Winter PEP Final Project Report

Result Management System(RMS)

Submitted By:

Name: Shivansh Mishra

Section:9W084

Reg.No.:12216777

Roll No.:R9W084A38

To:

Mentor Name: Mr. Lokesh

1. Introduction

The Result Management System (RMS) is designed to efficiently process student performance data, compute relevant metrics, and generate insights. The project leverages **Python, Apache Spark (PySpark), Pandas, and SQL/NoSQL databases** to handle large datasets and extract meaningful statistics.

1.1 Objective

- To generate synthetic student performance data.
- To process large datasets efficiently using PySpark.
- To apply data transformation, aggregation, and visualization techniques.
- To prepare the system for an interactive dashboard (implementation not covered in this document).

---

2. Tools & Technologies Used

Component	Technology Used	Purpose
Programming Language	Python	Core implementation
Big Data Processing	Apache Spark (PySpark)	Handling large datasets
Data Manipulation	Pandas	Converting data into a structured format
Database (Optional)	PostgreSQL / MySQL / MongoDB	Storing student records
IDE	VS Code	Development environment
Version Control	Git & GitHub	Code management

---

3. Data Generation & Processing

3.1 Input Data Generation

- **Generating synthetic student profiles** with randomly assigned names.
- **Creating random marks (0-100) for subjects** such as Electronics, Programming, Mathematics, etc.
- **Storing data in a Pandas DataFrame** before converting it to PySpark.

Code Snippet for Student Data Generation:

data_generating.py > ...

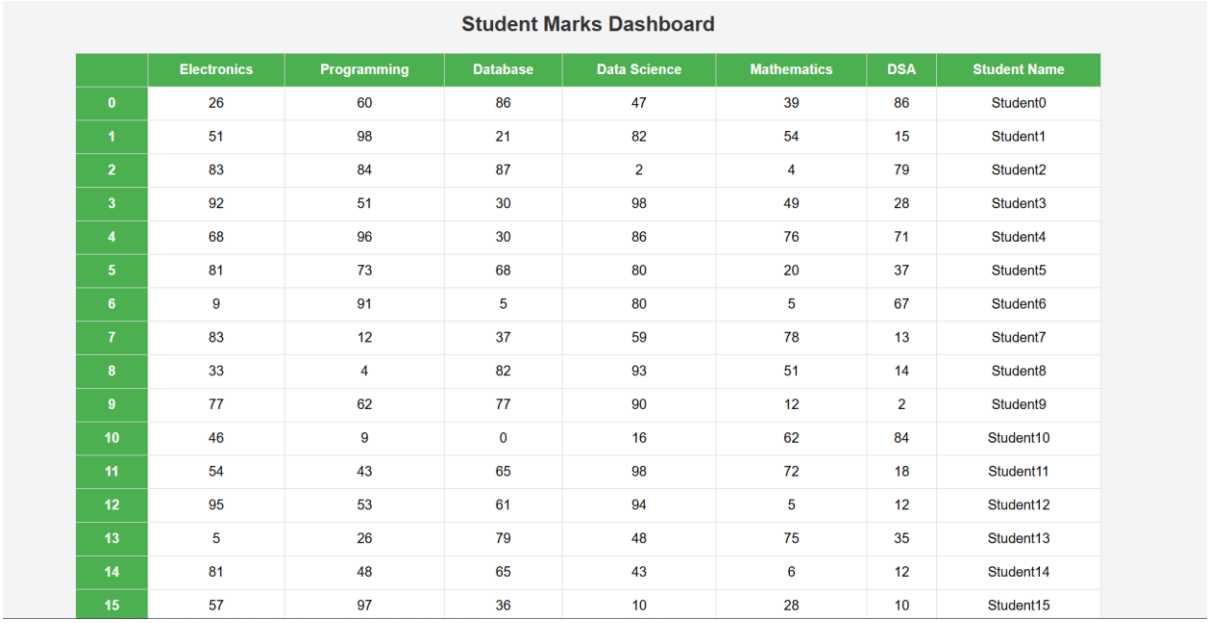
```python
from pyspark.sql import SparkSession
import pandas as pd
import random
spark = SparkSession.builder.appName("ResultManagementSystem").getOrCreate()
subjects = ["Electronics", "Programming", "Database", "Data Science", "Mathematics", "DSA"]
def generate_student_profiles(num_students=10000):
    return [f'Student{i}' for i in range(num_students)]
def generate_marks(num_students=10000, subjects=subjects):
    marks = {subject: [random.randint(0, 100) for _ in range(num_students)] for subject in subjects}
    return pd.DataFrame(marks)
students = generate_student_profiles()
marks_df = generate_marks(len(students))
marks_df['Student Name'] = students
spark_df = spark.createDataFrame(marks_df)
marks_df.to_csv("student_results.csv", index=False)

```

Code Snippet for Student Data Processing:

data_processing.py > ...

```python
from pyspark.sql import functions as F
import pandas as pd

def process_student_data():
    spark = SparkSession.builder.appName("RMS_Processing").getOrCreate()

    # Load data from CSV
    df_pandas = pd.read_csv("../data/student_data.csv")
    spark_df = spark.createDataFrame(df_pandas)

    # Compute Aggregate Metrics
    spark_df = spark_df.withColumn("Total", sum(spark_df[subject] for subject in ["Math", "Science", "Programming"]))
    spark_df = spark_df.withColumn("Percentage", (spark_df["Total"] / (100 * 3)) * 100)

    spark_df.show(10)
    return spark_df

if __name__ == "__main__":
    process_student_data()

```

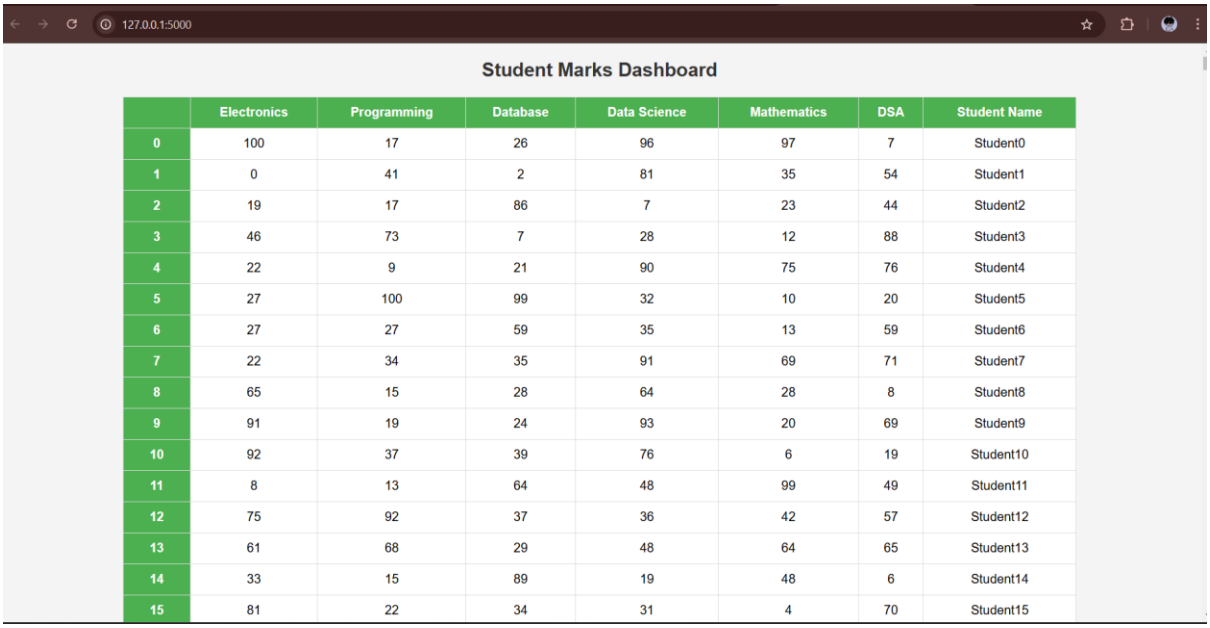
Dashboard Image:



Student Marks Dashboard

	Electronics	Programming	Database	Data Science	Mathematics	DSA	Student Name
0	26	60	86	47	39	86	Student0
1	51	98	21	82	54	15	Student1
2	83	84	87	2	4	79	Student2
3	92	51	30	98	49	28	Student3
4	68	96	30	86	76	71	Student4
5	81	73	68	80	20	37	Student5
6	9	91	5	80	5	67	Student6
7	83	12	37	59	78	13	Student7
8	33	4	82	93	51	14	Student8
9	77	62	77	90	12	2	Student9
10	46	9	0	16	62	84	Student10
11	54	43	65	98	72	18	Student11
12	95	53	61	94	5	12	Student12
13	5	26	79	48	75	35	Student13
14	81	48	65	43	6	12	Student14
15	57	97	36	10	28	10	Student15

After process the data:



Student Marks Dashboard

	Electronics	Programming	Database	Data Science	Mathematics	DSA	Student Name
0	100	17	26	96	97	7	Student0
1	0	41	2	81	35	54	Student1
2	19	17	86	7	23	44	Student2
3	46	73	7	28	12	88	Student3
4	22	9	21	90	75	76	Student4
5	27	100	99	32	10	20	Student5
6	27	27	59	35	13	59	Student6
7	22	34	35	91	69	71	Student7
8	65	15	28	64	28	8	Student8
9	91	19	24	93	20	69	Student9
10	92	37	39	76	6	19	Student10
11	8	13	64	48	99	49	Student11
12	75	92	37	36	42	57	Student12
13	61	68	29	48	64	65	Student13
14	33	15	89	19	48	6	Student14
15	81	22	34	31	4	70	Student15

3.2 Data Transformation & Aggregation (ETL Process)

1. **Extract**: Load data from Pandas into a PySpark DataFrame.
2. **Transform**:
    - Compute **average, minimum, and maximum marks** using agg().
    - Calculate **total marks and percentage** dynamically.

3. **Load**:
    - Store processed data in a **SQL database** (PostgreSQL/MySQL).
    - Alternatively, export it as **CSV/JSON** for further processing.

---

4. Challenges Faced & Solutions

Challenge	Solution Implemented
Handling large data sets	Used Apache Spark for distributed processing
Storing results efficiently	Implemented PostgreSQL/MongoDB storage
Processing speed optimization	Optimized Spark transformations

---

5. Future Scope & Enhancements

- Adding **student login** to fetch personalized reports.
- Implementing **predictive analytics** to identify weak students.
- Cloud deployment for scalability.
- Enhancing database integration for real-time updates.

---

6. Conclusion

This project successfully demonstrates the use of **Apache Spark and Python** for efficiently processing large-scale student performance data. By leveraging **SQL/NoSQL storage and aggregation techniques**, the system provides valuable insights into academic performance. The upcoming dashboard implementation will visualize this data dynamically.
