Internship project report

Belguebli Rayane

Table of Contents

Abstract

Our project aims to create an innovative web platform that acts as a management centre for maintenance tasks. Data from a CMMS relating to tasks to be carried out, planned by managers, is retrieved and presented interactively to technicians via glasses equipped with AR technology. This information, integrated directly into their field of vision, gives technicians rapid access to instructions and task details, improving their efficiency and accuracy when working in the field. By using Django based on the MVT model, we are ensuring a robust and scalable architecture to support this complex integration between maintenance data and AR technology.

Chapter 1

Introduction

In the field of industrial maintenance, the efficient management of tasks and resources is crucial to ensuring the productivity and reliability of operations (Malta, Mendes, and Farinha 2021) [1]. Our project addresses this issue by proposing an innovative solution that exploits emerging technologies to facilitate the maintenance process. Inspired by recent advances on ICMMS (Fu et al. 2002) [2], we aim to design an integrated Django-based web platform to orchestrate coordination between managers and technicians, while integrating Augmented Reality (AR) features for an immersive experience.

This solution meets a growing need in the field of industrial maintenance, where the complexity of equipment and operations requires agile and efficient management. By combining the power of asset management with the ease of use of an intuitive web interface, our solution aims to streamline maintenance processes while providing an optimal user experience.

Our aim is to provide a versatile and adaptable platform that can be tailored to the specific needs of different industries and businesses. Through the integration of AR, technicians will be able to access contextual information in real time, improving their efficiency and accuracy when working in the field.

This project represents an exciting opportunity for an internship abroad, offering the chance to explore new technologies and contribute to innovative solutions in the field of industrial maintenance. By taking on this challenge, we aim to acquire new skills.

Chapter 2

What is CMMS ?

A Computerised Maintenance Management System (CMMS) is an essential tool for the efficient management of maintenance activities within an organisation. Whether in industry, services or public institutions, a CMMS provides a centralised platform where maintenance teams can effectively plan, execute and monitor their tasks.

One of the key benefits of a CMMS is its ability to integrate new technologies such as mobility and traceability applications. Using mobile devices such as smartphones or tablets, field technicians can access maintenance information in real time, enter data on the spot and receive instant instructions. This significantly improves the responsiveness and efficiency of the maintenance team, reducing unplanned downtime.

In addition, traceability of maintenance activities is a crucial element in ensuring regulatory compliance and optimising processes. Modern CMMSs offer advanced monitoring and reporting capabilities, enabling managers to track maintenance histories, analyse trends and make informed decisions to improve overall asset performance.

Studies and articles have been published to demonstrate the effectiveness of CMMS in different sectors. For example, research carried out by experts in maintenance management revealed that the implementation of a CMMS in an industrial company led to a significant reduction in downtime and maintenance costs, while improving equipment availability (Shankar, Singh, and Singh 2021) [3].

Similarly, another case study that highlighted the benefits of a CMMS in the utilities sector, showing how a municipality was able to optimise its public infrastructure maintenance operations through the use of an integrated CMMS solution (Kour et al. 2022) [4].

In summary, CMMS are essential tools for maintenance teams, providing efficient management of maintenance activities while taking advantage of new technologies to improve responsiveness and traceability. Studies and research articles support the positive impact of CMMS in different sectors, highlighting their value as a key element of asset management and preventive maintenance.

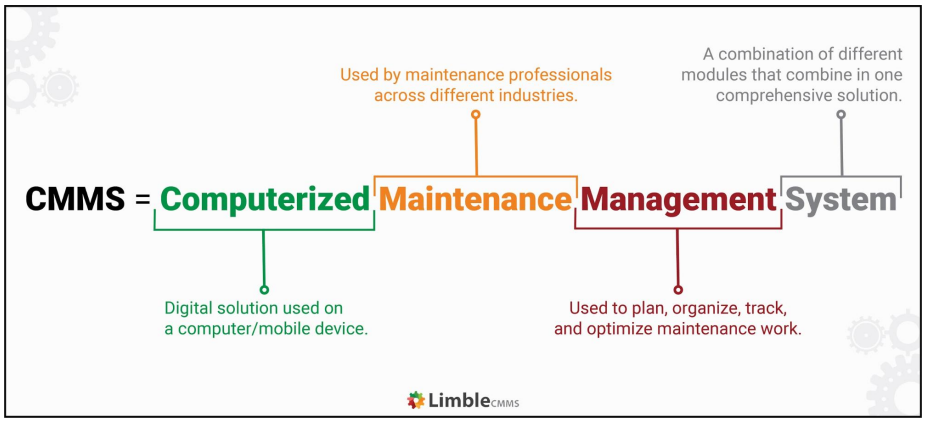

Figure 1: CMMS source(04/2024) : https://limblecmms.com/cmms/

Examples of CMMS

- **MaintainX** :

Figure 2: MAINTAINX source(04/2024) : https://www.prnewswire.com/news-releases/maintainx-raises-50m-with-12x-revenue-growth-from-boosting-productivity-of-industrial-and-frontline-workers-as-america-gets-back-to-work-301308326.html

MaintainX is a CMMS based on a mobile application and a web platform that aims to simplify the maintenance process for teams in the field.

It offers functionalities such as work order creation, asset management, spare parts inventory tracking, preventive maintenance task planning and report generation.

The mobile application allows technicians to receive real-time notifications, update the status of tasks and upload photos or notes to document interventions.

The web platform provides managers with a centralised dashboard where they can track the progress of jobs, analyse performance data and generate reports to optimise maintenance processes. source(04/2024)

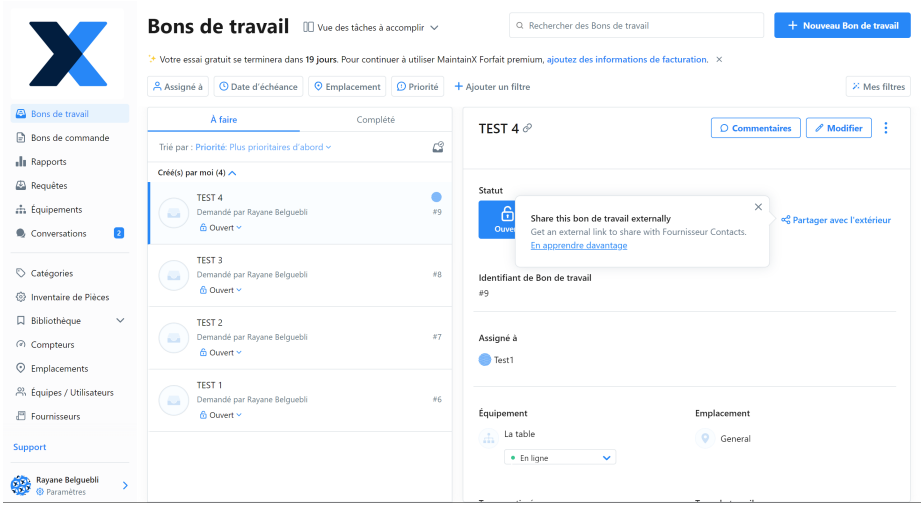

Figure 3: workorders

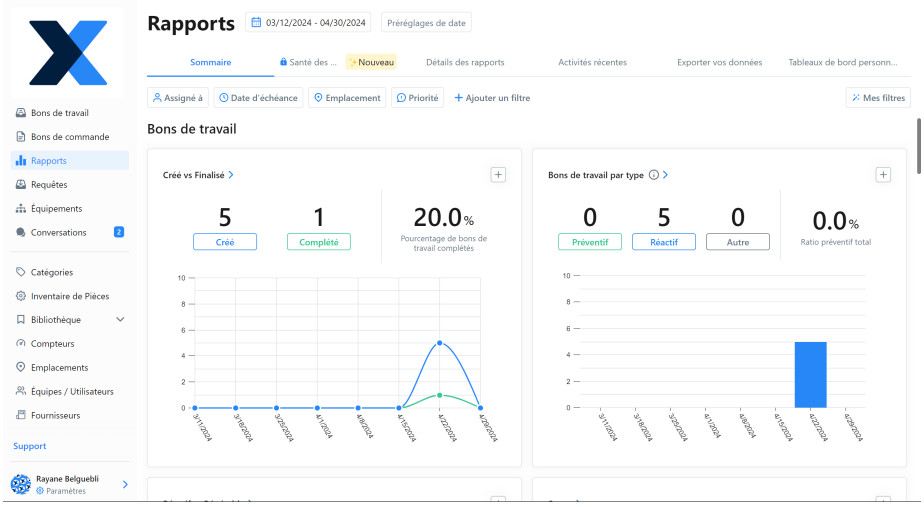

Figure 4: reports

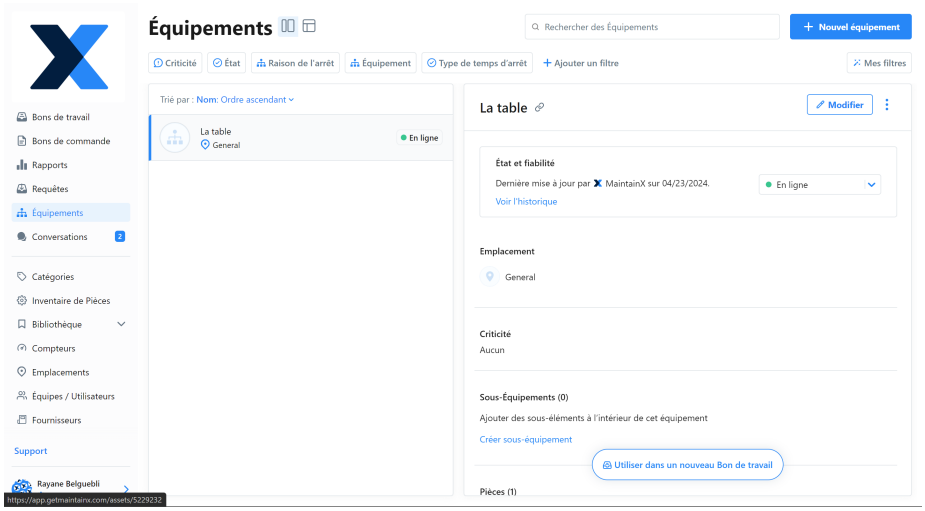

Figure 5: assets

- **Limble** :

Figure 6: LIMBLE source(04/2024) : https://www.prnewswire.com/news-releases/limble-announces-58m-series-b-funding-round-led-by-goldman-sachs-asset-management-bringing-total-valuation-to-450m-301857646.html

Limble is also a modern CMMS designed to simplify the management of maintenance activities and extend the life of assets.

It offers similar functionality to MaintainX, such as work order creation, scheduling, spares management and reporting.

Limble is distinguished by its user-friendly interface and advanced asset management features, such as equipment hierarchy modelling, warranty management and replacement planning.

It also incorporates predictive maintenance and performance dashboard capabilities to help users anticipate breakdowns and make informed decisions to optimise maintenance. source(04/2024)

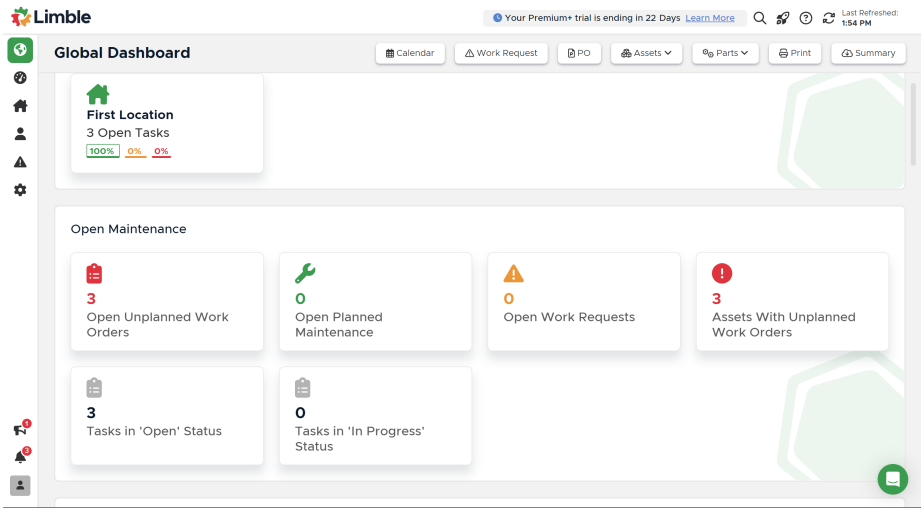

Figure 7: dashboard

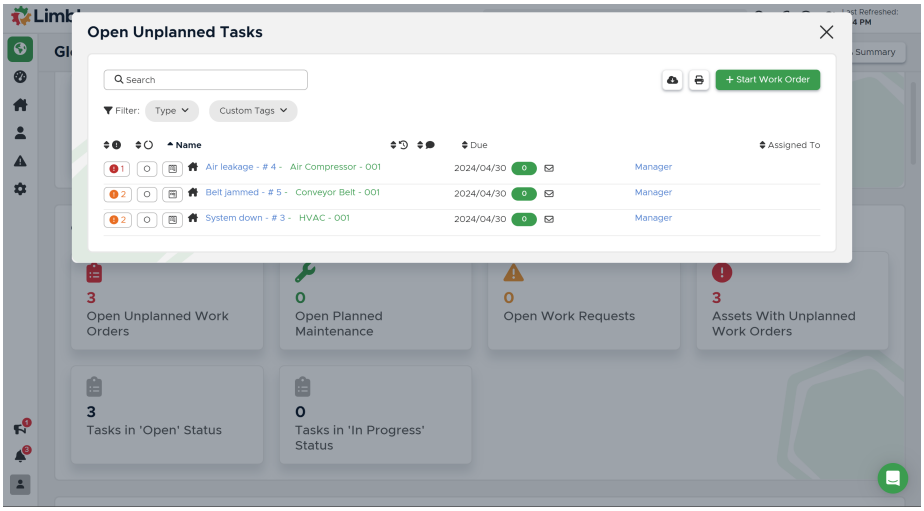

Figure 8: Tasks

- **Odoo Maintenance**

Figure 9: ODOO source(04/2024 : https://ubi.edu/odoo/)

Odoo is a suite of open source applications covering all your business needs: CRM, eCommerce, Accounting, Inventory, Point of Sale, Project Management, etc.

Odoo has a dedicated maintenance section that allows you to manage work orders and assets. Odoo Maintenance is perhaps less advanced than the previous CMMS but is sufficient to solve our problem and above all is free to download. source(04/2024)

Odoo is based on a 3-tier architecture:

- a PostgreSQL database server, which can contain several databases ;
- an application server containing the management objects, workflow engine, editing generator, etc. ;
- a presentation server that allows users to connect to OpenERP using any Web browser (with the Flash player installed for displaying graphics). This last server is not necessary if the user uses the native client, which does require installation on the user's workstation.

The server part is written in Python. The various building blocks are organised into modules. A module is a folder with a predefined structure containing Python code and XML files. A module defines the data structure, forms, reports, menus, procedures, workflow, etc. source(04/2024)

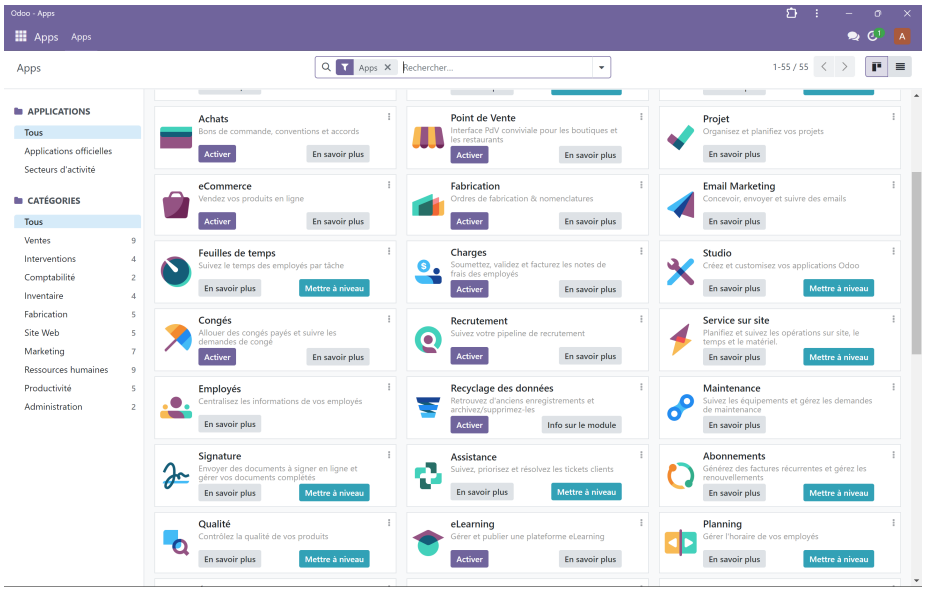

Figure 10: apps

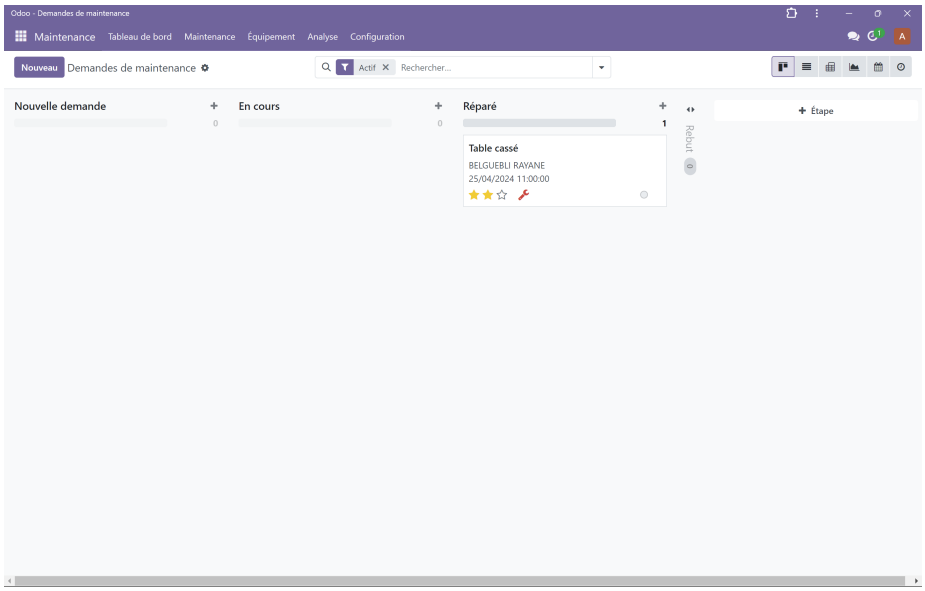

Figure 11: maintenace

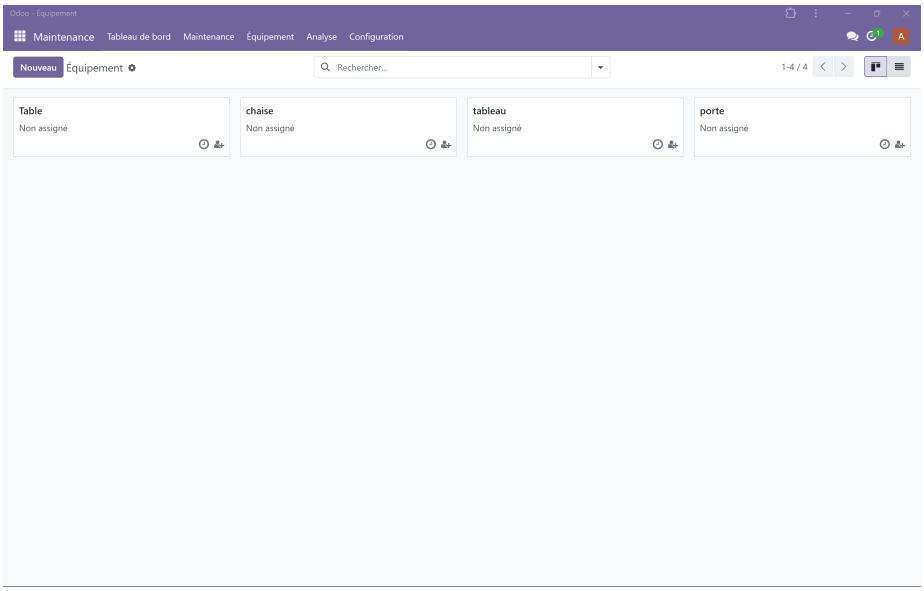

Figure 12: assets

CMMS comparison

	MaintainX	Limble	Odoo
Main features	work orders, digital checklists and real-time notifications.	work order, preventive maintenance, asset management and analytical dashboards.	work order management, preventive maintenance planning, asset tracking, spare parts stock management and report generation.
User interface	user-friendly mobile interface for fast, efficient communications.	intuitive interface, with simple navigation and clear menus.	user interface is renowned, ease of use and customisation.
Prices	paid enterprise version and online community version.	paid enterprise version and online community version.	paid enterprise version and open source community version.

	MaintainX	Limble	Odoo
Integration and customisation	MaintainX offers integrations with messaging tools and communication platforms such as Slack, as well as the ability to customise checklists and forms.	asset management systems, accounting software and other productivity tools.	other Odoo applications as well as with third-party software via additional modules.

What are Maintenance Management Systems?

Maintenance is a critical aspect of any manufacturing or industrial process. It ensures that equipment, machines and facilities operate at their optimum level, avoiding breakdowns and costly downtime. To achieve this, companies need to put in place a well-defined Maintenance Management System (MMS). An MMS is a holistic approach to maintenance management that involves the integration of people, processes and technology to optimise maintenance activities.

The main objective of an MMS is to improve equipment reliability, minimise downtime and increase productivity. It does this by providing a structured approach to maintenance planning, scheduling and execution. This helps maintenance teams to proactively identify and resolve potential problems before they become major issues. By leveraging data, analytics and technology, an MMS enables organisations to optimise maintenance processes, reduce costs and increase asset life. Here are some key points highlighting their importance and benefits :

- **Optimising maintenance operations** : MMS provides a centralised platform for planning, executing and monitoring maintenance activities. This enables efficient resource allocation, priority management and coordination of maintenance teams.
- **Reduced unplanned downtime** : By enabling preventive and predictive maintenance, MMS helps to identify and resolve problems before they become major breakdowns. This reduces unplanned downtime and minimises production interruptions.
- **Extended equipment life** : By providing regular, preventive maintenance, MMS helps to extend the life of equipment. By identifying and correcting potential problems at an early stage, MMS reduces wear and tear on assets.
- **Maximising asset availability** : By ensuring that equipment is well maintained and available when needed, MMS maximises asset availability. This enables organisations to maintain productivity and respond effectively to market demand.

In summary, MMS plays a crucial role in optimising maintenance operations, reducing downtime, extending equipment life and maximising asset availability. The above references illustrate the positive impact of MMS in various sectors and highlight their value as strategic tools for asset management and operational performance (Sorić 2024) [5].

What is the django framework et and why use this for our web application ?

Figure 13: DJANGO source(04/2024) : https://www.djangoproject.com/

In our project to create an innovative web platform that acts as a management centre for maintenance tasks, Django proved to be a wise choice for several key reasons :

- **MVC/MVT structure adapted to our architecture** : Django follows the MVT model, which is a variant of the MVC model. This architecture is particularly well-suited to our project because it allows us to clearly separate the different responsibilities of our application.
  - The Model represents data from the CMMS, such as maintenance task details, equipment information, schedules, etc.
  - The Template is responsible for presenting the data to technicians. In our case, this means generating content adapted to augmented reality for display on their glasses.
  - The View manages the business logic of our application, including the integration of data from the CMMS with augmented reality functionalities to provide technicians with precise, interactive instructions.
- **Managing complex data with Django's ORM** : Django's ORM (Object-Relational Mapping) is one of its most powerful and popular features. It greatly simplifies data manipulation by allowing developers to interact with the database using Python objects rather than direct SQL queries. Here are some key aspects of Django's ORM and why it's beneficial for our project:
  - **Database abstraction** : Django's ORM provides a high-level abstraction of the database, meaning developers don't need to worry

about the specific details of the underlying database (such as the type of database used or the SQL language).

This allows Django applications to be developed in a more portable way, as the code can be easily adapted to different types of database without having to change the business logic.

- **Using Python objects to represent data** : Database tables are represented by Python classes called "models" in Django. Each model defines the table's fields and relationships with other models.

  For example, in our project, we might have a Task model to represent the various maintenance tasks, with fields such as description, start date, end date, status, etc.

- **Easy data manipulation** : Once models are defined, CRUD (Create, Read, Update, Delete) operations can be performed using simple and intuitive methods on model objects.

  For example, to create a new task in our project, we would simply create an instance of the Task class, assign it the appropriate values for its fields, and then call the save() method to save it in the database.

- **Transparent management of relationships between tables** : Django's ORM makes it easy to define relationships between models, such as foreign keys and many-to-many relationships.

  For example, in our project, a task could be linked to a specific piece of equipment. This relationship can be easily defined in the Task model by adding a foreign key field that points to the Equipment model.

- **Protection against SQL injections** : Using Django's ORM provides built-in protection against SQL injections, as queries are securely generated using query parameters, preventing SQL injection attacks.

- **Flexible presentation of AR data** : Django has a flexible template system that allows dynamic content to be generated for a variety of devices, including augmented reality glasses.

  Templates can be customised to deliver instructions interactively, displaying details of maintenance tasks directly in the technicians' field of vision, improving their productivity and accuracy.

- **Security and scalability**: Security is a major concern, especially when it comes to handling sensitive data such as that associated with industrial maintenance. Django incorporates robust security features, such as protection against SQL injections and CSRF attacks, to ensure the security of user data.

  In addition, Django is known for its ability to manage large-scale web applications efficiently. Its scalability will enable our platform to grow

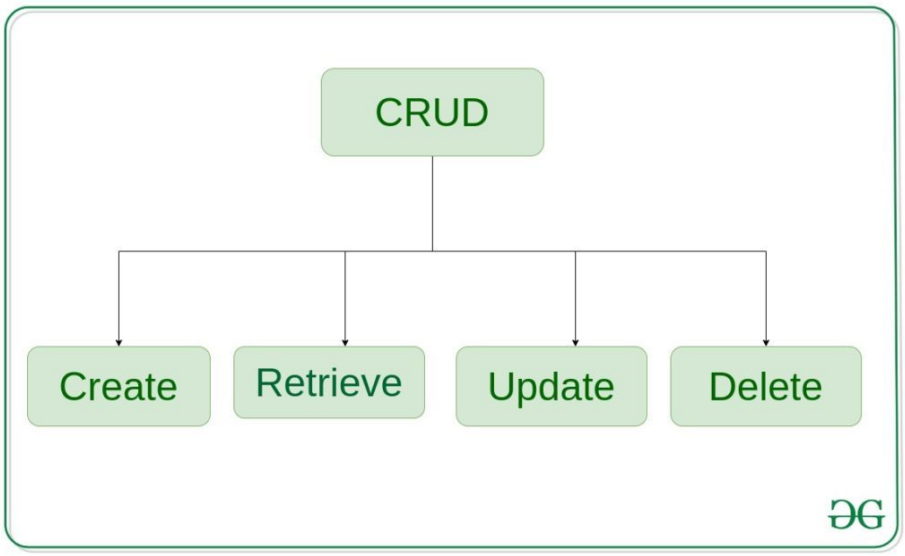

Figure 14: CRUD source(04/2024) : https://www.geeksforgeeks.org/django-crud-create-retrieve-update-delete-function-based-views/

with the addition of new features and larger volumes of data.

In short, Django offers a combination of powerful features, built-in security and flexibility that make it the ideal choice for our maintenance task management platform project incorporating augmented reality. Its MVC/MVT architecture, robust ORM and flexible templating system ensure that our solution can be implemented efficiently and scalably.

Chapter 3

Requirements

Use Case

Figure 15: Use case

Description of uses cases Scenarios for Admin

Scenario 1: Create a Manager - **Actor**: Admin - **Actions**: 1. The Admin logs into the system. 2. The Admin accesses the Managers management section. 3. The Admin creates a new Manager by filling out the form. 4. The Admin submits the form. - **Expected Result**: A new Manager is created and added to the list of Managers. - **Error Case**: If the Manager already exists (checked by email or unique identifier), display an error message: "Manager already exists."

Scenario 2: Modify a Manager - **Actor**: Admin - **Actions**: 1. The Admin logs into the system. 2. The Admin accesses the Managers management section. 3. The Admin selects the Manager to modify. 4. The Admin modifies the Manager's information in the form. 5. The Admin submits the modifications. - **Expected Result**: The Manager's information is updated. - **Error Case**: If the entered information already exists for another Manager, display an error message: "Information already exists for another Manager."

Scenario 3: Delete a Manager - **Actor**: Admin - **Actions**: 1. The Admin logs into the system. 2. The Admin accesses the Managers management section. 3. The Admin selects the Manager to delete. 4. The Admin confirms the deletion. - **Expected Result**: The Manager is deleted from the system. - **Error Case**: If the Manager is associated with workorders, display an error message: "Cannot delete Manager as associated with workorders."

Scenario 4: View the list of Managers - **Actor**: Admin - **Actions**: 1. The Admin logs into the system. 2. The Admin accesses the Managers management section. 3. The Admin views the list of Managers. - **Expected Result**: The Admin sees a list of all Managers with their information.

Scenario 5: Modify their profile - **Actor**: Admin - **Actions**: 1. The Admin logs into the system. 2. The Admin accesses "My Profile". 3. The Admin modifies their personal information. 4. The Admin submits the modifications. - **Expected Result**: The Admin's profile is updated. - **Error Case**: If the new information entered already exists for another user, display an error message: "Information already exists for another user."

Scenarios for Manager

Scenario 1: Create a Team - **Actor**: Manager - **Actions**: 1. The Manager logs into the system. 2. The Manager accesses the team management section. 3. The Manager creates a new team by specifying a name. 4. The Manager submits the team creation. - **Expected Result**: A new team is created and added to the Manager's team list. - **Error Case**: If the team already exists, display an error message: "Team already exists."

Scenario 2: Create a Technician - **Actor**: Manager - **Actions**: 1. The Manager logs into the system. 2. The Manager accesses the Technicians management section. 3. The Manager creates a new Technician by filling out the form. 4. The Manager submits the form. - **Expected Result**: A new Technician is created and added to the list of Technicians. - **Error Case**: If

the Technician already exists (checked by email or unique identifier), display an error message: “Technician already exists.”

Scenario 3: Modify a Technician - **Actor**: Manager - **Actions**: 1. The Manager logs into the system. 2. The Manager accesses the Technicians management section. 3. The Manager selects the Technician to modify. 4. The Manager modifies the Technician’s information in the form. 5. The Manager submits the modifications. - **Expected Result**: The Technician’s information is updated. - **Error Case**: If the entered information already exists for another Technician, display an error message: “Information already exists for another Technician.”

Scenario 4: Delete a Technician - **Actor**: Manager - **Actions**: 1. The Manager logs into the system. 2. The Manager accesses the Technicians management section. 3. The Manager selects the Technician to delete. 4. The Manager confirms the deletion. - **Expected Result**: The Technician is deleted from the system. - **Error Case**: If the Technician is associated with workorders, display an error message: “Cannot delete Technician as associated with workorders.”

Scenario 5: Add Technicians to their team - **Actor**: Manager - **Actions**: 1. The Manager logs into the system. 2. The Manager accesses the team management section. 3. The Manager selects existing Technicians to add to their team. 4. The Manager confirms the addition. - **Expected Result**: The selected Technicians are added to the Manager’s team. - **Error Case**: If the team already exists, display an error message: “Team already exists.”

Scenario 6: View workorders associated with their team - **Actor**: Manager - **Actions**: 1. The Manager logs into the system. 2. The Manager accesses the workorders management section. 3. The Manager views the list of workorders associated with their team. - **Expected Result**: The Manager sees the workorders assigned to their team.

Scenario 7: Modify their profile - **Actor**: Manager - **Actions**: 1. The Manager logs into the system. 2. The Manager accesses “My Profile”. 3. The Manager modifies their personal information. 4. The Manager submits the modifications. - **Expected Result**: The Manager’s profile is updated. - **Error Case**: If the new information entered (such as email) already exists for another user, display an error message: “Information already exists for another user.”

Scenarios for Technician

Scenario 1: Modify their profile - **Actor**: Technician - **Actions**: 1. The Technician logs into the system. 2. The Technician accesses “My Profile”. 3. The Technician modifies their personal information. 4. The Technician submits the modifications. - **Expected Result**: The Technician’s profile is updated. - **Error Case**: If the new information entered (such as email) already exists for another user, display an error message: “Information already exists for another user.”

Scenario 2: View assigned workorders - **Actor**: Technician - **Actions**: 1. The Technician logs into the system. 2. The Technician accesses the workorders management section. 3. The Technician views the list of workorders assigned to them. - **Expected Result**: The Technician sees the workorders assigned to them.

Scenario 3: Select workorders - **Actor**: Technician - **Actions**: 1. The Technician logs into the system. 2. The Technician accesses the workorders management section. 3. The Technician selects one or more workorders to process. 4. The Technician confirms their selection. - **Expected Result**: The workorders are marked as selected by the Technician.

Scenario 4: Modify the state of workorders - **Actor**: Technician - **Actions**: 1. The Technician logs into the system. 2. The Technician accesses the workorders management section. 3. The Technician selects a workorder. 4. The Technician modifies the state of the workorder (e.g., in progress, completed). 5. The Technician submits the modifications. - **Expected Result**: The state of the workorder is updated.

Scenario 5: Add notes to a workorder - **Actor**: Technician - **Actions**: 1. The Technician logs into the system. 2. The Technician accesses the workorder details. 3. The Technician adds notes to the workorder. 4. The Technician saves the notes. - **Expected Result**: Notes are successfully added to the workorder.

Non-Functional Requirements

- Performance: The backend system should be highly performant and scalable to handle a large number of concurrent users and data requests. Optimize database queries and data processing to minimize response times and maintain overall application responsiveness.
- Security: Implement robust security measures to protect sensitive data, including encryption of data at rest and in transit.
- Maintainability: Employ modular and well-documented code to facilitate easy understanding, modification, and maintenance of the backend system.

External API Odoo

Connection

```
url = <insert server URL>
db = <insert database name>
username = 'admin'
password = <insert password for your admin user (default: admin)>
```

GET

```
models.execute_kw(db, uid, password, 'res.users', 'search_read', [[]], {'fields': []})
```

PATCH

```
models.execute_kw(
        db, uid, password,
        'maintenance.request', 'write',
        [[33], {'schedule_date': formatted_now}]
    )
```

CREATE

```
new_user_data = {
    'name': "signup_username",
    'login': "signup_email",
    'password': "signup_password",
}
models.execute_kw(db, uid, password, 'res.users', 'create', [new_user_data])
```

DELETE

```
models.execute_kw(db, uid, password, 'res.users', 'unlink', [[2]])
```

source (05/2024)

Database

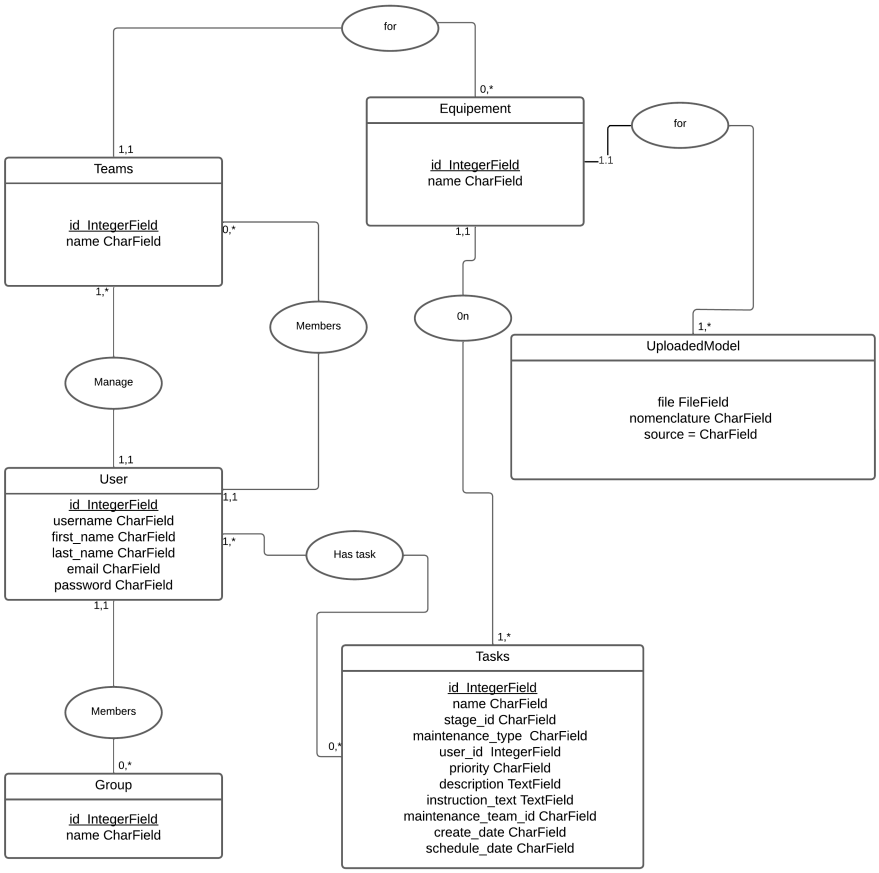

Figure 16: Database

Sprints

Sprint 1 Admin

Test Report: Create a Manager

Test Case 1: Successfully Create a New Manager

Scenario	Create a new Manager
Preconditions	- The Admin is registered in the system. - The Admin has the necessary rights to create a Manager.
Steps	1. The Admin logs into the system. **Input Data**: Admin username and password. 2. The Admin accesses the Managers management section. **Input Data**: None (simple access to the section). 3. The Admin fills out the form to create a new Manager. **Input Data**: First name, last name, email, unique identifier, password. 4. The Admin submits the form. **Input Data**: None (simple submission action).
Expected Result	- A new Manager is created and added to the list of Managers. - The system displays a confirmation message indicating the Manager has been created successfully. - The new Manager appears in the list of Managers with the provided information.

Test Case 2: Attempt to Create an Existing Manager

Scenario	Create a Manager with an existing email or unique identifier
Preconditions	- The Admin is registered in the system. - The Admin has the necessary rights to create a Manager. - A Manager with the same email or unique identifier already exists in the system.
Steps	1. The Admin logs into the system. **Input Data**: Admin username and password. 2. The Admin accesses the Managers management section. **Input Data**: None (simple access to the section). 3. The Admin fills out the form to create a new Manager with an existing email or unique identifier. **Input Data**: First name, last name, email (existing), unique identifier (existing), password. 4. The Admin submits the form. **Input Data**: None (simple submission action).

Scenario	Create a Manager with an existing email or unique identifier
Expected Result	- The system displays an error message: “Manager already exists.” - No new entry is created in the list of Managers.

Test Report: View the List of Managers

Test Case 1: Successfully View the List of Managers

Scenario	View the list of Managers
Preconditions	- The Admin is registered in the system. - The Admin has the necessary rights to view the list of Managers.
Steps	1. The Admin logs into the system. **Input Data**: Admin username and password. 2. The Admin accesses the Managers management section. **Input Data**: None (simple access to the section). 3. The Admin views the list of Managers. **Input Data**: None (simple action of viewing the list).
Expected Result	- The Admin sees a list of all Managers with their information, including first name, last name, email, and unique identifier.

Manager

Test Report: Create a Team

Test Case 1: Successfully Create a New Team

Scenario	Create a new Team
Preconditions	- The Manager is registered in the system. - The Manager has the necessary rights to create a team.
Steps	1. The Manager logs into the system. **Input Data**: Manager username and password. 2. The Manager accesses the team management section. **Input Data**: None (simple access to the section). 3. The Manager fills out the form to create a new team by specifying a name. **Input Data**: Team name. 4. The Manager submits the team creation. **Input Data**: None (simple submission action).
Expected Result	- A new team is created and added to the Manager's team list. - The system displays a confirmation message indicating the team has been created successfully. - The new team appears in the Manager's team list with the specified name.

Test Case 2: Attempt to Create an Existing Team

Scenario	Create a Team with an existing name
Preconditions	- The Manager is registered in the system. - The Manager has the necessary rights to create a team. - A team with the same name already exists under the Manager's teams.
Steps	1. The Manager logs into the system. **Input Data**: Manager username and password. 2. The Manager accesses the team management section. **Input Data**: None (simple access to the section). 3. The Manager fills out the form to create a new team with an existing name. **Input Data**: Team name (existing). 4. The Manager submits the team creation. **Input Data**: None (simple submission action).
Expected Result	- The system displays an error message: "Team already exists." - No new entry is created in the Manager's team list.

Test Report: Create a Technician

Test Case 1: Successfully Create a New Technician

Scenario	Create a new Technician
Preconditions	- The Manager is registered in the system. - The Manager has the necessary rights to create a Technician.
Steps	1. The Manager logs into the system. **Input Data**: Manager username and password. 2. The Manager accesses the Technicians management section. **Input Data**: None (simple access to the section). 3. The Manager fills out the form to create a new Technician. **Input Data**: First name, last name, email, unique identifier, password. 4. The Manager submits the form. **Input Data**: None (simple submission action).
Expected Result	- A new Technician is created and added to the list of Technicians. - The system displays a confirmation message indicating the Technician has been created successfully. - The new Technician appears in the list of Technicians with the provided information.

Test Case 2: Attempt to Create an Existing Technician

Scenario	Create a Technician with an existing email or unique identifier
Preconditions	- The Manager is registered in the system. - The Manager has the necessary rights to create a Technician. - A Technician with the same email or unique identifier already exists in the system.
Steps	1. The Manager logs into the system. **Input Data**: Manager username and password. 2. The Manager accesses the Technicians management section. **Input Data**: None (simple access to the section). 3. The Manager fills out the form to create a new Technician with an existing email or unique identifier. **Input Data**: First name, last name, email (existing), unique identifier (existing), password. 4. The Manager submits the form. **Input Data**: None (simple submission action).
Expected Result	- The system displays an error message: “Technician already exists.” - No new entry is created in the list of Technicians.

Test Report: View Workorders Associated with Their Team

Test Case 1: Successfully View Workorders Associated with the Manager's Team

Scenario	View workorders associated with their team
Preconditions	- The Manager is registered in the system. - The Manager has the necessary rights to view workorders. - There are workorders assigned to the Manager's team.
Steps	1. The Manager logs into the system. **Input Data**: Manager username and password. 2. The Manager accesses the workorders management section. **Input Data**: None (simple access to the section). 3. The Manager views the list of workorders associated with their team. **Input Data**: None (simple action of viewing the list).
Expected Result	- The Manager sees the workorders assigned to their team. - The list displays workorder details such as workorder ID, description, status, and assigned technician.

Technician

Test Report: View Assigned Workorders

Test Case 1: Successfully View Workorders Assigned to the Technician

Scenario	View assigned workorders
Preconditions	- The Technician is registered in the system. - The Technician has the necessary rights to view workorders. - There are workorders assigned to the Technician.
Steps	1. The Technician logs into the system. **Input Data**: Technician username and password. 2. The Technician accesses the workorders management section. **Input Data**: None (simple access to the section). 3. The Technician views the list of workorders assigned to them. **Input Data**: None (simple action of viewing the list).
Expected Result	- The Technician sees the workorders assigned to them. - The list displays workorder details such as workorder ID, description, status, and priority.

Test Report: Select Workorders

Test Case 1: Successfully Select Workorders to Process

Scenario	Select workorders to process
Preconditions	- The Technician is registered in the system. - The Technician has the necessary rights to view and select workorders. - There are workorders available for the Technician to select.
Steps	1. The Technician logs into the system. **Input Data**: Technician username and password. 2. The Technician accesses the workorders management section. **Input Data**: None (simple access to the section). 3. The Technician selects one or more workorders to process. **Input Data**: Workorder IDs to be selected. 4. The Technician confirms their selection. **Input Data**: None (simple confirmation action).
Expected Result	- The selected workorders are marked as selected by the Technician. - The system updates the status of the selected workorders to reflect that they are being processed by the Technician. - A confirmation message is displayed to the Technician.

Test Report: Modify the State of Workorders

Test Case 1: Successfully Modify the State of a Workorder

Scenario	Modify the state of a workorder
Preconditions	- The Technician is registered in the system. - The Technician has the necessary rights to view and modify workorders. - There are workorders available for the Technician to modify.
Steps	1. The Technician logs into the system. **Input Data**: Technician username and password. 2. The Technician accesses the workorders management section. **Input Data**: None (simple access to the section). 3. The Technician selects a workorder. **Input Data**: Workorder ID to be modified. 4. The Technician modifies the state of the workorder (e.g., in progress, completed). **Input Data**: New state of the workorder. 5. The Technician submits the modifications. **Input Data**: None (simple submission action).
Expected Result	- The state of the workorder is updated. - The system displays a confirmation message indicating the state has been updated successfully. - The workorder's new state is reflected in the workorders list.

Sprint 2 Admin

Test Report: Modify a Manager

Test Case 1: Successfully Modify a Manager's Information

Scenario	Modify a Manager's information
Preconditions	- The Admin is registered in the system. - The Admin has the necessary rights to modify Managers. - There is at least one Manager available for modification.
Steps	1. The Admin logs into the system. **Input Data**: Admin username and password. 2. The Admin accesses the Managers management section. **Input Data**: None (simple access to the section). 3. The Admin selects the Manager to modify. **Input Data**: Manager ID or name to be modified. 4. The Admin modifies the Manager's information in the form. **Input Data**: Updated information for the Manager (e.g., name, email, unique identifier). 5. The Admin submits the modifications. **Input Data**: None (simple submission action).
Expected Result	- The Manager's information is updated with the modifications made by the Admin. - The system displays a confirmation message indicating that the modifications have been saved successfully.

Test Case 2: Error Handling - Information Already Exists for Another Manager

Scenario	Modify a Manager with existing information
Preconditions	- The Admin is registered in the system. - The Admin has the necessary rights to modify Managers. - There is at least one other Manager with the same information (e.g., email, unique identifier).

Scenario	Modify a Manager with existing information
Steps	1. The Admin logs into the system. **Input Data**: Admin username and password. 2. The Admin accesses the Managers management section. **Input Data**: None (simple access to the section). 3. The Admin selects the Manager to modify. **Input Data**: Manager ID or name to be modified. 4. The Admin modifies the Manager's information in the form with information that already exists for another Manager. **Input Data**: Updated information for the Manager (e.g., email, unique identifier). 5. The Admin submits the modifications. **Input Data**: None (simple submission action).
Expected Result	- The system displays an error message: "Information already exists for another Manager." - The modifications are not saved.

Test Report: Delete a Manager

Test Case 1: Successfully Delete a Manager

Scenario	Delete a Manager
Preconditions	- The Admin is registered in the system. - The Admin has the necessary rights to delete Managers. - There is at least one Manager available for deletion.
Steps	1. The Admin logs into the system. **Input Data**: Admin username and password. 2. The Admin accesses the Managers management section. **Input Data**: None (simple access to the section). 3. The Admin selects the Manager to delete. **Input Data**: Manager ID or name to be deleted. 4. The Admin confirms the deletion. **Input Data**: Confirmation action (e.g., clicking "Delete" button).
Expected Result	- The selected Manager is deleted from the system. - The system displays a confirmation message indicating that the Manager has been successfully deleted.

Test Case 2: Error Handling - Manager Associated with Workorders

Scenario	Attempt to Delete a Manager Associated with Workorders
Preconditions	- The Admin is registered in the system. - The Admin has the necessary rights to delete Managers. - The Manager selected for deletion is associated with one or more workorders.
Steps	1. The Admin logs into the system. **Input Data**: Admin username and password. 2. The Admin accesses the Managers management section. **Input Data**: None (simple access to the section). 3. The Admin selects the Manager associated with workorders to delete. **Input Data**: Manager ID or name to be deleted. 4. The Admin confirms the deletion. **Input Data**: Confirmation action (e.g., clicking "Delete" button).
Expected Result	- The system displays an error message: "Cannot delete Manager as associated with workorders." - The Manager is not deleted from the system.

Manager

Test Report: Modify a Technician

Test Case 1: Successfully Modify a Technician's Information

Scenario	Modify a Technician's information
Preconditions	- The Manager is registered in the system. - The Manager has the necessary rights to modify Technicians. - There is at least one Technician available for modification.
Steps	1. The Manager logs into the system. **Input Data**: Manager username and password. 2. The Manager accesses the Technicians management section. **Input Data**: None (simple access to the section). 3. The Manager selects the Technician to modify. **Input Data**: Technician ID or name to be modified. 4. The Manager modifies the Technician's information in the form. **Input Data**: Updated information for the Technician (e.g., name, email, unique identifier). 5. The Manager submits the modifications. **Input Data**: None (simple submission action).
Expected Result	- The Technician's information is updated with the modifications made by the Manager. - The system displays a confirmation message indicating that the modifications have been saved successfully.

Test Case 2: Error Handling - Information Already Exists for Another Technician

Scenario	Modify a Technician with existing information
Preconditions	- The Manager is registered in the system. - The Manager has the necessary rights to modify Technicians. - There is at least one other Technician with the same information (e.g., email, unique identifier).

Scenario	Modify a Technician with existing information
Steps	1. The Manager logs into the system. **Input Data**: Manager username and password. 2. The Manager accesses the Technicians management section. **Input Data**: None (simple access to the section). 3. The Manager selects the Technician to modify. **Input Data**: Technician ID or name to be modified. 4. The Manager modifies the Technician's information in the form with information that already exists for another Technician. **Input Data**: Updated information for the Technician (e.g., email, unique identifier). 5. The Manager submits the modifications. **Input Data**: None (simple submission action).
Expected Result	- The system displays an error message: "Information already exists for another Technician." - The modifications are not saved.

Test Report: Delete a Technician

Test Case 1: Successfully Delete a Technician

Scenario	Delete a Technician
Preconditions	- The Manager is registered in the system. - The Manager has the necessary rights to delete Technicians. - There is at least one Technician available for deletion.
Steps	1. The Manager logs into the system. **Input Data**: Manager username and password. 2. The Manager accesses the Technicians management section. **Input Data**: None (simple access to the section). 3. The Manager selects the Technician to delete. **Input Data**: Technician ID or name to be deleted. 4. The Manager confirms the deletion. **Input Data**: Confirmation action (e.g., clicking "Delete" button).
Expected Result	- The selected Technician is deleted from the system. - The system displays a confirmation message indicating that the Technician has been successfully deleted.

Test Case 2: Error Handling - Technician Associated with Workorders

Scenario	Attempt to Delete a Technician Associated with Workorders
Preconditions	- The Manager is registered in the system. - The Manager has the necessary rights to delete Technicians. - The Technician selected for deletion is associated with one or more workorders.
Steps	1. The Manager logs into the system. **Input Data**: Manager username and password. 2. The Manager accesses the Technicians management section. **Input Data**: None (simple access to the section). 3. The Manager selects the Technician associated with workorders to delete. **Input Data**: Technician ID or name to be deleted. 4. The Manager confirms the deletion. **Input Data**: Confirmation action (e.g., clicking "Delete" button).
Expected Result	- The system displays an error message: "Cannot delete Technician as associated with workorders." - The Technician is not deleted from the system.

Here's the test report for the scenario "Add Technicians to their team" in English and in Markdown table format.

Test Report: Add Technicians to Their Team

Test Case 1: Successfully Add Technicians to Their Team

Scenario	Add Technicians to their team
Preconditions	- The Manager is registered in the system. - The Manager has the necessary rights to manage teams. - There are existing Technicians available to be added to the Manager's team.
Steps	1. The Manager logs into the system. **Input Data**: Manager username and password. 2. The Manager accesses the team management section. **Input Data**: None (simple access to the section). 3. The Manager selects existing Technicians to add to their team. **Input Data**: Technician IDs or names to be added. 4. The Manager confirms the addition. **Input Data**: Confirmation action (e.g., clicking "Add" button).
Expected Result	- The selected Technicians are added to the Manager's team. - The system displays a confirmation message indicating that the Technicians have been successfully added to the team.

Sprint 3 Technician

Test Report: Add Notes to a Workorder

Test Case 1: Successfully Add Notes to a Workorder

Scenario	Add notes to a workorder
Preconditions	- The Technician is registered in the system. - The Technician has the necessary rights to add notes to workorders. - There is at least one workorder available for the Technician to add notes to.
Steps	1. The Technician logs into the system. **Input Data**: Technician username and password. 2. The Technician accesses the workorder details. **Input Data**: Workorder ID or name to view details. 3. The Technician adds notes to the workorder. **Input Data**: Text of the notes to be added. 4. The Technician saves the notes. **Input Data**: Confirmation action (e.g., clicking "Save" button).
Expected Result	- Notes are successfully added to the workorder. - The system displays a confirmation message indicating that the notes have been saved successfully.

References

Fu, Chuang, Lu-Qing Ye, Yuan-Chu Cheng, Yong-Qian Liu, and Benoi Iung. 2002. "MAS-Based Model of Intelligent Control-Maintenance-Management System (ICMMS) and Its Application" 1: 376–80.

Kour, Ravdeep, Miguel Castaño, Ramin Karim, Amit Patwardhan, Manish Kumar, and Rikard Granström. 2022. "A Human-Centric Model for Sustainable Asset Management in Railway: A Case Study." *Sustainability* 14 (2): 936.

Malta, Ana, Mateus Mendes, and Torres Farinha. 2021. "Augmented Reality Maintenance Assistant Using Yolov5." *Applied Sciences* 11 (11): 4758.

Shankar, Lakshmi, Chandan Deep Singh, and Ranjit Singh. 2021. "Impact of Implementation of CMMS for Enhancing the Performance of Manufacturing Industries." *International Journal of System Assurance Engineering and Management*, 1–22.

Sorić, Dario. 2024. "How to Optimize Operations and Maintenance for Success." *Maintenance World*.