Ollscoil Teicneolaíochta an Oirdheiscirt
South East Technological University

Fig 1 SETU logo

Software Development Year 4 Project

Project Title: Maintenex

Design Document

Fig. 2 Maintenex logo

Student Name: Nebojsa Kukic

Student Number: C00283550

Supervisor: Tom Corcoran

Date: 2026

Table of Contents

Introduction

This design document outlines a detailed plan on how Maintenex will be developed following its outlined requirements. This document explains how Maintenex will fulfill its requirements by showing the design and technologies used. The primary purpose of this application is to predict the probability of failure on a fake/virtual machine using synthetically generated data and machine learning techniques. In addition, the system manages maintenance of general machines through scheduling, logging, and alert notifications.

We are using a fake/virtual machine because I do not have access to real plant machinery. The system also manages maintenance for general assets/machines. When it comes to the general machines technicians will have a tool to see previous services done, log work, see the current condition of machines along with being able to receive real-time alerts on the machine's needs. The application will have an emphasis on creating an easy to learn user interface which will be aesthetically pleasing.

Technologies

Much research has been exercised on what technologies will be used to create the Maintenex application. Below is an overview on what technologies were chosen to use for this application and their roles for this application to be built.

Python: serves as the main programming language of this application. It is powerful when it comes to the generation of synthetic data and machine learning techniques, thanks to its powerful libraries, such as Pandas, NumPy, and Scikit-Learn.

Flask: Will be used to create and structure a secure web application. It will drive the backend of the application, thus providing a powerful connection between the web interface and database.

HTML/JavaScript/CSS: will be used to create the front end of the application, this helping with created a UI which is both easy to use and aesthetically pleasing.

Simple Mail Transfer Protocol: Will be used to send automated email alerts to technicians and engineers when a machine requires planned and unplanned maintenance. It will also be used for when a user forgets their password.

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.

Creation of Synthetic Data

The use of real data from machines is not available. The generation of synthetic data will be need. This is a complicated task to do as the data must be realistic and follow the engineering rules of how machines operate. With the help of my previous engineering supervisor and resources, the generation of realistic synthetic data will be created with the help of Python's NumPy and Pandas following real-world engineering rules.

Here is an example of some of the data that was generated following these real-world engineering rules:

machine_id	cycle	day	day_in_cycle	temperature	vibration	rpm	load	service_flag	failure_event
1	1	1	1	70.64	0.0771	1448.2	88.3	0	0
1	1	2	2	71.05	0.0879	1445.2	86.4	0	0
1	1	3	3	72.63	0.0931	1444.8	89.6	0	0
1	1	4	4	68.74	0.0887	1448.5	77.6	0	0
1	1	5	5	67.32	0.0596	1447.8	59.7	0	0
1	1	6	6	66.6	0.083	1446.1	48.1	0	0
1	1	7	7	67.74	0.0726	1450.0	58.5	0	0
1	1	8	8	69.5	0.0859	1448.8	75.5	0	0
1	1	9	9	69.29	0.1055	1441.9	73.2	0	0
1	1	10	10	66.51	0.0602	1444.5	45.0	0	0
1	1	11	11	69.66	0.0779	1447.7	75.7	0	0
1	1	12	12	67.03	0.0881	1449.9	48.6	0	0

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.

These applications are extremely complex such as that from SAP and PEMAC assets and it would be good if an application for maintaining a variety of items in an easier and more simple fashion for smaller businesses such

Database

The database is a critical part of the Maintenex system because it stores and manages all the core information the application depends on. Without it, the system would not be able to track machines, manage users, or generate alerts.

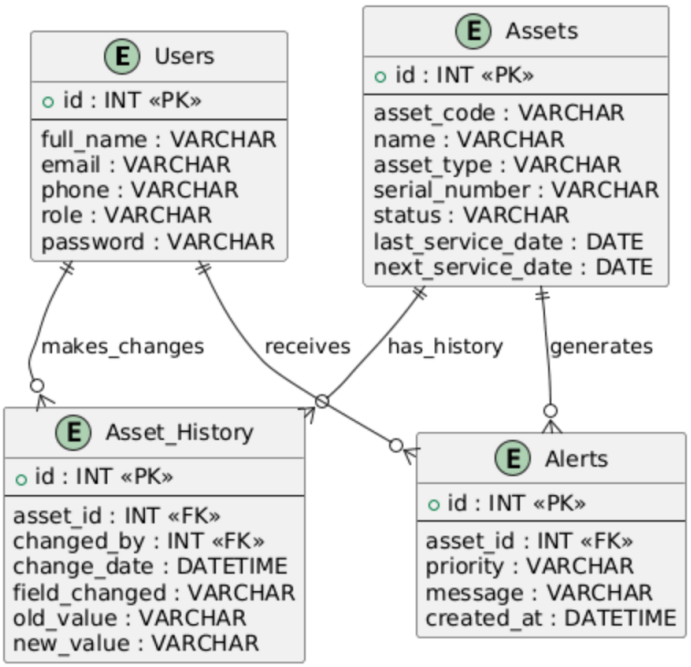
All user accounts are stored in the database, including roles such as Admin, Supervisor, and Technician. This allows the system to control access and ensure that only authorised users can perform certain actions.

The database also stores all asset information, including service dates and machine status. The logic that highlights machines as “due soon” or “overdue” depends entirely on this stored data. When a machine is serviced, the updated dates are saved in the database, ensuring accurate tracking over time.

In addition, maintenance history and alerts rely on structured database records. This ensures traceability, accountability, and proper documentation of all servicing activities.

Overall, the database acts as the backbone of Maintenex, connecting the frontend, backend logic, and predictive features into one reliable and structured system.

Entity Relationship Diagram (ERD)



Dashboard

The dashboard is the main overview page of the Maintenex system. It gives users a quick and clear summary of what is happening with all the assets in one place.

On the dashboard, users can see important information such as which machines are due for service, which are overdue, and any active alerts. This helps technicians and supervisors quickly understand what needs attention without having to search through the whole system.

It also shows key asset details, making it easy to check the current status of machines. The colour coding (such as yellow for due soon and red for overdue) allows users to instantly recognise priority issues.

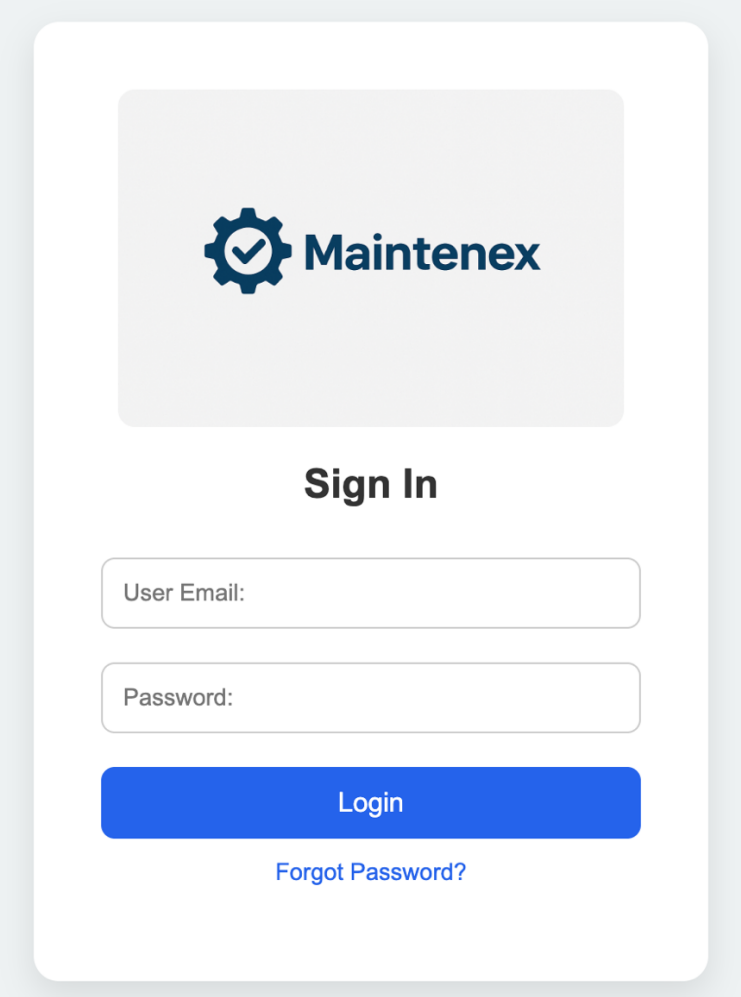
Overall, the dashboard is designed to save time and make maintenance management simple by showing the most important information at a glance.

UI/UX

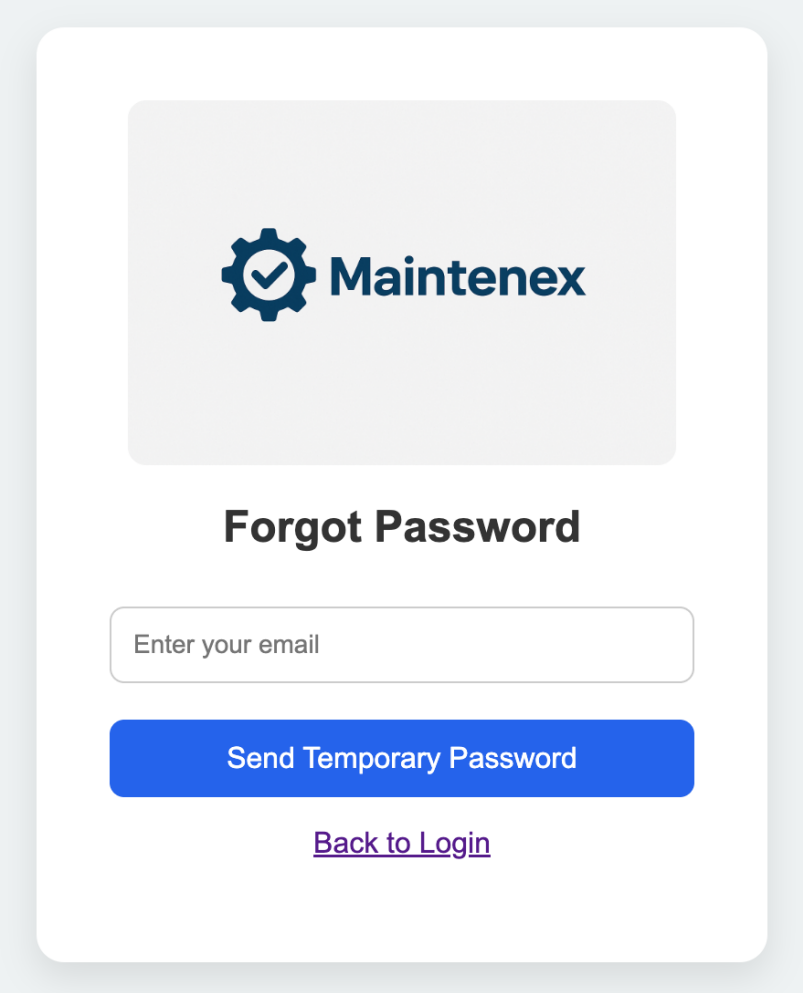
The creation of a great User Interface and User Experience is a complicated procedure. The UI this case must follow a simple design for technicians to be able to use it, along with the fact of being aesthetically pleasing to the eye.

The website will implement a few different pages to have access to the different features. At the top of the page there will be a navbar and this is the primary route of navigating the webpage with a hamburger dropdown.

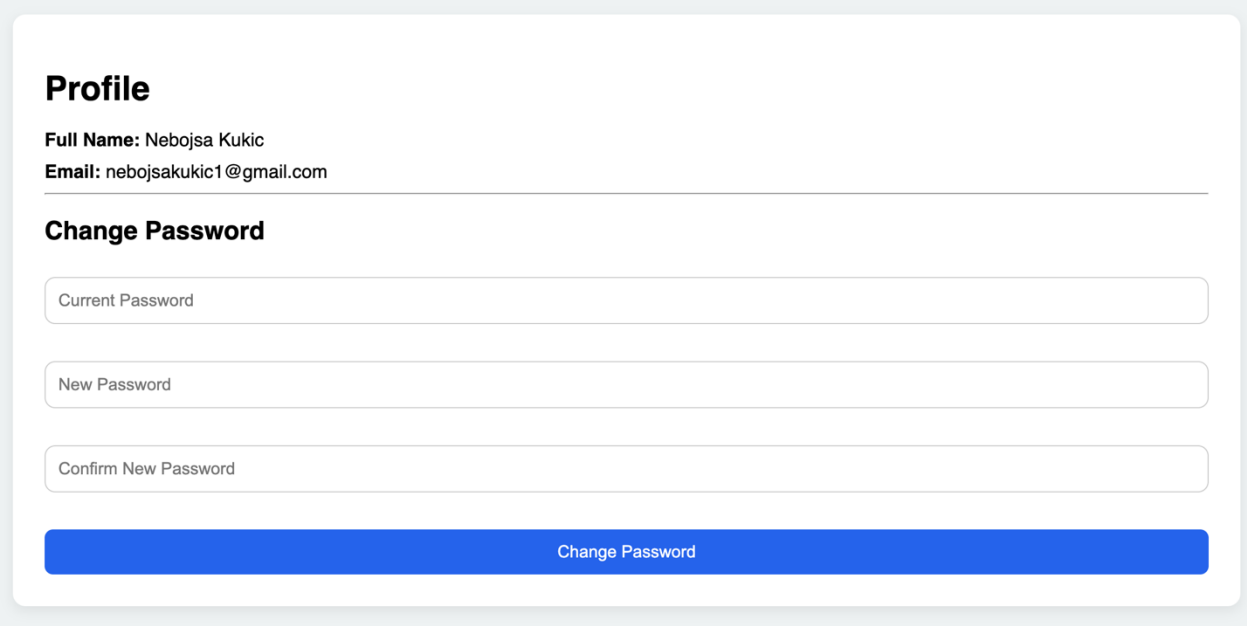
Maintenex Login Page



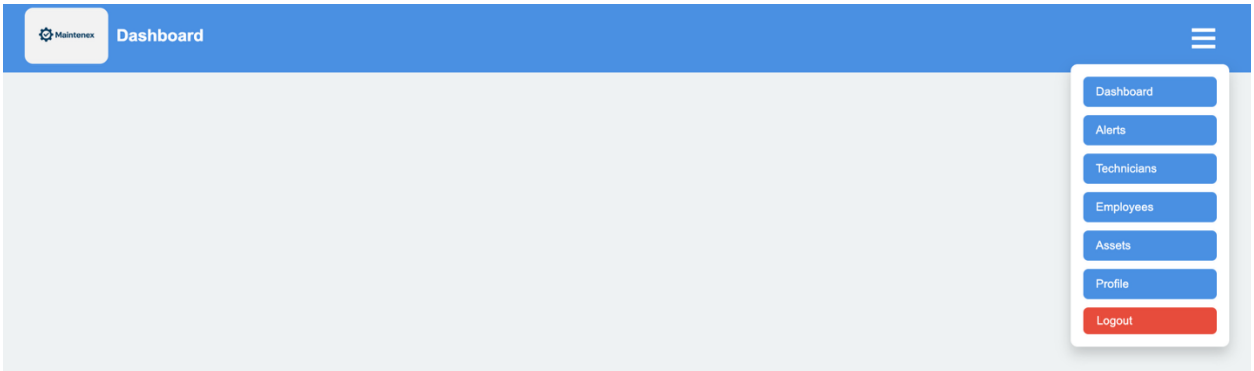
Forgot Password Page



Change Password Page (With SMTP)



Nav Bar



Possible Dashboard Design

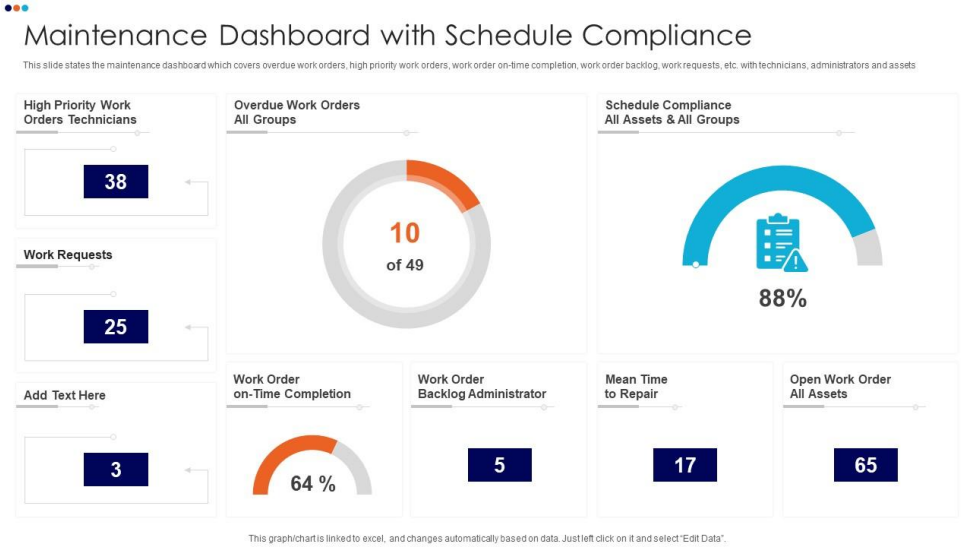
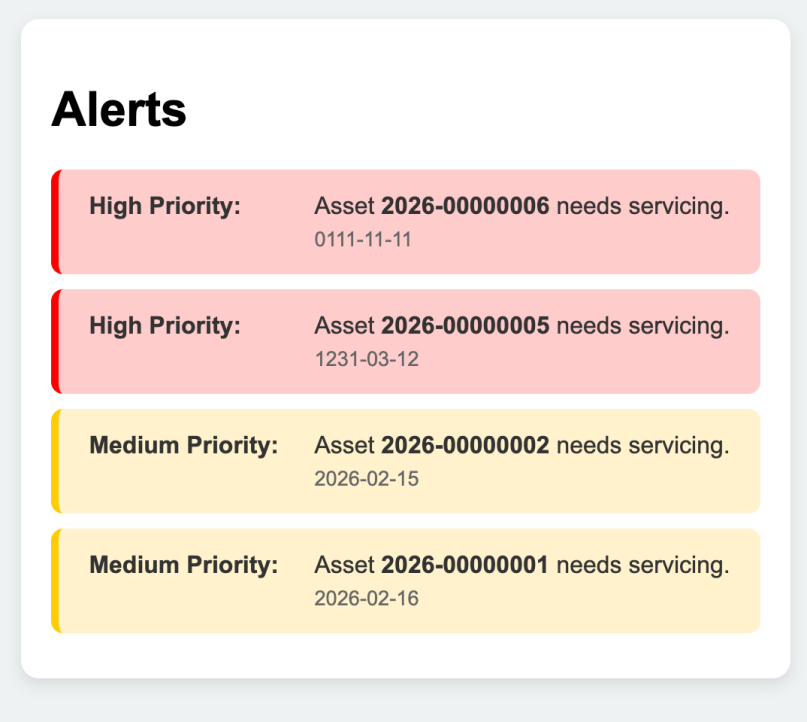


Fig. 3 Possible Dashboard Design

Alert Notifications



Show All Employees Page

admins

Name	Email	Phone
nebojsa kukic	nebojsakukic1@gmail.com	0838935494

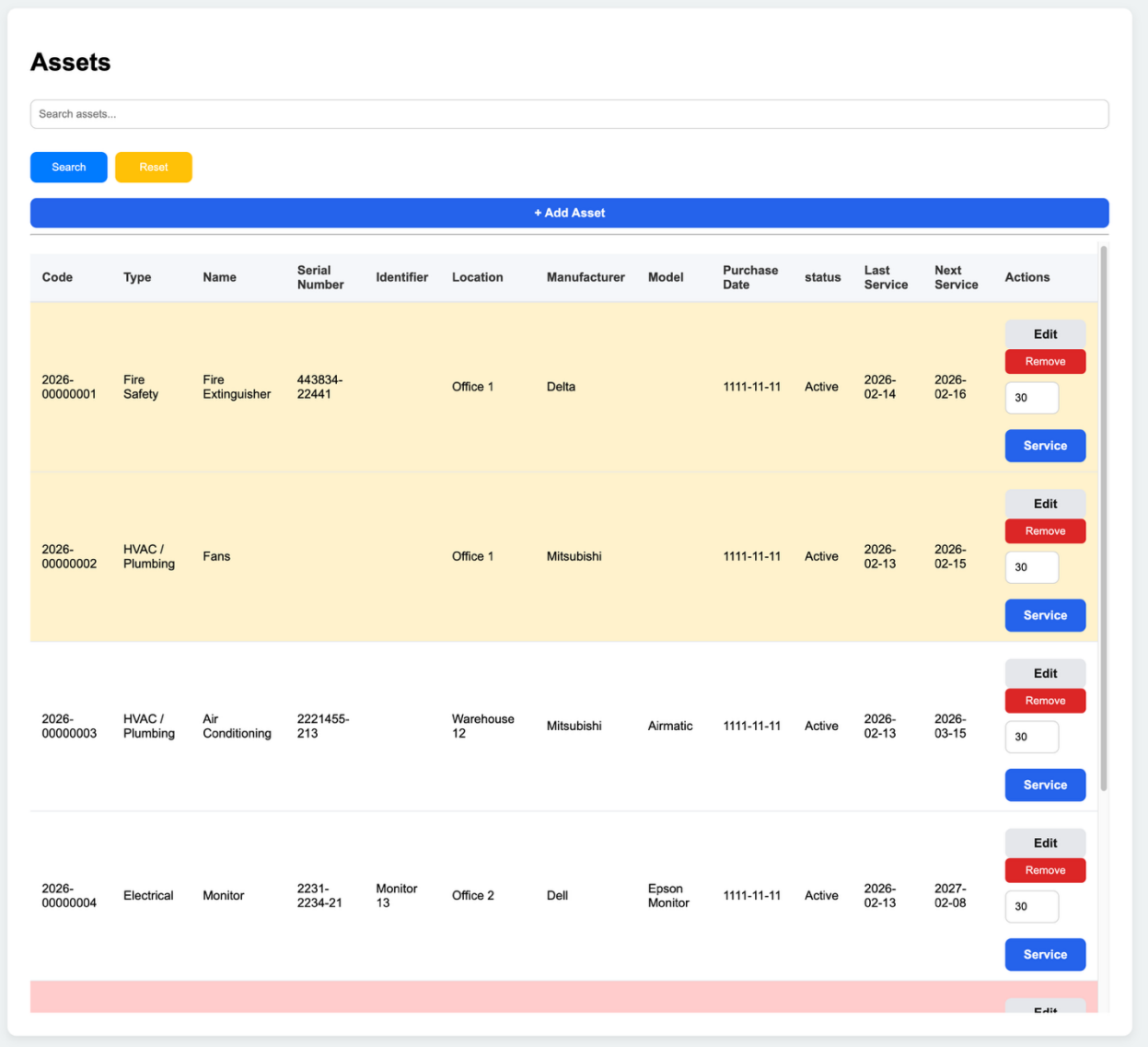
supervisors

Name	Email	Phone
Aimee Delaney	aimmedelaney@setu.ie	0834434120

technicians

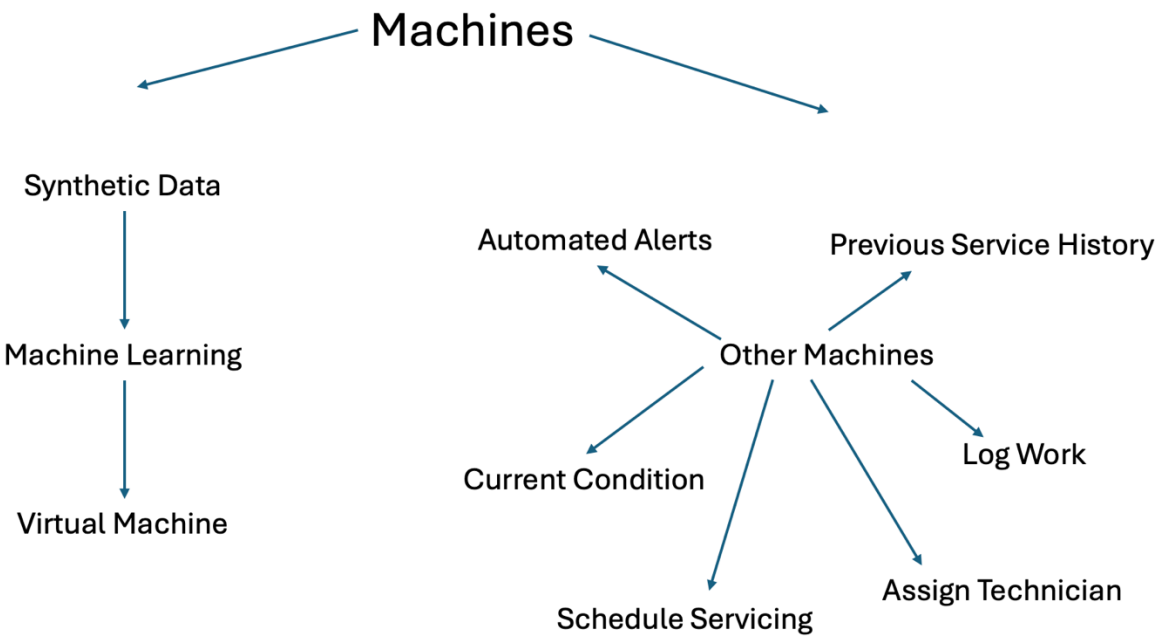
Name	Email	Phone
Bob Flavin	bobby.flav@setu.ie	0873542943
Sam Beckett	sambeck@setu.ie	0854361123

Review Assets (Admin's Perspective)



The difference between Machines

There will be two sections of the application when it comes to the machines/assets. They will split on the fake/virtual machine which show the predicting power of the ML on the fake/virtual machine using the synthetically generated data, and the general machines in which technicians will have work to do, check previous services, schedule work, log work, along with real-time alerts notifying technicians on any work that must be carried out on any machines that are needing any servicing done.



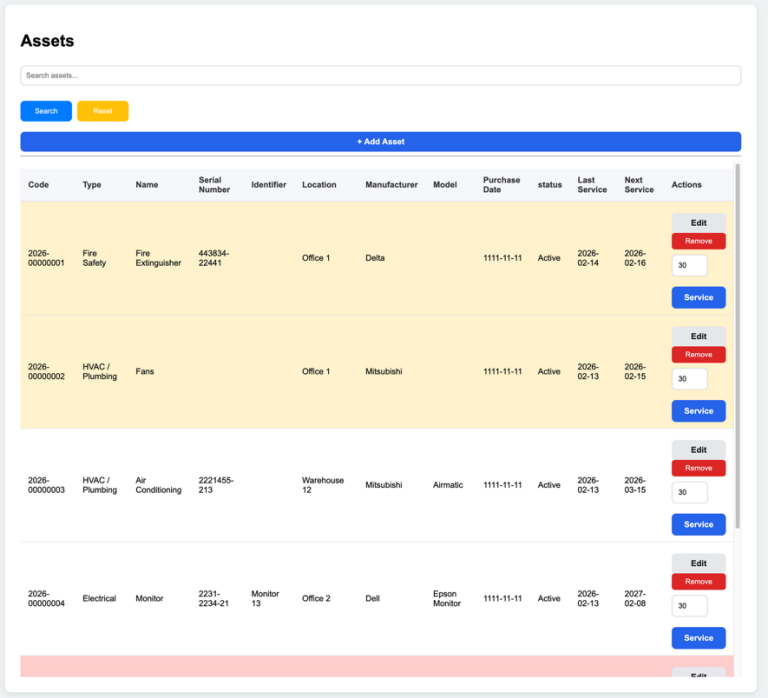
Fake/Virtual Machine

When it comes to the FakeVirtual Machine. Synthetic data has been generated to be used with ML to predict the possibility of failure of on the fake/virtual machine that is being used when adjusting some parameters. The general route would be using a random Forest model on the prediction on the probability of failure of the machine using the rule-based synthetic data.

machine_id	cycle	day	day_in_cycle	temperature	vibration	rpm	load	service_flag	failure_event
1	1	1	1	70.64	0.0771	1448.2	88.3	0	0
1	1	2	2	71.05	0.0879	1445.2	86.4	0	0
1	1	3	3	72.63	0.0931	1444.8	89.6	0	0
1	1	4	4	68.74	0.0887	1448.5	77.6	0	0
1	1	5	5	67.32	0.0596	1447.8	59.7	0	0
1	1	6	6	66.6	0.083	1446.1	48.1	0	0
1	1	7	7	67.74	0.0726	1450.0	58.5	0	0
1	1	8	8	69.5	0.0859	1448.8	75.5	0	0
1	1	9	9	69.29	0.1055	1441.9	73.2	0	0
1	1	10	10	66.51	0.0602	1444.5	45.0	0	0
1	1	11	11	69.66	0.0779	1447.7	75.7	0	0
1	1	12	12	67.03	0.0881	1449.9	48.6	0	0
1	1	13	13	67.66	0.0577	1445.3	49.5	0	0
1	1	14	14	69.79	0.0819	1447.3	71.5	0	0
1	1	15	15	65.48	0.0746	1451.3	51.7	0	0
1	1	16	16	75.2	0.1024	1446.5	95.5	0	0
1	1	17	17	69.3	0.082	1449.1	75.0	0	0
1	1	18	18	68.95	0.0766	1446.1	71.6	0	0
1	1	19	19	69.16	0.055	1449.5	57.7	0	0
1	1	20	20	68.08	0.068	1447.0	55.9	0	0
1	1	21	21	68.39	0.0674	1446.7	70.3	0	0
1	1	22	22	68.09	0.0845	1445.7	66.3	0	0
1	1	23	23	68.55	0.077	1446.0	54.1	0	0
1	1	24	24	69.24	0.0926	1446.8	61.0	0	0
1	1	25	25	69.65	0.0618	1446.4	71.5	0	0

General Machines

Here we will have the storing on all the rest of the normal assets/machines. Most things can be stored such as furniture, electronics, vehicles, plant machines.



Security Considerations

When it comes to the security aspect of the application. A few key considerations were taken when designing and developing this application.

First, the user passwords are all fully hashed and not stored as plain text files inside the database. This adds an important layer of security especially if someone were to gain access to the database and see everyone's security passwords.

Secondly the system uses a role-based (Admin, Supervisor, Technician) access approach when navigating the Maintenex website. This allows for users to only have access to certain pages, features and perform certain actions. This is very important as this prevents users from have access to sensitive data and actions, along with preventing inappropriate changes and deletions from being made.

Finally, all users must be logged in to access protected parts of the website. If they are not authenticated, they are redirected back to the login page. This goes a long way when it comes to protecting sensitive information from unauthorized individuals/users, this is why Flask is a great module when it comes to developing website, it lets you develop websites in an easy way, but also in a safe manner.

Database Tables

Users Table

This table will store all the information about the users, including their roles (admin, supervisor, or technician), department and hashed password for users to login. This table is very important as it allows for system permissions based on users' access.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	password	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	full_name	varchar(100)	utf8mb4_general_ci		Yes	NULL			Change Drop More
4	user_email	varchar(255)	utf8mb4_general_ci		Yes	NULL			Change Drop More
5	phone	varchar(20)	utf8mb4_general_ci		Yes	NULL			Change Drop More
6	role	enum('admin', 'supervisor', 'technician')	utf8mb4_general_ci		Yes	technician			Change Drop More
7	department	varchar(100)	utf8mb4_general_ci		Yes	NULL			Change Drop More
8	created_at	timestamp			No	current_timestamp()			Change Drop More
9	last_login	timestamp			Yes	NULL			Change Drop More
10	is_active	tinyint(1)			Yes	1			Change Drop More

Assets Table

This table is to store information on all the general assets that are owned/used by a company. It generally can range from furniture to HVAC systems to Vehicles.

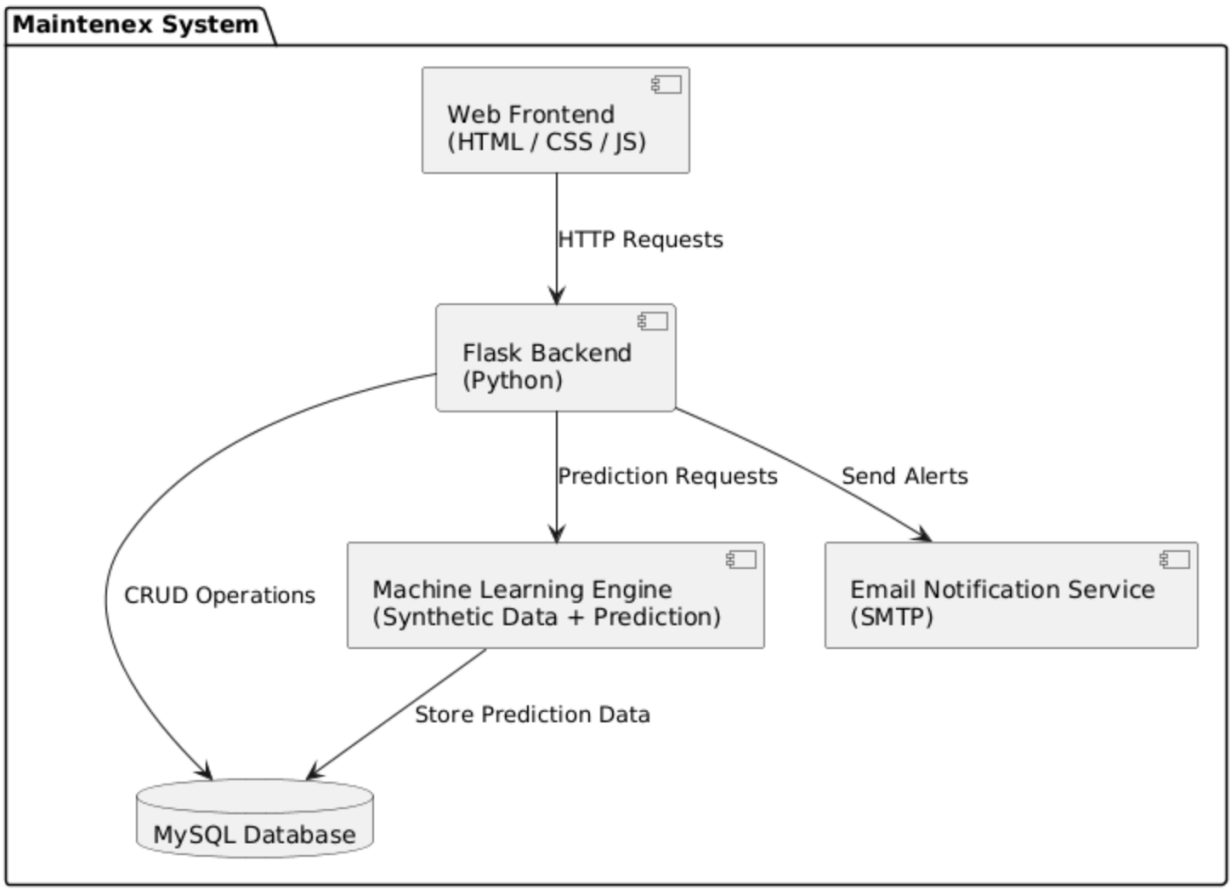
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	asset_code	varchar(20)	utf8mb4_general_ci		Yes	NULL			Change Drop More
3	asset_type	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
4	name	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More
5	serial_number	varchar(100)	utf8mb4_general_ci		Yes	NULL			Change Drop More
6	identifier	varchar(50)	utf8mb4_general_ci		Yes	NULL			Change Drop More
7	location	varchar(100)	utf8mb4_general_ci		Yes	NULL			Change Drop More
8	manufacturer	varchar(100)	utf8mb4_general_ci		Yes	NULL			Change Drop More
9	model	varchar(100)	utf8mb4_general_ci		Yes	NULL			Change Drop More
10	purchase_date	date			Yes	NULL			Change Drop More
11	status	enum('Active', 'Maintenance', 'Inactive', 'Retired...	utf8mb4_general_ci		Yes	Active			Change Drop More
12	created_at	timestamp			No	current_timestamp()			Change Drop More
13	last_service_date	date			Yes	NULL			Change Drop More
14	next_service_date	date			Yes	NULL			Change Drop More

Asset History Table (for historical tracking and auditing purposes)

This is an important table that allows for general lookup on individual assets whether they were edited and when they were serviced. Along with the ability for easy management when it comes to auditing purposes.

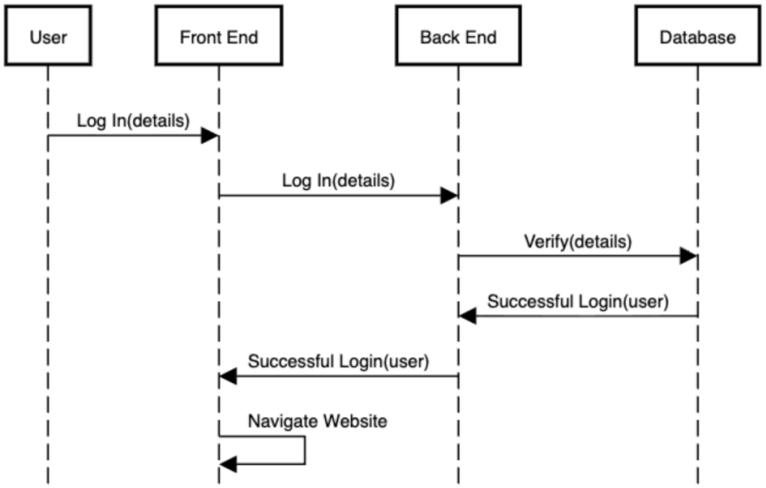
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	**id**	int(11)			No	*None*		AUTO_INCREMENT	Change Drop More
2	**asset_id**	int(11)			No	*None*			Change Drop More
3	**changed_by**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
4	**action**	varchar(50)	utf8mb4_general_ci		No	*None*			Change Drop More
5	**field_changed**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
6	**old_value**	text	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
7	**new_value**	text	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
8	**change_date**	datetime			Yes	current_timestamp()			Change Drop More

Component Diagram

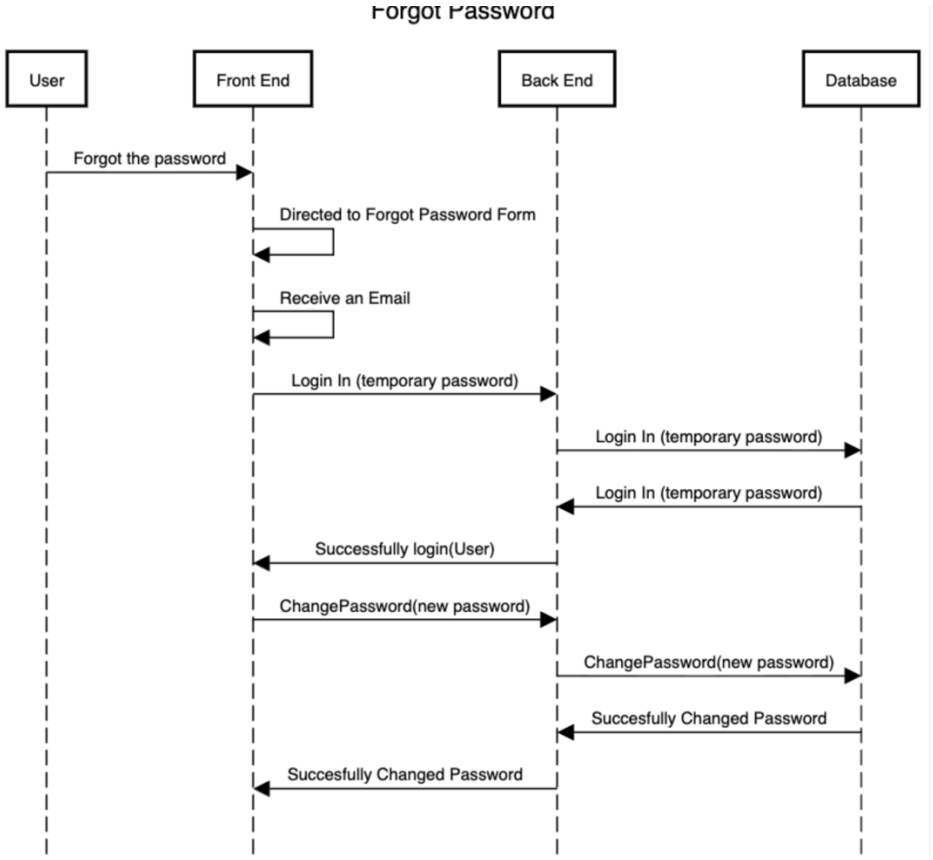


Sequence Diagrams

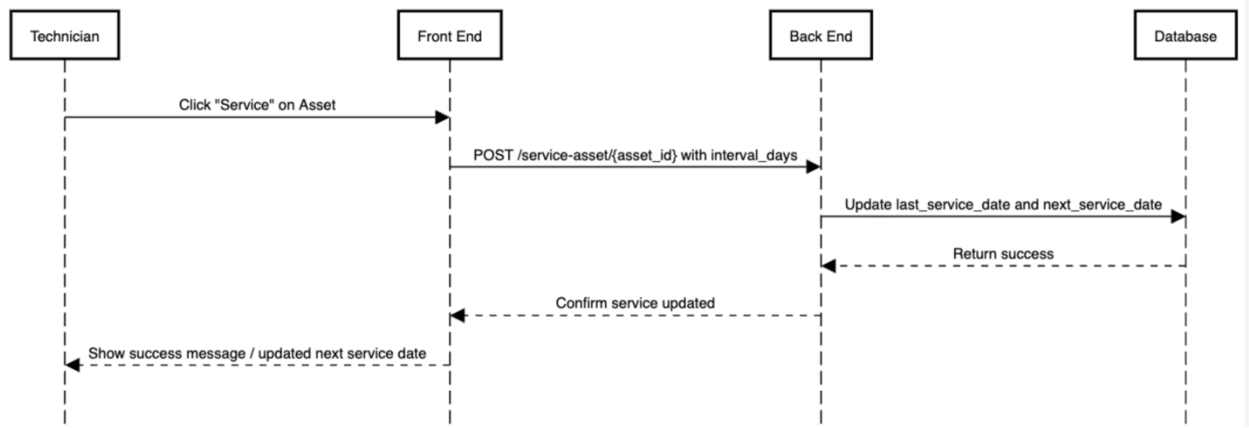
Login



Forgot Password



Servicing an Asset



Creating an Alert when a Service is due

References

Figures:

Fig 1. SETU LOGO: https://waterfordlibraries.ie/europe-direct-waterfords-repair-and-recycle-workshop/setu-logo-002/

Fig 2. Maintenex Logo.

Fig 3. Possible Dashboard Design: https://www.slideteam.net/media/catalog/product/cache/1280x720/m/a/maintenance_dashboard_with_schedule_compliance_slide01.jpg