Safety Critical Systems Project Report
Predictive Maintenance in Vehicle Systems

Pushpita Sarkar(Matriculation No: 1384152), Nidhi Nayak (Matriculation No: 1404524), Deepak Kumar (Matriculation No: 1400489), and Ashlesh Mithur (Matriculation No: 1386367)

Frankfurt University of Applied Sciences

1 Introduction

Predictive maintenance is a technique that uses data analysis tools and techniques to detect anomalies in your operation and possible defects in equipment and processes so you can fix them before they result in failure. Ideally, predictive maintenance allows the maintenance frequency to be as low as possible to prevent unplanned reactive maintenance, without incurring costs associated with doing too much preventive maintenance. When predictive maintenance is working effectively as a maintenance strategy, maintenance is only performed on machines when it is required. That is, just before failure is likely to occur. This brings several cost savings:

Minimizing the time, the equipment is being maintained Minimizing the production hours lost to maintenance Minimizing the cost of spare parts and supplies These cost savings come at a price, however. Some condition monitoring techniques are expensive and require specialist and experienced personnel for data analysis to be effective.

2 Process Model

Agile-Scrum software development: Scrum is an agile methodology where products are developed iteratively. Planning, sprints, stand-ups, and retrospectives are integral parts of scrum methodology. In this model we are going to use Vmodel XT in combination with Agile scrum.

3 Team Organization

- Scrum Master
- Front-end team
- Backend team
- Testing team
- Deployment team
- Maintenance team

4 Task Distribution

Since we are following Agile-Scrum methodology, we will be having bi-weekly scrums wherein we will discuss updates related to the tasks assigned and blockers if any. Since we are a small team of 4 people, every member will contribute in all the phases of the project life-cycle.

5 Requirement Management

5.1 Functional Requirement

- The application shall display statistical inferences drawn from the data along with the threshold that will be used to trigger predictive maintenance warning.
- The application shall be able to start/stop on user request.
- The application shall predict whether gearbox is healthy or maintenance is required due or gearbox has broken once it is started.
- The application shall graphically display the status of the gearbox based on the sensor values.

5.2 Non-functional Requirement

- The application shall continuously log information so that it can be used to diagnostics in case of system failure.
- The application should be able to operate irrespective of the size of the data.
- The application shall be platform independent and ensure the privacy and security of the data.
- The application shall be safeguarded against vulnerabilities.

6 Use Cases

A gearbox is a mechanical device that increases the output torque of a motor or changes its speed. The shaft of the motor is connected to one end of the gearbox, which delivers a particular output torque and speed dictated by the gear ratio according to the internal gear design of the gearbox.
The physical components of gearboxes vary depending on the type of gearbox, as well as between manufacturers. The majority of gearboxes are made of steel, such as iron, aluminum, and brass. As a result, it is open and susceptible to wear and tear throughout its lifespan.

When the gearbox fails, it is because the gears are worn out.

Vibration condition monitoring can be utilized as a technique for early detection of a wide range of defects in mechanical parts. Using the same technique, we can predict failures in gearboxes. For this we have used a dataset which was derived using four vibration sensors placed in four different direction inside the gearbox subjected to variable load. The data was recorded in 2 gear conditions: healthy and broken tooth, each under variable load.

6.1 Use-Case 1

Title: Application Start
Description: The application will start at the 'Statistical Inferences' page smoothly
Primary Actor: Client
Precondition: None
Postcondition: Application starts with 'Statistical Inferences' page.
Main Success Scenario: Application starts smoothly.

6.2 Use-Case 2

Title: Loading of 'Statistical Inferences' page
Description: The user should be notified with proper GUI if there is any delay in loading this page.
Primary Actor: Database
Precondition: Database has valid data and the user starts the application
Postcondition: 'Statistical Inferences' page loads up properly or the user gets notified about the delay if any
Main Success Scenario: 'Statistical Inferences' page loads up properly

6.3 Use-Case 3

Title: Starting 'Gearbox Maintenance & Failure Prediction'
Description: The application will inform the user if the gearbox needs maintenance.
Primary Actor: Client
Precondition: Client has triggered 'Gearbox Maintenance & Failure Prediction'
Postcondition: The user gets notified if the gearbox needs maintenance with a pop-up window.
Main Success Scenario: The user gets notified if the gearbox is healthy / needs maintenance or has broken and needs replacement

6.4 Use-Case 4

Title: Pop-up window for notifying the client
Description: The application will inform the user about the maintenance status of the gearbox with a pop-up.

Primary Actor:
Precondition: Client has triggered ‘Gearbox Maintenance & Failure Prediction’ using the Start button
Postcondition: The user gets notified with a pop-up window display the current status of the gearbox.
Main Success Scenario: The user gets notified if the gearbox is healthy / needs maintenance or has broken and needs replacement

6.5 Use-Case 5

Title: Loading of ‘Historical Logs’ page
Description: The application should display timestamped logs to the user
Primary Actor: Database
Precondition: ‘Gearbox Maintenance & Failure Prediction’ has already been triggered by the client at least once
Postcondition: The application should display relevant logs to the client
Main Success Scenario: Logs are displayed to the client

6.6 Use-Case 6

Title: Display of gearbox status using an GUI Indicator
Description: The application should display the gearbox status to the client with the help of a GUI Indicator
Primary Actor: -
Precondition: ‘Gearbox Maintenance & Failure Prediction’ has already been triggered by the client at least once
Postcondition: The application should green for healthy gearbox, yellow for a gearbox which needs maintenance and red for a gearbox that has broken and needs replacement
Main Success Scenario: The application displays the gearbox status to the client

7 List of Deliverables

- Requirement Specification Document
- Functional Specification Document
- Design Document
- Unit Test Results
- Integration Test Results
- System Test Results
- Deployment Document
- User Manual

8 Milestone

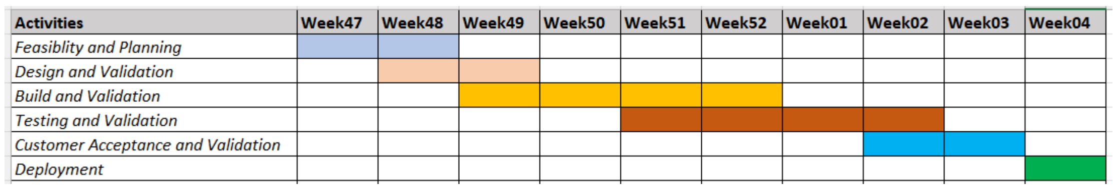

Fig. 1. Scheduled Milestone

9 Risk involved in the Project

- Improper interpretation of customer requirements
- Inadequate requirement analysis
- Inter component interactions

10 Requirement Analysis

10.1 Functional Requirement

1. The application shall display statistical inferences drawn from the data along with the threshold that will be used to trigger predictive maintenance warning.
2. The application shall be able to start/stop on user request.
3. The application shall predict whether gearbox is healthy / maintenance is due or gearbox has broken once it is started.
4. The application shall graphically display the status of the gearbox based on the sensor values

10.2 Non-Functional Requirement

1. The application shall continuously log information so that it can be used to diagnostics in case of system failure.
2. The application should be able to operate irrespective of the size of the data.
3. The application shall be platform independent and ensure the privacy and security of the data.
4. The application shall be safeguarded against vulnerabilities.

11 Effort Estimation

Calculating Unadjusted Functional Point

Function type	Simple	Average	Complex	Considered For Project	Count	Total
External Inputs	3	4	5	3	2	6
External Output	4	5	7	5	2	10
External Inquiries	3	4	6	5	3	12
Internal Logical Files	7	10	15	10	3	30
External Interface Files	5	7	10	5	0	0
						58

UFP = 58

Sr. No	Characteristics	$(0-5)$
1	Data communications	3
2	Distributed data processing	0
3	Performance	3
4	Heavily used configuration	0
5	Transaction rate	5
6	On-Line data entry	0
7	End-user efficiency	0
8	On-Line update	0
9	Complex processing	5
10	Reusability	2
11	Installation ease	2
12	Operational ease	2
13	Facilitate change	3
14	Multiple sites	0
		25

CAF $= 0.65 + (0.01 * Fi)$

CAF $= 0.65 + (0.01 * 25) = 0.9$

FPC $= UFP * CAF =$58 $* 0.9 = 52.2$ 52.

12 Constructive Cost Model

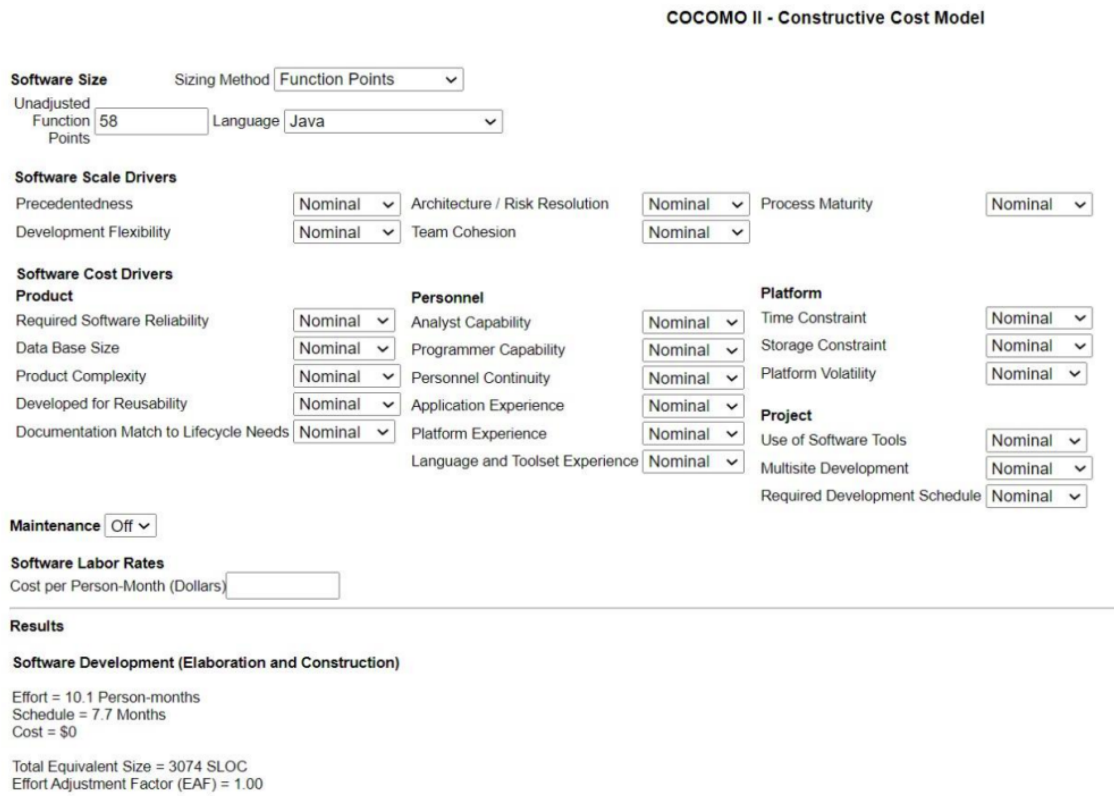

Fig. 2. COCOMO II- Constructive Cost Model

13 Data Flow Diagram

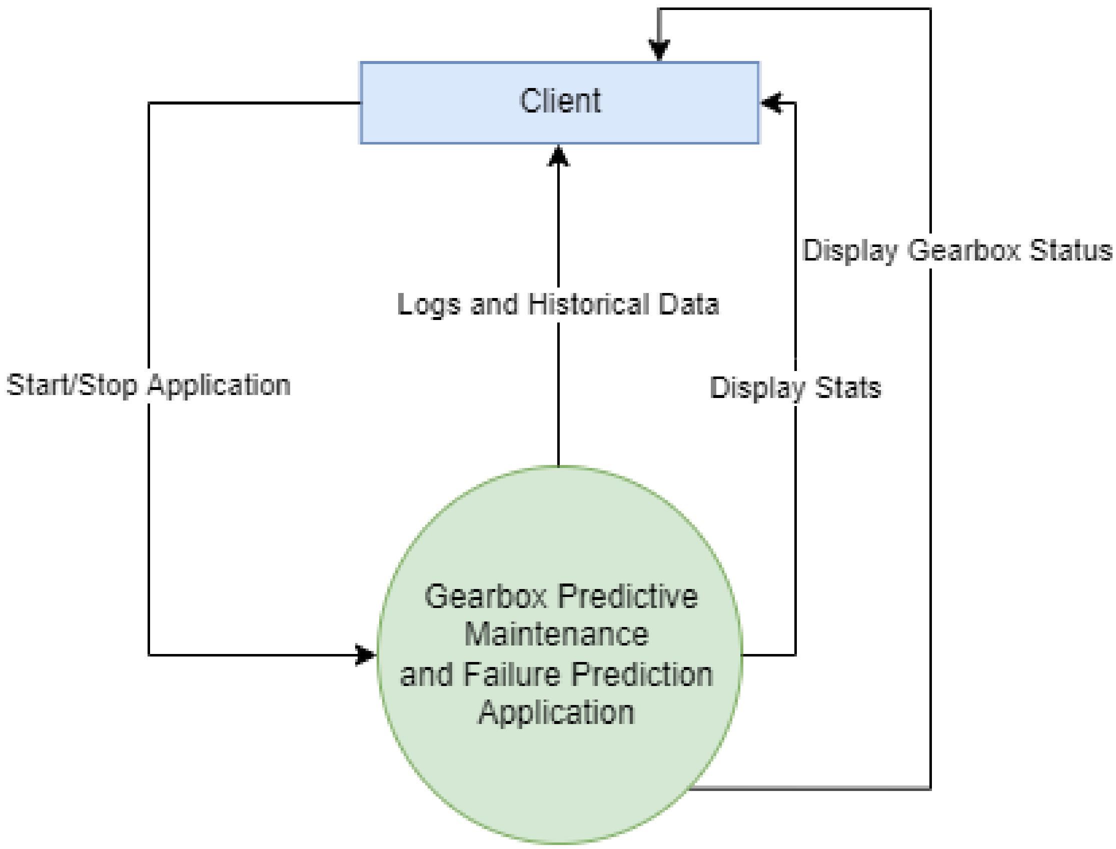

Fig. 3. Data Flow Diagram Level-0

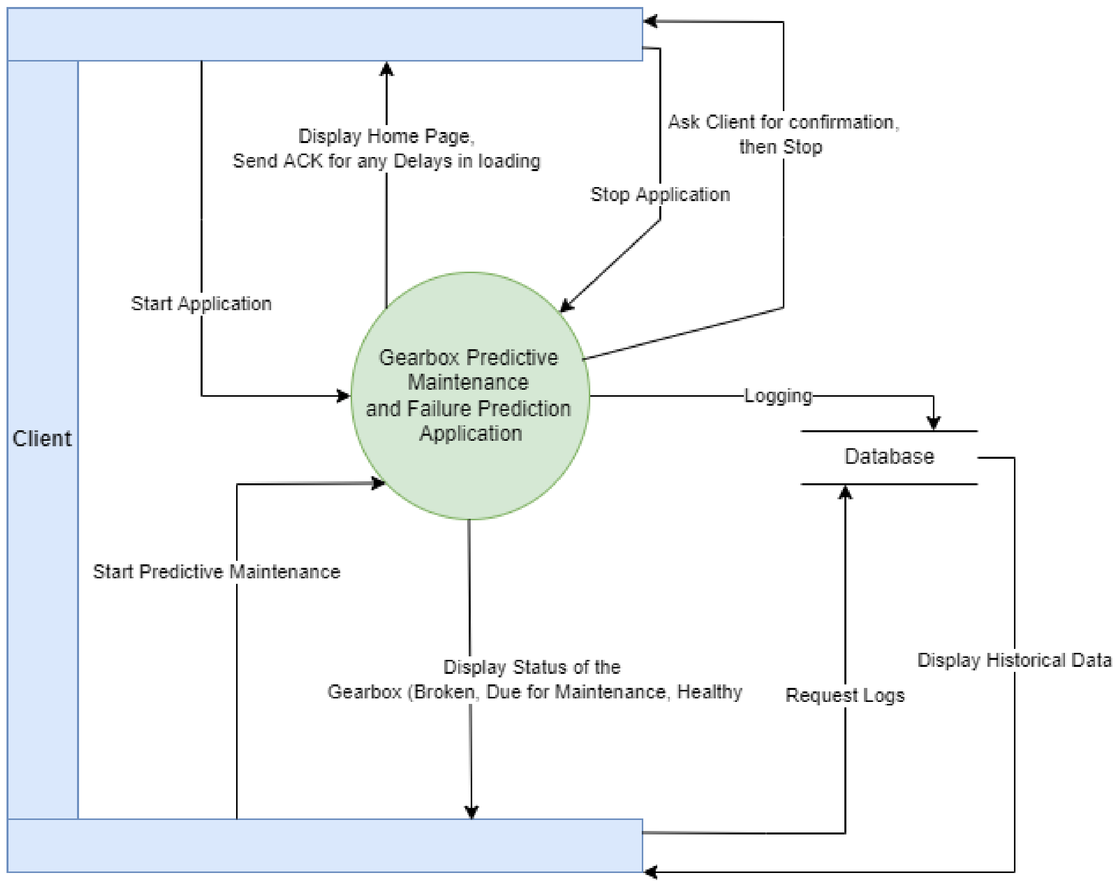

Fig. 4. Data Flow Diagram Level-1

14 Control Flow Diagram

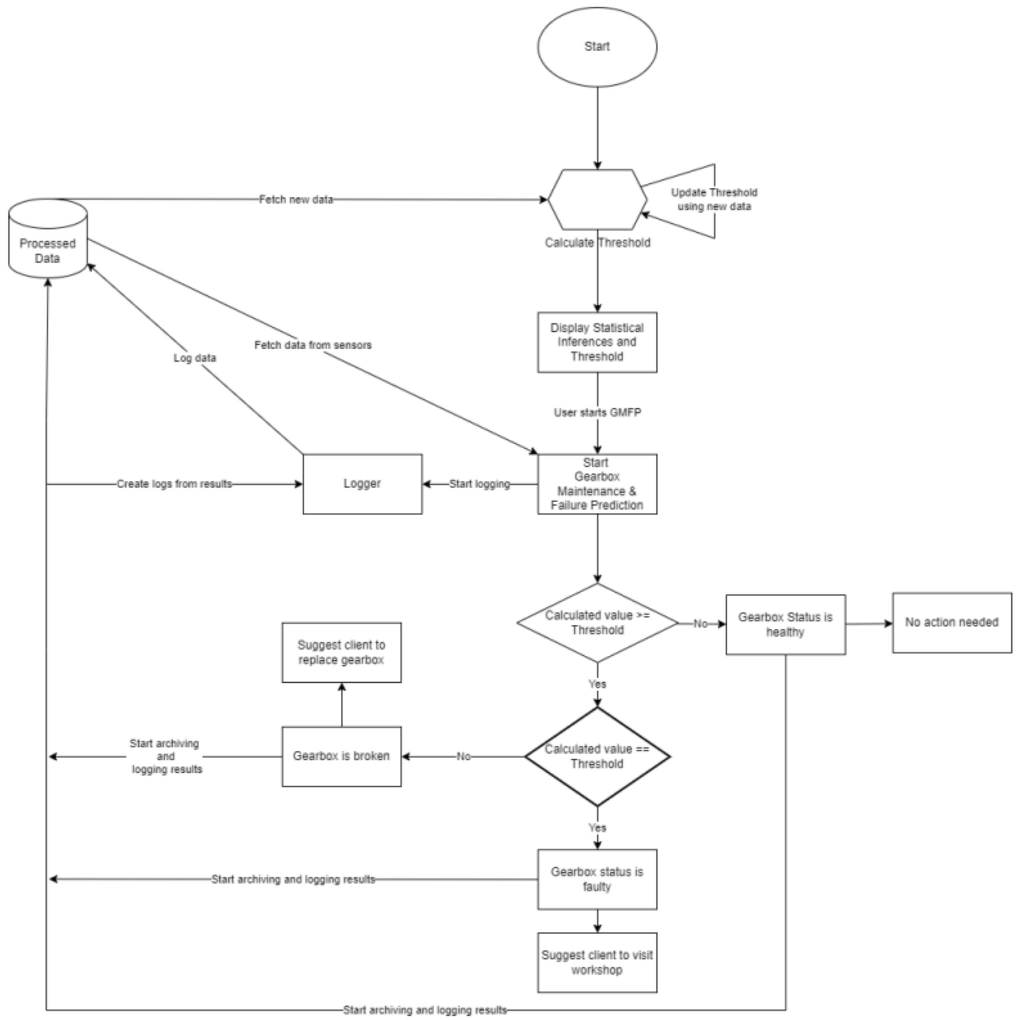

Fig. 5. Control Flow Diagram

15 Use Case Diagram

Fig. 6. Use Case Diagram

16 Sequence Diagram

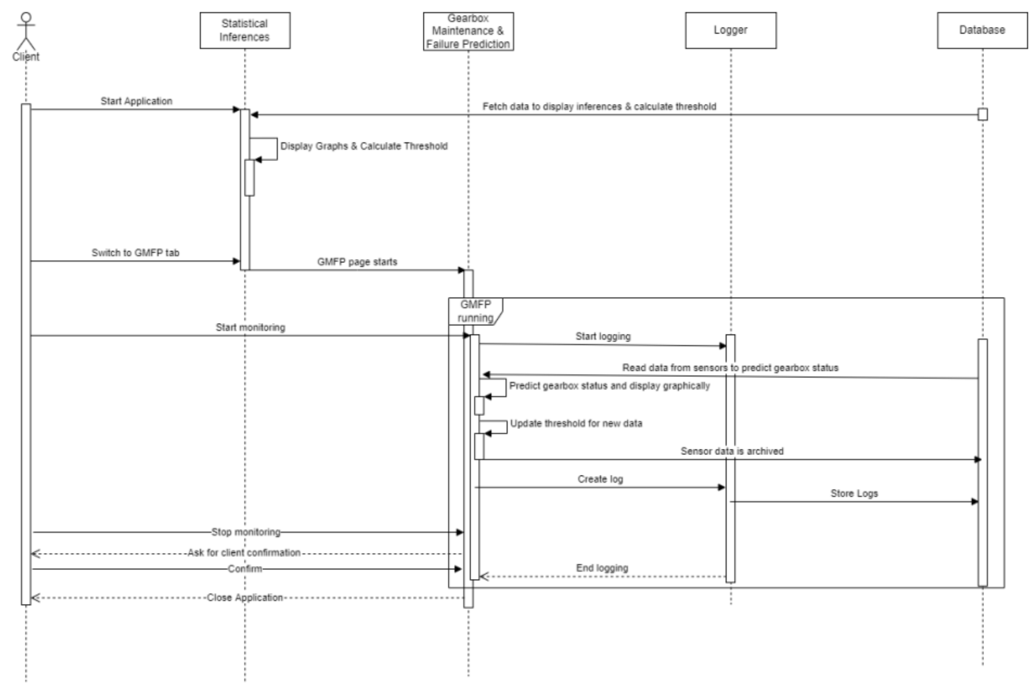

Fig. 7. Use Case Diagram

17 System Architecture

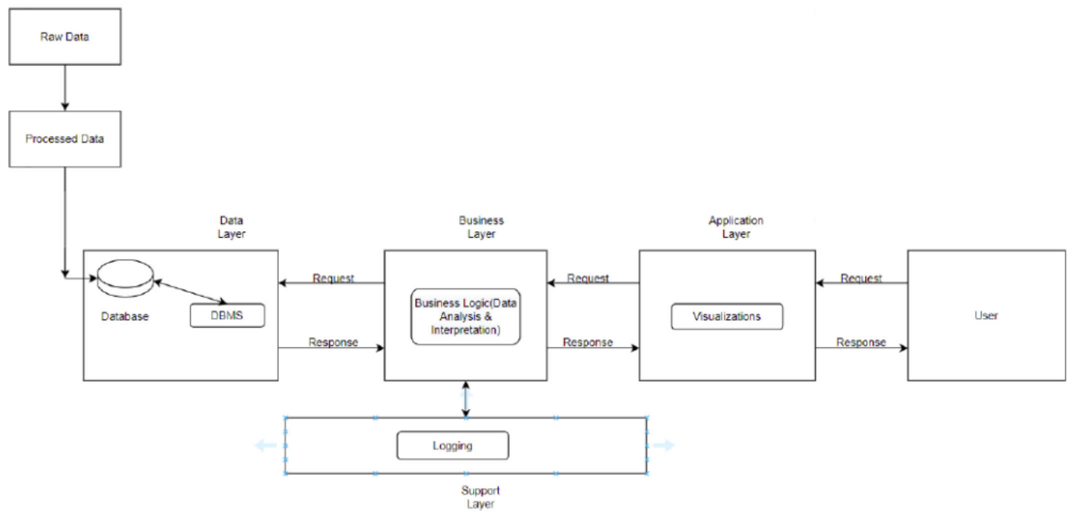

Fig. 8. Use Case Diagram