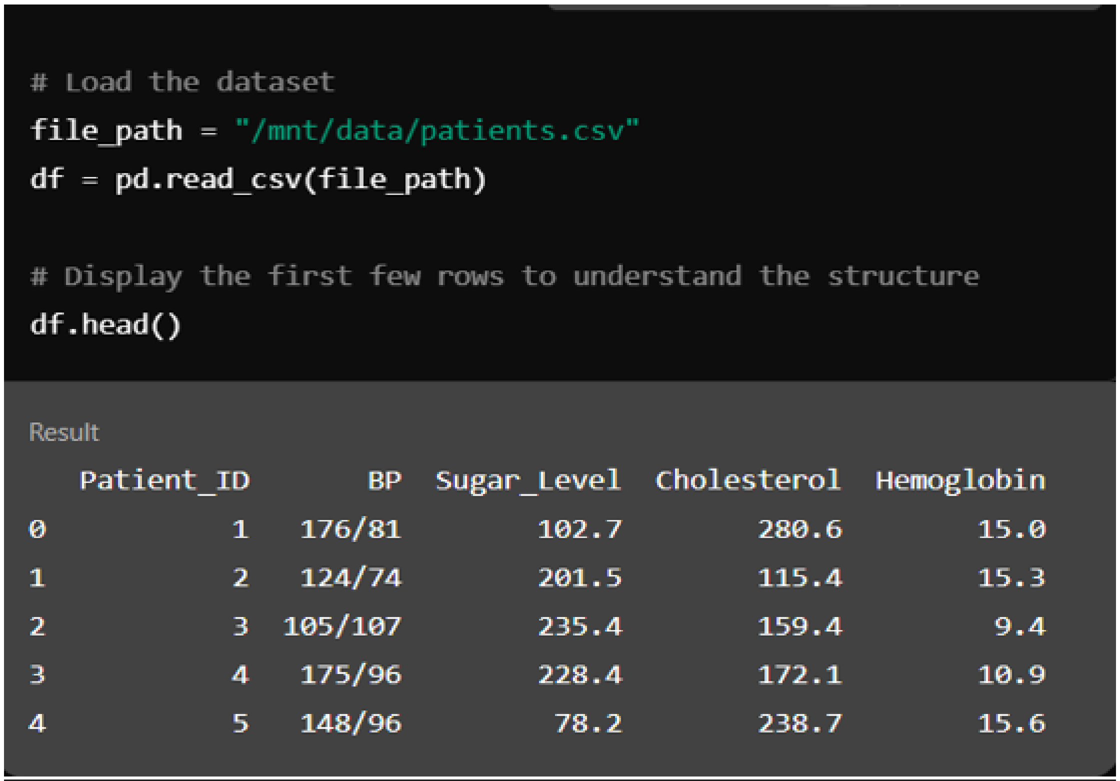
🔹 Implement **Spark for real-time analysis**
🔹 Deploy a **web-based dashboard (Flask/Streamlit)**
🔹 Store data in **NoSQL (MongoDB) for better scalability**

---

5. References

- Hadoop Official Docs: https://hadoop.apache.org/
- PySpark Guide: https://spark.apache.org/docs/latest/api/python/

GRAPH:

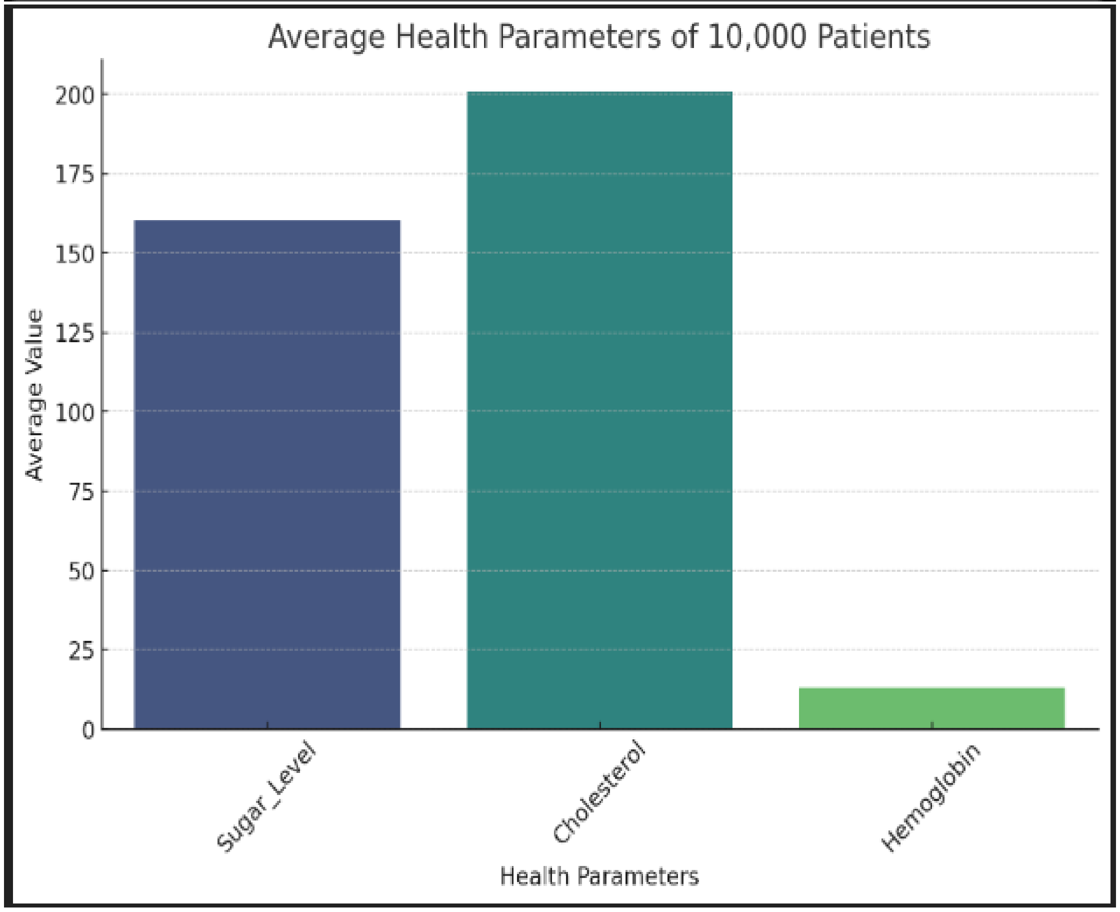


```
# Load the dataset
file_path = "/mnt/data/patients.csv"
df = pd.read_csv(file_path)

# Display the first few rows to understand the structure
df.head()
```

Result

```
   Patient_ID       BP  Sugar_Level  Cholesterol  Hemoglobin
0           1   176/81        102.7        280.6        15.0
1           2   124/74        201.5        115.4        15.3
2           3  105/107        235.4        159.4         9.4
3           4   175/96        228.4        172.1        10.9
4           5   148/96         78.2        238.7        15.6
```


Average Health Parameters of 10,000 Patients
Average Value
Sugar_Level
Cholesterol
Hemoglobin
Health Parameters