Ollscoil Teicneolaíochta an Oirdheiscirt
South East Technological University

Fig. 1 SETU logo

Software Development Year 4 Project

Planned and Unplanned Maintenance Application

Functional Specifications

Fig. 2 Maintenex logo

Student: Nebojsa Kukic

Student No: C00283550

Date: 2025/2026

Supervisor: Tom Corocoran

Introduction

This is a functional specification for my proposed web application which can help predict when machines are likely to fail and to automatically schedule maintenance work such as servicing and repairs. The idea is to build a system that would be easy to use by engineers and technicians and managers, even if they don't have a background in data science and machine learning.

The application will have realistic synthetic data on machinery, such as breakdowns, cycle running time, heat, vibration, load, rpm, etc. From this the application can predict the failure risk of a machine down to a rough percentage. For example, the app would be able to predict a certain machine to have a failure point of 80% in the next 30 days based on previous data of machinery that have failed, due to some failure points.

Overall, the goal of this project to reduce unexpected machine breakdowns, save time on planning, and make planning more organized using AI-driven prediction and intelligent scheduling of services/repairs.

Technologies

For the technologies section I have done research to decide on what would be the best technologies to use in this project.

1. Python

Fig. 3 Python logo

Description:

Python will be used for implementing machine learning models to predict machine failures and calculate the probability of failure for each machine. The reason I picked Python is because of the abundance of many powerful libraries such as, Pandas, NumPy, and Scikit-Learn. It integrates in a nice fashion with Flask, which I will also be using.

2. Flask

Fig. 4 Flask logo

Description:

Flask is a lightweight web framework that is very good for quick web development. There is no unnecessary overhead compared to other frameworks, such as Django. It's simplicity and flexibility is perfect to use seamlessly between the frontend and machine learning models.

3. React

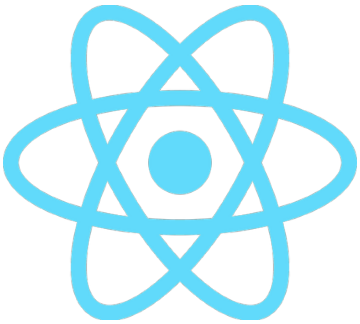


Fig. 5 React logo

Description:

React is a JavaScript library used to make aesthetically pleasing, dynamic and responsive web interfaces. This is a key when it comes to users to have a consistent layout to view machine failure predictions, maintenance schedules and alerts.

4. Simple Mail Transfer Protocol (SMTP)

Fig.6 SMTP logo

Description:

Simple Mail Transfer Protocol will be used to send automated email alerts to technicians and engineers when a machine requires planned and unplanned maintenance. For example, the system can send an email to a technician that a machine has a possibility of failure of 70% or a machine is ready for a scheduled service by x. SMTP integrates seamlessly with Python and Flask.

5. JSON

Fig.7 JSON logo

Description:

JSON will be used to store the machine maintenance specification such as, service intervals and configuration settings. These JSON configuration files will be adjustable as some parts require difference service interval times. JSON is lightweight and easy to read and integrates very well with Python and Flask.

6. MySQL

Fig.8 MySQL logo

Description:

When it comes to the backend MySQL will be used to store the necessary data. Here the machine information, maintenance logs, user accounts and scheduling data will be stored. It is useful in providing reliable and structured storage that can be accessed and analyzed efficiently.

Description

When it comes to the maintenance, especially on an industrial scale, machines require regular maintenance to operate efficiently, but scheduling maintenance can be very challenging, due unexpected breakdowns, limited availability of technicians or delayed parts.

These issues increase the downtime of machines, raise the repair costs and this reduce the production efficiency.

This proposed system will address these issues by:

- Predictive Maintenance: Using historical machine data to predict the failure probabilities of machines for the next 30-180 days.
- Automatically Maintenance Scheduling: Using the data in the JSON files, the system can alert technicians with SMTP on when a service is due.

Users will be able to:

- View all machines and their current health status.
- See the probability of failures of machines for example: Machine 7 has a 70% risk of failing.
- Get an optimized maintenance schedule.
- Log the maintenance work and update any repairs done.

Background

My motivation for developing a Planned and Unplanned Maintenance Application rises from my previous work experience at Bausch & Lomb. A long leading giant in ocular health. They have a tremendous amount of machinery making and packaging contact lenses and these machines break down on a frequent basis. They use a similar system to track the scheduling of servicing and maintenance of these machines. My motivation comes from this system as in today's world it is critical to have such applications to track, predict and schedule maintenance intervals for such machines.

I am driven to leverage my skills in AI and machine learning to create a solution that reduces the downtime of machinery, to predict failure of these machines and intelligent scheduling seamlessly into one application.

Core Features

1. Predictive Maintenance:

The application will use real-time data and machine learning algorithms to predict future machine failures before they occur, reducing downtime and maintained costs.

2. Automated Scheduling:

Automatically generate the most efficient scheduling for services and repairs, optimizing the resources, task and number of technicians, thus improving the efficiency and minimizing conflicts

3. Automated Alert System:

Automatic alerts through Email (SMTP) to inform technicians of anomalies, future failures and future services to be done.

Functional Requirements

1. Predictive maintenance:

The application must analyze the current machine data compared to historical data to predict future failures and generate maintenance recommendations

2. Automated Maintenance:

The application must automatically create optimized schedules for maintenance, services and repairs, considering the resource availability and priority.

3. Automated Alert System:

The application must send real-time alerts via email to technicians when anomalies occur or when a service is required

4. User Management:

The application must have different user roles, with role-based access across the application (admin, technician, engineer and managers)

5. Reporting and Analysis

The application must be able to generate reports on maintenance history and performance.

Non-Functional Requirements

1. Performance:

The application must process large amounts of data and give predictions/alerts within seconds.

2. Reliability:

The application must operate with minimal downtime

3. Scalability:

The application must support the increasing addition of new features, data, and users.

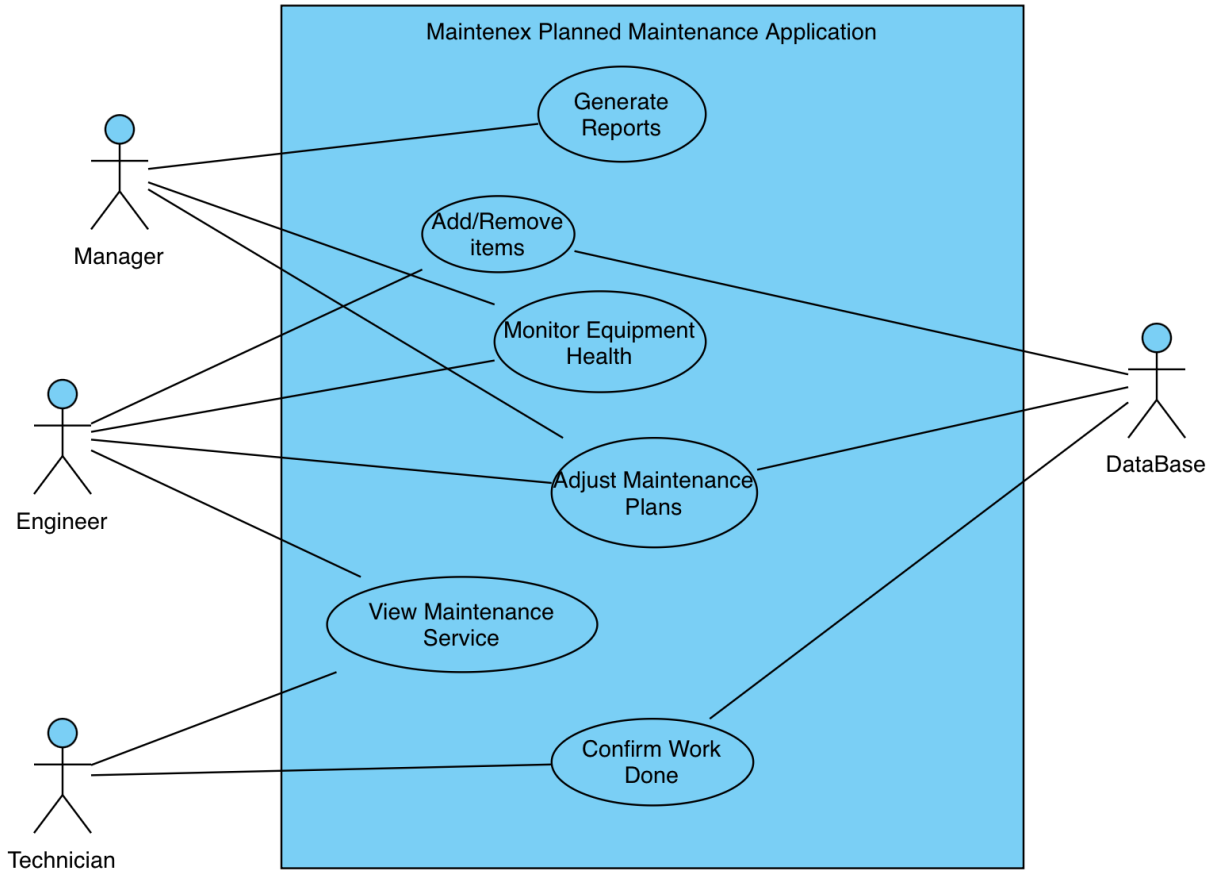
4. Security:

The application must have secure authentication, and encryption to protect sensitive data.

5. Usability

The application must have a simple and concise UI to allow user to perform their tasks with minimal training.

Use Case Diagram



Use Case: General Use of application

Actors: Manager, Engineer, Technician, Manager

Description: Personal can use this application to look at previous machine logs, generate reports, confirm work done and monitor the current health of machines.

References

Figures:

Fig. 1: SETU Logo: https://www.kilkennychamber.ie/directory-members/listing/setu-faculty-of-life-long-learning/

Fig. 2: My Logo

Fig. 3: Python Logo: https://en.wikipedia.org/wiki/Python_%28programming_language%29

Fig. 4: Flask Logo: https://commons.wikimedia.org/wiki/File:Flask_logo.svg

Fig. 5: React Logo: https://commons.wikimedia.org/wiki/File:React-icon.svg

Fig. 6: SMTP Logo: https://www.mlypn.com/2023/01/smtp-protocol-summary.html

Fig. 7: JSON Logo: https://dev.to/techlearners/what-is-json-and-why-do-you-need-it-21nd

Fig. 8: MySQL Logo: https://seeklogo.com/vector-logo/96578/mysql

General Information:

Flask documents: https://flask.palletsprojects.com/en/stable/

React documents: https://react.dev/learn

SMTP Guide: https://www.socketlabs.com/blog/beginners-smtp-guide/

Planned Maintenance Information: https://upkeep.com/learning/planned-maintenance/#:~:text=Planned%20maintenance%20is%20the%20process,it%20needs%20to%20be%20done.

Python Documents: https://docs.python.org/3/