System Document for Cesla Waste Management Website

Zihe Liu 21207341
Xu Zhong 21207315
Niu Miaohe 21207525
Shiqiu Yang 21207321
Ziqi Wang 21207298

Contents

1 Abstract

Cesla Automotive is dedicated to producing high-quality electric cars while minimising environmental impact. They are mainly divided into different production and office departments. Our mission is to design a waste management system exclusively tailored to deal with waste production needs of large-scale car manufacturing firms like Cesla Automotive.

Our team of skilled software developers understands that their task goes beyond the creation of an efficient waste management system. The target is to also optimise the different stages involving waste recording, analysis, transportation and recycling. This system also allows individual users (both companies and private individuals who have business dealings with Cesla) to use the system in a similar manner to departments, by submitting their personal orders. Additionally, the government can use this system to access waste disposal data, which can serve as a basis for policy-making.

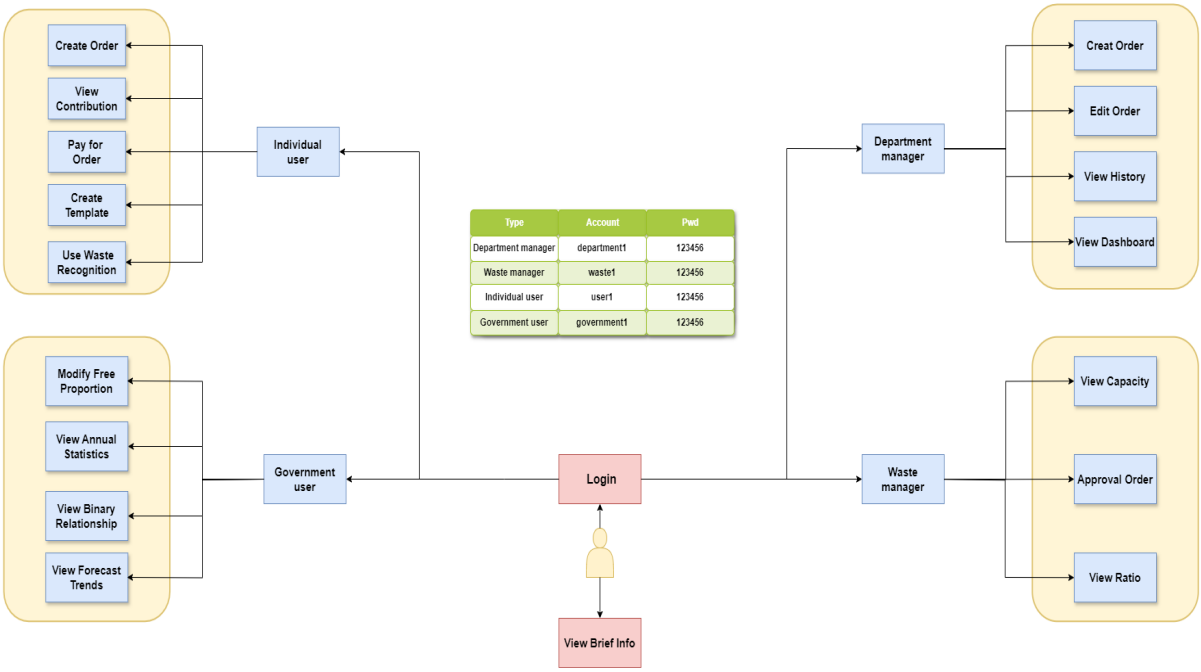


Figure 1: System Abstract

2 Introduction

2.1 Problem Statement

At present, Cesla faces key challenges that hinder their progress towards ideal waste management:

- Inefficiency in their current Waste Management: Traditional waste management methods employed lack in efficiency, causing delays in information relay and producing inaccurate data. This culminates in less-than-optimal waste management which needs urgent redressal.
- Environmental Impact: The notable volume of waste generated considerably hurts the environment, contributing to soil, water, and air pollution. This goes against Cesla Automotive's deeply ingrained environmental philosophy of minimising environmental impact.
- Inconsistent Waste Processing Standards: Different departments following disparate waste processing methods lead to inconsistent standards in handling waste.
- The stricter environmental policies introduced by the government currently prevent Cesla's handling methods from meeting legal requirements.
- Companies business dealings with Cesla on also have requirement of waste management.

This document aims to propose a systematic, efficient, and environment-friendly waste management system that can address these key challenges, thereby enhancing Cesla Automotive's commitment towards sustainability.

2.2 Solution

In response to these challenges, we present the Cesla Waste Management System (CWMS), an innovative solution specifically designed to transform and enhance waste management practices at Cesla Automotive. Leveraging cutting-

edge technology, CWMS offers a comprehensive solution for meticulous waste sorting, collection, processing, and insightful analysis. This system aligns with the principles of modern waste management strategies, environmental sustainability, and regulatory compliance. The core objective of CWMS is not just to address the immediate waste management challenges, but also to create a scalable and sustainable solution that will provide significant environmental benefits for Cesla Automotive, thereby strengthening the company's commitment to the environment.

This documentation provides an in-depth overview of CWMS, describing the system's architecture, functionality of each module, and the data flow processes. It also covers critical aspects such as database design, API documentation, strategic security measures in place, performance optimisation techniques employed, deployment strategy, as well as maintenance. The purpose of this document is to serve as a comprehensive guide for understanding, implementing, and maintaining this advanced waste management solution.

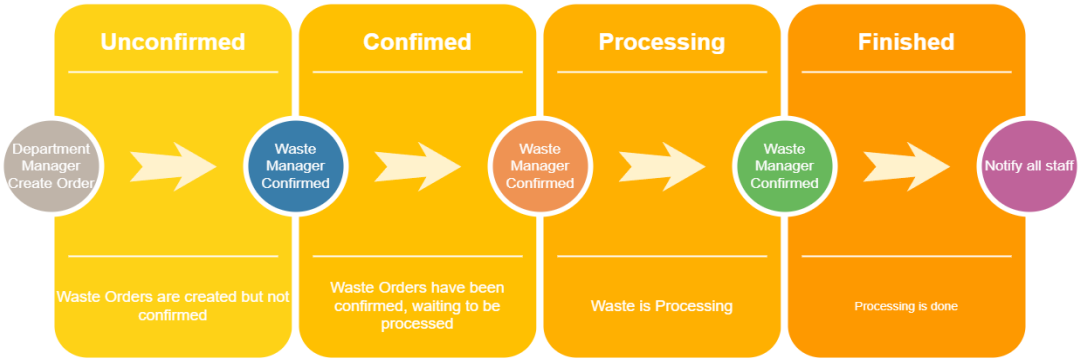


Figure 2: WorkFlow

2.3 Terms Explanation

Term	Description
Waste	Waste produced during Cesla's manufacturing process.
Department	The department of industrial management data provided to the development team by Cesla. It reflexes the real department in the Cesla
Department Manager	Responsible for reporting the department's waste over a period, requires writing detailed work orders, including waste type and properties.
Waste Manager	Oversees the entire process and approves work orders.
Order	Created by the department manager to record the attributes and related information of a batch of waste.
Government Representative	Environmental regulatory personnel from the government responsible for supervising Cesla's waste treatment to comply with sustainability policies and regulations.
Individual User	External individuals who can report external waste for processing through the system, which includes the company who has business dealing with Cesla and the individual user in public
Internal, external and external free	Waste from Cesla is internal waste, otherwise from external waste. External free is means the individual user can use this part capacity freely
Multiplier	An attribute of the work order; adjusts the waste treatment based on waste's hazard level and pollution, influencing resource use and processing capacity. It reflexes the waste disposal difficulties.
Storage Capacity	Reflects actual storage capabilities, varies with waste type, with specific data provided to the development team by Cesla.
Processing Capacity	Reflects actual waste processing capability, influenced by the multiplier, similar to storage capacity. However it is dynamic that depend on the orders and multiplier
Forecasting	The development team uses existing data to predict average processing times, capacity utilisation, and future waste production trends for each type of waste. It reflexes the trend of operation this system
Free proportion	It is a proportion that individual user can use in the external order. If is adjust be the government representative in this system. It depends on the total proportion of process capacity.

Table 1: Cesla Waste Management System Terminology

3 System Specification

The system is built using a three-tier architecture consisting of a presentation layer, application layer, and data layer. The presentation layer handles the user interface, the application layer manages the system's logic and functionalities, while the data layer stores and manages the system's data.

3.1 Technology Stack

FrontEnd	Backend	Database	AI Service API	Algorithm&Model
HTML&CSS&JS	FLASK	MySQL	ZHIPU AI	Yolo K-Means ARIMA

Figure 3: Tech Stack

- The back-end is developed using Flask, a Python-based web framework, and the front-end utilises HTML, CSS, and JavaScript. MySQL is chosen as the database management system to store and manage the system's data securely.
- In the back-end, Flask is primarily used as a lightweight framework, allowing for the creation of corresponding methods based on different routes. MySQL is chosen for the database due to its compatibility with Flask, facilitating connection through relevant code like 'db', and offering various ways to define primary and foreign keys for better retrieval and referencing between tables.
- For the front-end, native HTML is predominantly utilised. We have employed a variety of JavaScript libraries to facilitate the construction of our web pages. Specifically, we have utilised DataTables.js for the creation of dynamic tables, Chart.js and D3.js for the generation of intricate charts and data visualisations. Additionally, we have integrated several animation libraries to enhance the visual appeal of our web content. To augment the intelligence of our web-pages, we have incorporated AI models from Zhipuai, and we have also leveraged Bing's Map API to access geographical information.
- In terms of machine learning, both K-Means and ARIMA algorithms are employed. K-Means clusters centroids based on multidimensional data, with weight and processing time used as dimensions for two-dimensional clustering computation within the system. ARIMA predicts future waste generation trends based on historical data, effectively capturing peaks in waste production owing to its sensitivity to trends and seasonality.
- In deep learning, YOLO is planned for visual recognition tasks. It enables classification of waste types specific to Cesla by training on a customised dataset, allowing for inference based on a single user-provided image.
- Using a decision tree to predict the estimated completion time of waste disposal work orders, reasoning from waste type, weight, and chemical properties, using completed orders as the data model.

3.2 Functional Requirements

Below are the some core functional requirements of the system.

Req	PW	Description
Order Management	10	The system should allow the department manager to create, delete, view, and modify orders, which will be approved by the waste manager.
Order Process Visualisation	7	The system should provide department , waste managers and government representative with an intuitive visualisation of the current processing status of all orders.
Order Status Management	10	The system should allow the waste manager to conveniently manage the processing status of order. And also to check the data information of orders by dashboard
Visualisation Dashboard	8	The system should provide the waste manager with a visualisation dashboard.
Data Analysis	7	The system should perform data analysis to assist the government in making decisions for the next stage.
Waste Identification	6	The system should help individual users to identify waste types.
System Usability	5	The system needs to provide AI helper to allow users to get started quickly.

Table 2: Core Functional Requirements

At the initial stage of the project, we found it challenging at times to arrange task priorities. Higher PW values indicate a higher task priority. Therefore, we added a weight score, PW, in requirement analysis to assist us in task scheduling.

3.3 Non-Functional Requirements

Below are the system's non-functional requirements, providing a secure and user-friendly experience. This section covers security, performance, display feature, and usability.

1. **Security:**
   - Password in database and user insert is used hash code to keep safety.
   - All types user need to log in before using any function and operate the system.
   - All the configuration of this system only can access the Development and maintenance employee. Normal user can not access any data from front end.
   - The system should be robust enough to prevent any possibility of website breakdown due to surge in traffic from multiple accesses.
   - The system should be equipped with mechanisms to safeguard against cyber-attacks and unauthorised access to sensitive information database.
2. **Display:**
   - The application should feature smooth, visually appealing animations to enhance the user experience.
   - The interactive layout should be intuitive and logical, ensuring users can navigate the application easily and efficiently.
   - The interface of this system features dynamic data displays, which more intuitively reflect data trends and detailed information. This also allows for faster data presentation to users, improving user experience.
3. **Usability:**
   - The platform should be user friendly, allowing different types of users in different situations proficiently use all features in a few days.
   - The system add the editing and using templates to help user avoid repeating inserting something.

4 Design and Implementation

4.1 Database Design

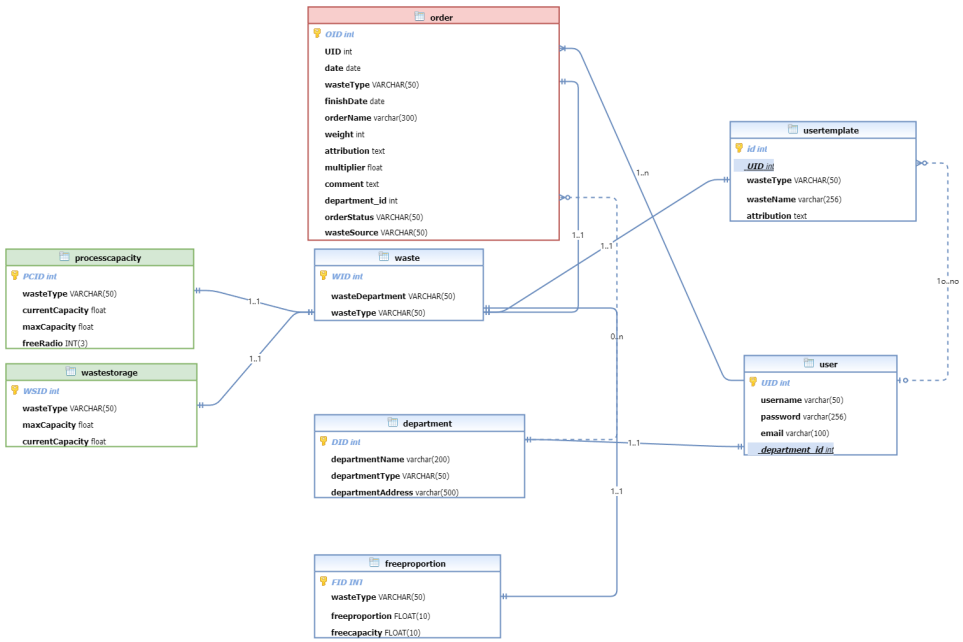


Figure 4: Database Design

We use MySQL as the database for the whole system, which has totally seven tables (shown in figure 4), the following are details about seven tables (user, department. order, waste, usertemplate, storagecapacity, processcapacity)

4.1.1 Fix Enum

Within the system, there are some fixed attributes that do not change as the enterprise evolves.

- Firstly, there are four types of users in the system:

```
DEPARTMENT
WASTEMANAGER
GOVERNMENT
INDIVIDUAL
```

- Detailed descriptions are already provided in term explanation, and will not be repeated here. Waste sources are categorised into two types:

```
INTERNAL
EXTERNAL
EXTERNALFREE
```

- Wherein, work orders filled out by DEPARTMENT are internal, and those filled out by INDIVIDUAL are external. There are four states for orders:

```
UNCONFIRMED
CONFIRM
PROCESSING
FINISHED
```

  These correspond to the four stages of actual waste processing.

4.1.2 Flexible Enum

Waste types and department also can be recognised as the Enum in the code to improve performance and more easily writing code. While in theory it should be variable, practical usage involving UI design, icon matching, and other front-end issues means that making related modifications is a difficult task for non-software developers; therefore, the functionality to add more categories and departments has not been provided. Since the delivery is one-time and actual usage will not change frequently, it is tentatively decided to design the UI with a fixed number of nine departments and twenty-one categories. The back-end code has been designed to retain extensibility to accommodate more categories and departments (stored as `VARCHAR` in the database, allowing for additions using database commands). The front-end will also dynamically fetch the relevant enumerations, so upgrading and expanding requires very minimal personnel and time costs.

- Department types data is provided by Cesla, and currently information for nine departments has been obtained. This includes all types of Cesla departments and the industries involved:

```
METALLURGY
EQUIPMENT_MANUFACTURING
COMPOSITE_MATERIAL
NEW_ENERGY
AUTOMATION_SYSTEM
MAINTENANCE
LABORATORY
DATA_CENTER
OFFICE
```

- Waste types are stored in the database in text form and converted into enumerations for use during operations. However, the department only can add order whose waste type in the relative department. The individual user can add all types waste type. The categories are as follows:

Waste Category	Department
HEAVY METAL WASTEWATER, EXHAUST GAS, MINERAL RESIDUE	METALLURGY
CUTTING FLUID, METAL CHIPS, PLASTIC, COMPOSITE MATERIAL CUTTING WASTE, WASTE PAINT	EQUIPMENT MANUFACTURING
DUST, CHEMICALS, CATALYZER	COMPOSITE MATERIAL
CHEMICAL PROPELLANTS, FUEL RESIDUES	NEW ENERGY
DISCARDED ELECTRONIC COMPONENTS	AUTOMATION SYSTEM
HYDRAULIC OIL, LUBRICANT WASTE	MAINTENANCE
HAZARDOUS CHEMICALS, WASTE EXPERIMENTAL EQUIPMENT	LABORATORY
WASTE HEAT	DATA CENTER
WASTE PAPER, HOUSEHOLD WASTE	OFFICE

Table 3: Waste Categories and Their Associated Departments

4.1.3 User

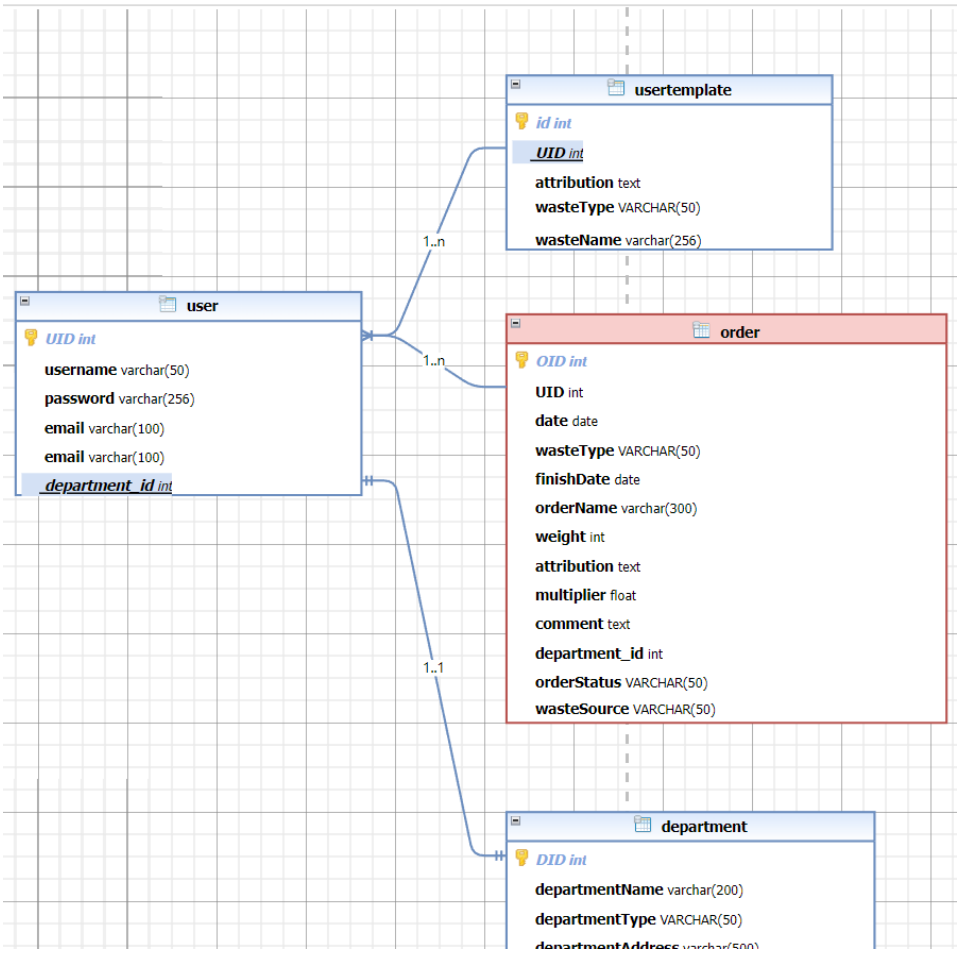


Figure 5: User table

User table (Figure 5) is a basic table, which records the basic information of users and their types, serving as a basis for later determining the source of waste.

4.1.4 Department

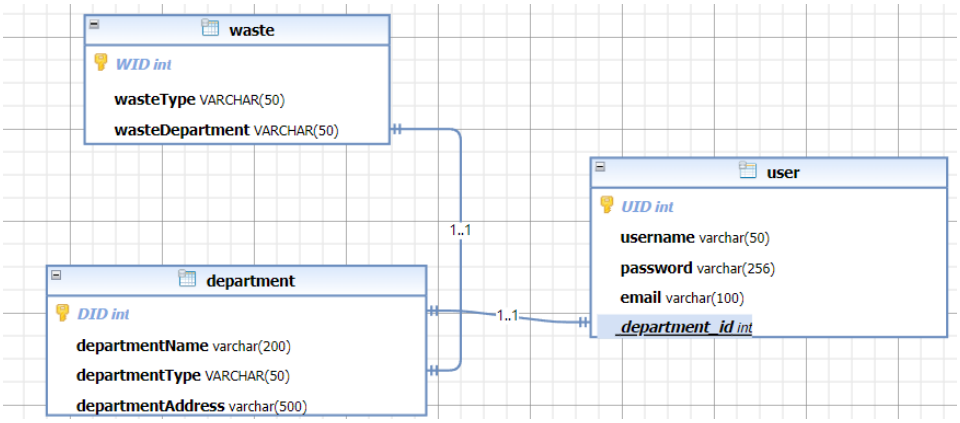


Figure 6: Department table

The department table (Figure 6) records the departments in Cesla where waste is generated. It contains basic information about the departments, their corresponding department managers (responsible for reporting waste work orders), and department types to correspond to the types of waste (which can be transformed into enums in code).

4.1.5 Waste

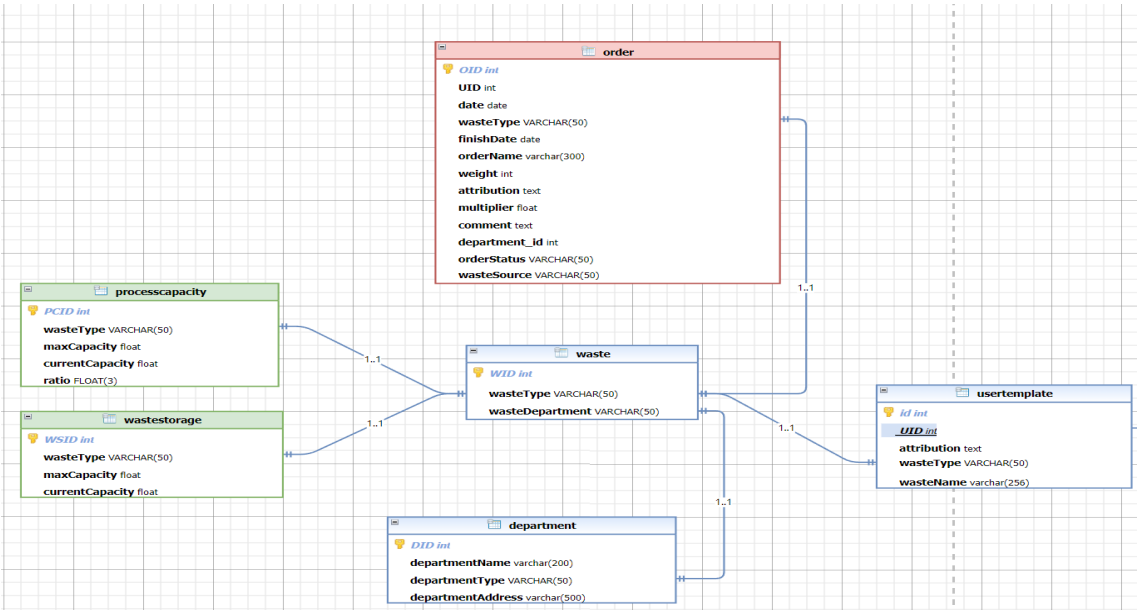


Figure 7: Waste table

The waste table (Figure 7) is the most important basis in the database, recording the name of the waste, its type (which can be transformed into enums in code), and the corresponding department (as listed in the department table).

4.1.6 Order

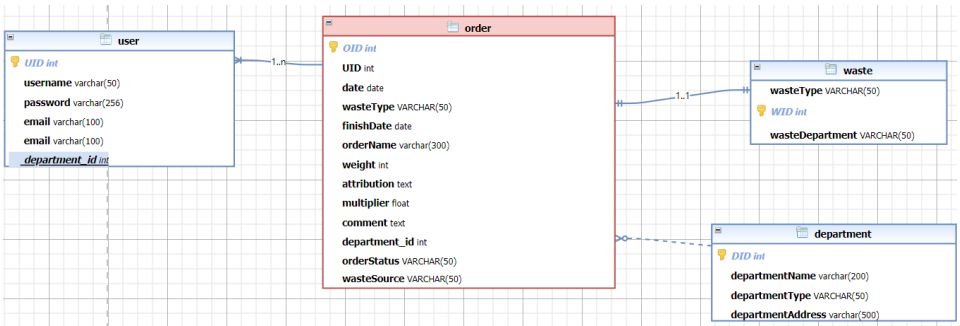


Figure 8: Order table

The order table (Figure 8) is crucial for recording the attributes of waste.

- OID: Unique ID of the work order
- UID: Department head or individual user
- date: Date the work order was created
- finishDate: Time when the work order reached completion
- orderName: Name of the work order
- wasteType: Type of waste, corresponding to the waste table
- weight: Weight of the waste
- attribution: Attributes, used to record the components and properties of the waste (such as the main component being sulfur dioxide, pH value being acidic)
- multiplier: Ratio used by waste administrators to manually adjust the actual processing capacity input based on professional knowledge
- comment: Used to write some remarks
- orderStatus: Processing status, with four possible statuses
- department-id: Records the department ID for internal work orders
- department: Records the department for internal work orders
- wasteSource: Records the source of the work order, used to differentiate trends and calculate usage shares

  The order, as a mapping of actual waste in the system, represents the transportation and actual treatment of waste in each state switch.

4.1.7 Usertemplate

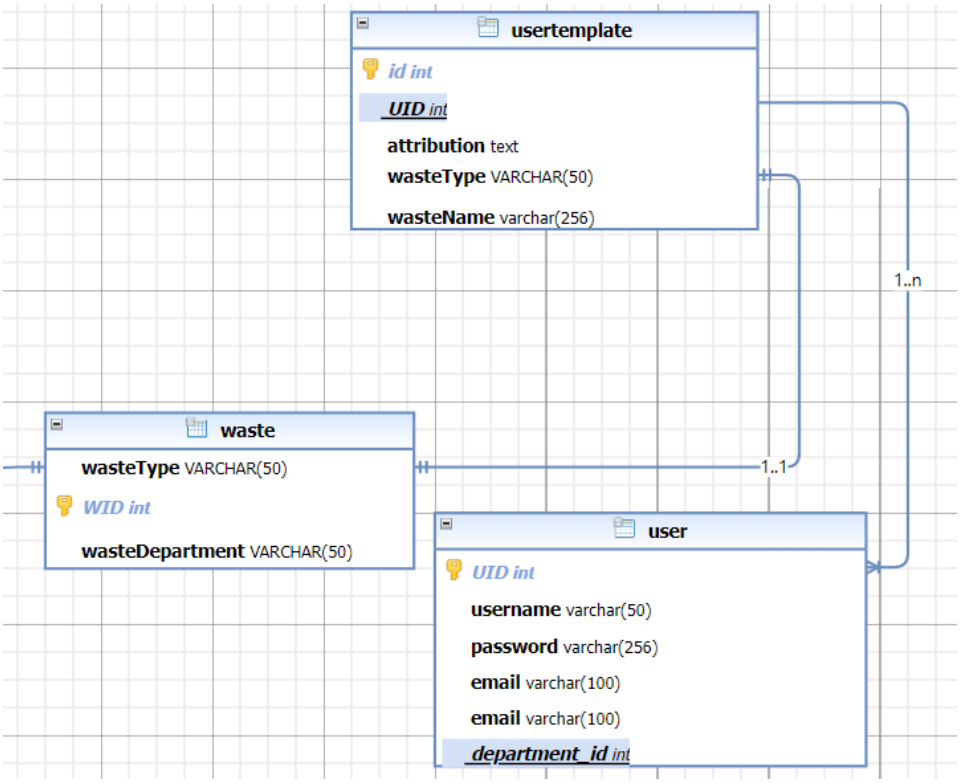


Figure 9: Usertemplate table

The usertemplate table (Figure 9) allows users to create fixed waste attributes, which help reduce the time required to fill out work orders. Templates are bound to users and only accessible to their own templates. The user can use template to change the attribution of orders and wastes which store in the database to create order both department managemer and individual user.

4.1.8 Storagecapacity and Processcapacity

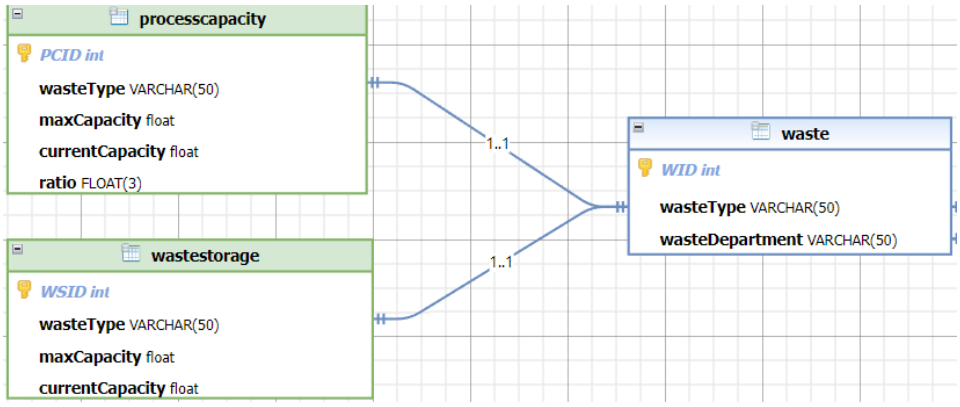


Figure 10: Storagecapacity and Processcapacity table

The Storagecapacity and Processcapacity tables (Figure 10) record the storage and processing capacities of actual processing equipment. Both tables record the maximum storage/processing capacity and the currently used capacity. However, compared to storage capacity, processing capacity has two additional differences: Firstly, storage capacity only calculates the actual weight, while processing capacity needs to calculate additional processing capacity occupation based on the multiplier. It is simply calculated as the multiplier multiplied by the actual weight as the final result. Secondly, an additional column is added for the free ratio, controlled by the government, indicating the portion of processing capacity that external users can use for free. Once the free processing capacity is used up, payment is required for further usage.

4.1.9 Free Proportion

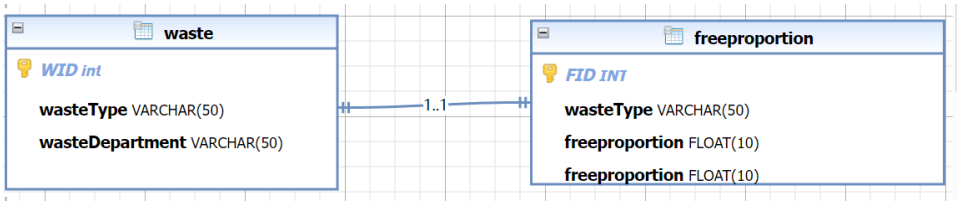


Figure 11: Free Proportion

Free Proportion (Figure 11) is aim to storage the free proportion which can be used by individual user. It is adjusted by the government who can check statistic of whole waste management system. The free proportion can adjust by the front end table and update the specific free capacity automatically (it will consider the part which has been used). If the user has run out the free proportion, the user need to pay for this order by can OR code.

4.2 Functional Requirements Implementation

In the following section, we will delve into the functional requirements as outlined by users and stakeholders, and the corresponding implementation details for each requirement. The functional requirements have been documented as a series of user stories, followed by the solutions that were designed to meet these requirements.

1. **Requirement 1: CRUD operations of orders by the department head**

   User Story: As a department head, I want to have the ability to perform create, read, update, and delete (CRUD) operations on orders. This capability is necessary for effective management of resources and operational efficiency.

   Solution: Our system provides an intuitive and user-friendly interface allowing department heads to create new orders, modify existing ones, view the entire list of orders, and delete unnecessary ones. The interface combines server-side scripting for database manipulation, and client-side scripting for the user interface, providing a smooth user experience.

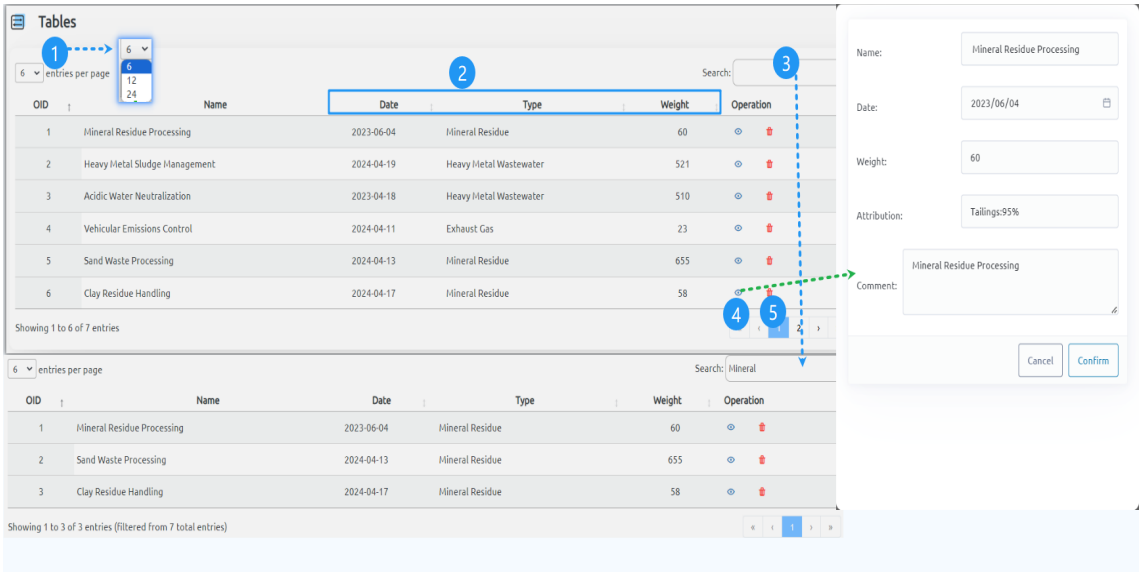


Figure 12: Order CRUD

2. **Requirement 2: Visualise the current process status of all orders for the head of department**

   User Story: As the head of department, I want to intuitively see the current processing status of all orders. This visibility will allow me to monitor progress and address issues promptly, enhancing efficiency and accountability.

   Solution: Our system implements a dashboard offering a visual representation of the current status of all orders. The dashboard system updates in real-time and utilises colour-coding to differentiate the stages of each order- allowing the head of department to understand the order status at a glance. Additionally, clicking on orders gives access to more detailed information, supporting informed decision-making.

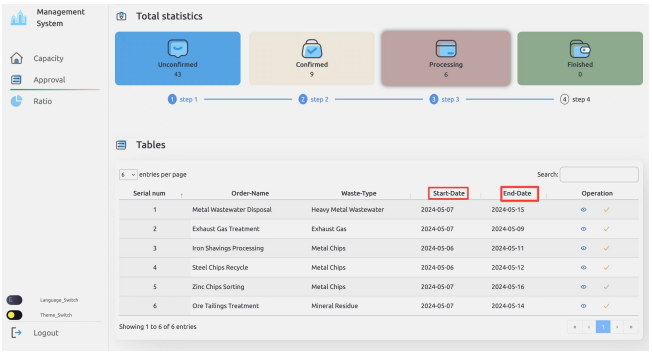


Figure 13: Order Management

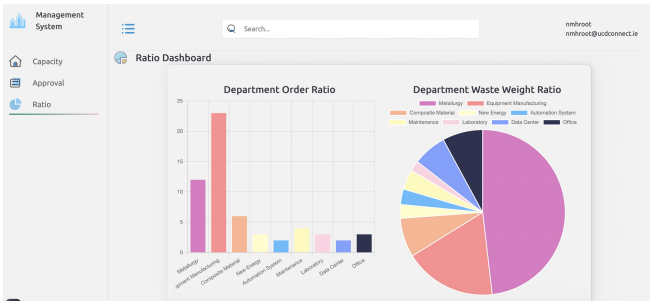


Figure 14: Ratio View

3. **Requirement 3: Visual dashboard for the waste manager**

   User Story: As a waste manager, I want to have a visual dashboard where I can easily view, manage, and analyse all of the orders. A comprehensive overview will simplify the monitoring process and lead to efficient administration.

   Solution: Our system introduces a highly interactive and visually appealing dashboard for the administrator. Therein, complex data related to orders are represented in the form of charts, graphs, and tables, making it easier to analyse and understand the system's current state. It updates in real-time, providing the administrator with the most recent information. There are also options to filter, sort, and search orders, allowing for an easier and more efficient management process.

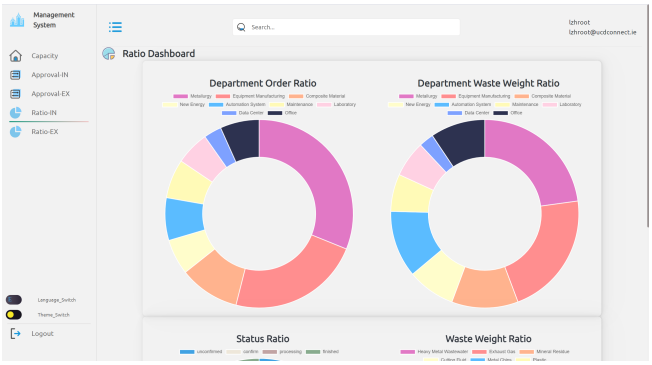


Figure 15: View ratio in ring

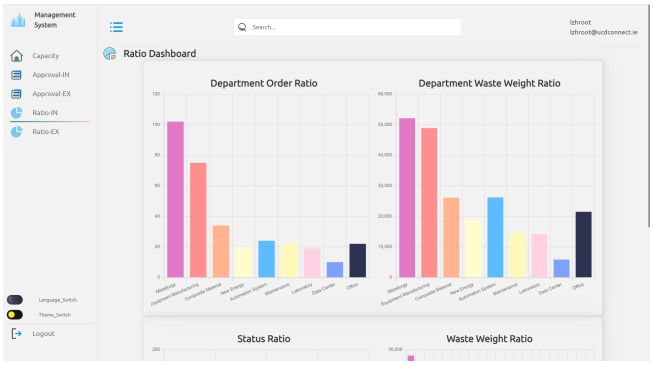


Figure 16: View ratio in histogram

4. **Requirement 4: Data analysis to aid government decision-making**

   User Story: As a government official, I want the system to provide data analysis that can assist me in making informed decisions. The use of data-driven insights will allow us to optimise our operations and policies.

   Solution: Our system is equipped with a powerful data analytics module. It can collect meaningful data and generate comprehensive reports presenting trends, patterns, and projections. Additionally, it can create interpretive visualisations and predictive models to provide decision-makers with actionable insights. These insights can guide the formation of effective strategies and execution of conscious decisions. System has already implemented mean prediction using K-Means and used ARIMA to forecast future trends.

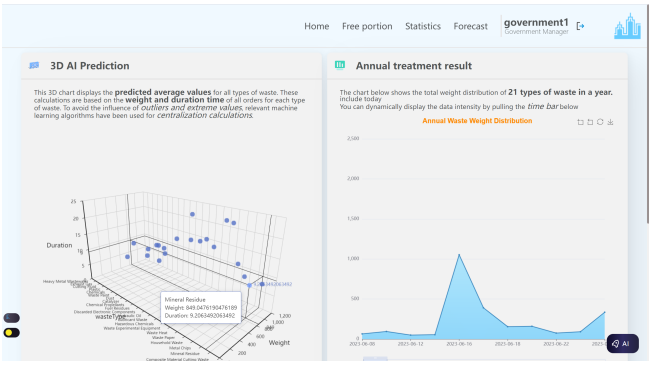


Figure 17: View statistics

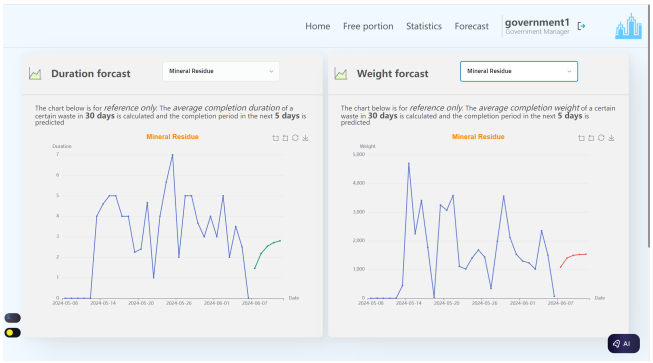


Figure 18: View trend

5. **Requirement 5: Photo uploads by individual users with automatic wastage identification**

   User Story: As an individual user, I want to be able to upload photos so that the system can automatically identify the type of waste. This feature would make it easier and more efficient for me to handle waste disposal without misunderstanding or confusion about waste categorisation.

   Solution: In our system, we integrate advanced image recognition technology. With this, when a user uploads a photo of waste, the system can analyse the photo and accurately identify the type of waste it represents. The result is promptly displayed to the user, assisting them in handling their waste in the most appropriate way according to its classification. This feature brings convenience and promotes the correct way of waste management.

   System plans to use YOLO with custom weights to identify images uploaded by users.

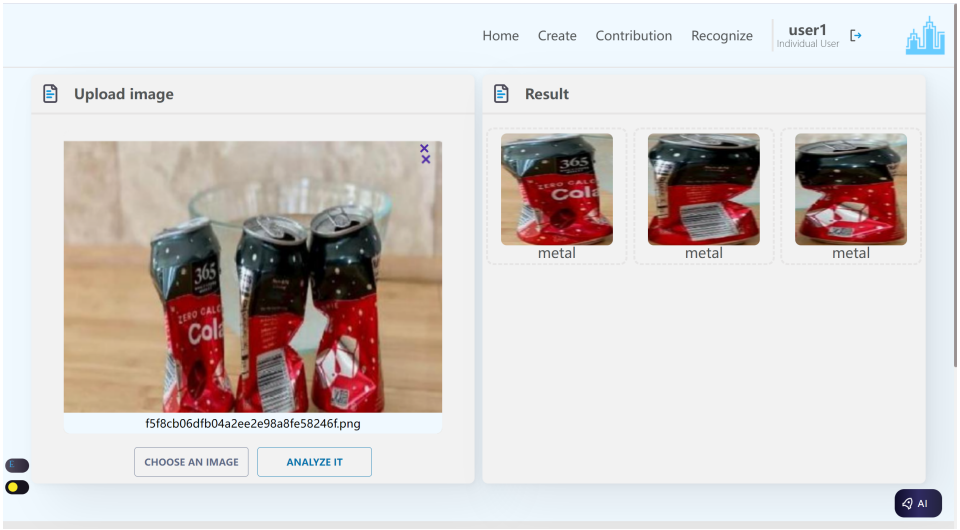


Figure 19: Visual Function

4.3 Non-Functional Requirements Implementation

The following non-functional requirements reflect the overall vision for the system's operations. They emphasise the performance, usability, and security aspects of the system to ensure a seamless user experience for our diverse user base. Please note that these requirements are not ordered based on priority.

1. **Feature 1: Theme toggling**

   The system shall include an option for users to freely switch between light and dark themes to promote optimal user experience.

2. **Feature 2: Language toggling**

   The system shall support both English and Chinese language selection for global and local users. (Currently only English and Chinese Mandarin are provided)

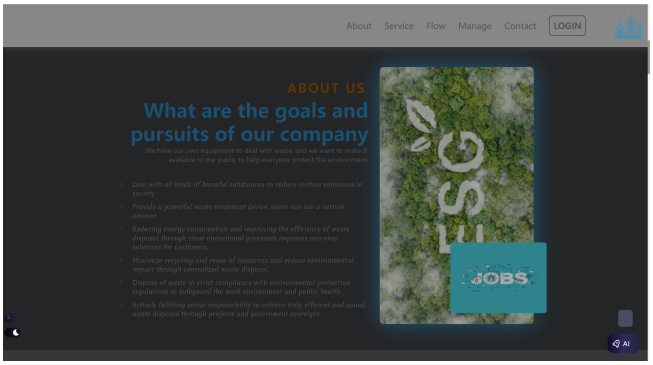


Figure 20: Dark Theme in English

Figure 21: Light Theme in Mandarin

3. **Feature 3: Seamless animation**

   The system shall deploy smooth and fluid animations within its interface to enhance the visual experience for users. To illustrate, when the website is loading, related animations will be presented to entertain the users while they wait. Another instance is during theme switching, transitioning from light to dark mode (or vice versa) will be accompanied by a visually pleasing animation that makes the switch appear seamless. Furthermore, interactive elements such as images will incorporate animations - for example, when users hover their mouse over an image, it will progressively enlarge, providing a dynamic and interactive user experience. We have also engineered a suite of intriguing front-end effects to enhance user interaction. For instance, we have implemented an automatic generation of progress bars that dynamically reflect the status of ongoing processes. Furthermore, we have designed a feature that allows for the seamless transformation of a specific section from vertical to horizontal scrolling as the user navigates up or down the page, effectively representing a workflow. Additionally, we have constructed an infinite scrolling effect, which enables the continuous dragging of the page within a designated area. This effect is underpinned by a robust arrangement mechanism that ensures the content within is always displayed in an orderly and accessible manner.

4. **Feature 4: Window Responsiveness**

   The system shall be capable of automatically adapting its layout based on the size of the window in which it is viewed to ensure visibility and accessibility of system features.

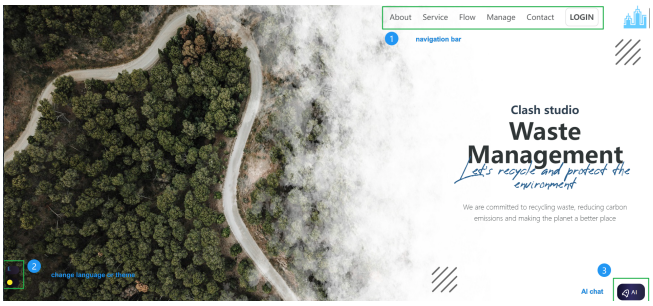


Figure 22: Full-screen Display1

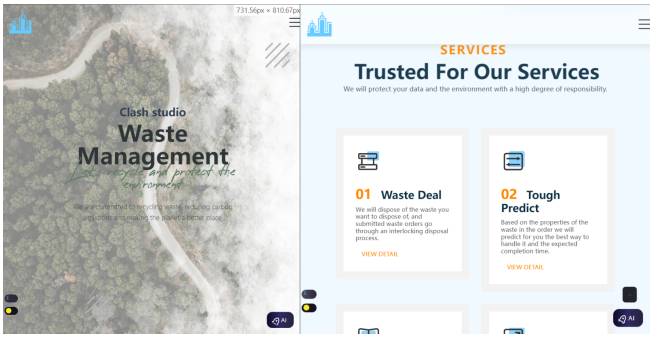


Figure 23: Half-screen Display1

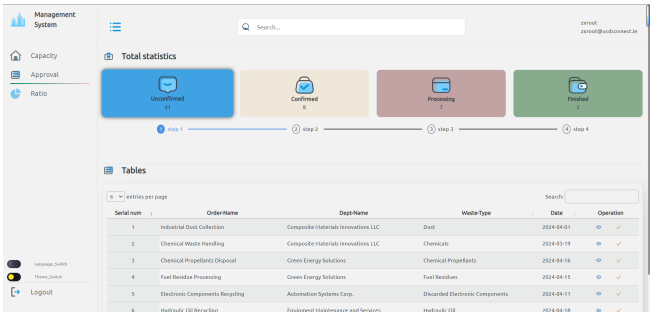


Figure 24: Full-screen Display2

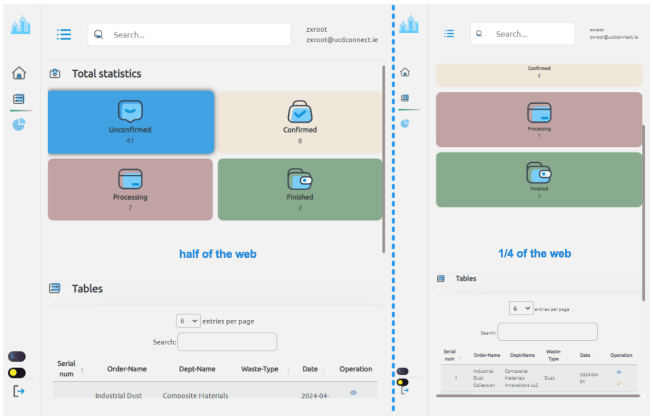


Figure 25: Half-screen Display2

5. **Feature 5: Cross-platform Availability**

   The system shall be accessible and functional on both desktop and mobile platforms to ensure availability of service for users regardless of device.

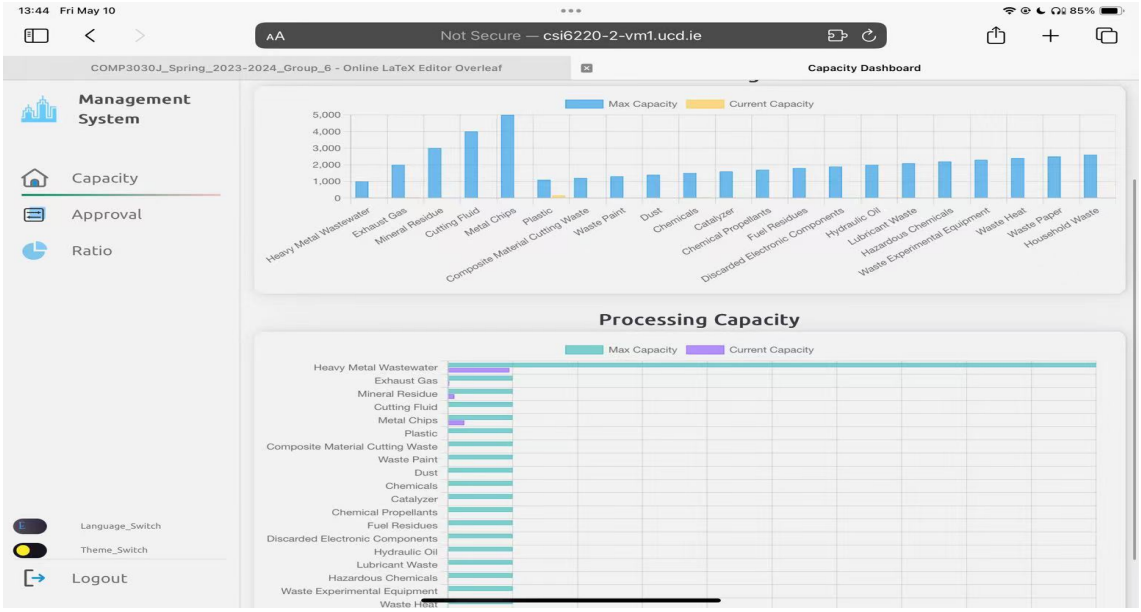
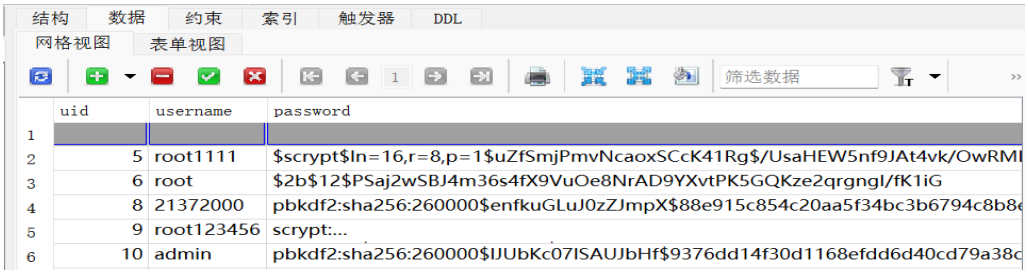


Figure 26: Login via iPad

6. **Feature 6: Database Hash Encryption**

The system shall employ hash encryption on the database to ensure security of user and system data.



	uid	username	password
1			
2	5	root1111	$scrypt$ln=16,r=8,p=1$uZfSmjPmvNcaoxSCcK41Rg$/UsaHEW5nf9JAt4vk/OwRMI
3	6	root	$2b$12$PSaj2wSBJ4m36s4fX9VuOe8NrAD9YXvtPK5GQKze2qrgngl/fK1iG
4	8	21372000	pbkdf2:sha256:260000$enfkuGLuJ0zZJmpX$88e915c854c20aa5f34bc3b6794c8b8
5	9	root123456	scrypt:...
6	10	admin	pbkdf2:sha256:260000$IJUbKc07ISAUJbHf$9376dd14f30d1168efdd6d40cd79a38c

Figure 27: Encrypted PWD

5 Test Guide

Our team recognises and practices the significance of software testing. Our testing protocol embodies our technical commitment and professionalism. Each of our members does not test their own developed functionalities. Instead, we test the functionalities that are written by other team members. This approach aids in eliminating the bias and blind spots that come with coding, bolstering our confidence in code quality.

5.1 Unit Testing

Flask provides powerful interface to write Unit Test, here is an instance:

```
import unittest
from flask_testing import TestCase
from App import app, db
from App.models import Department

class TestDepartments(TestCase):
    def create_app(self):
        app.config['TESTING'] = True
        app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///:memory:'  # Use in-
            memory SQLite database for testing
        return app

    def setUp(self):
        db.create_all()
        test_department = Department(departmentType='Test Department')
        db.session.add(test_department)
        db.session.commit()

    def tearDown(self):
        db.session.remove()
        db.drop_all()

    def test_get_departments_name(self):
        response = self.client.get('/get_departments_name')
        self.assertEqual(response.status_code, 200)
        data = response.get_json()
        self.assertEqual(len(data), 1)
        self.assertEqual(data[0]['departmentType'], 'Test Department')

if __name__ == '__main__':
    unittest.main()
```

The selected code is a route handler for the Flask web application, specifically for the route **/get_departments_name** with a **GET** method. When this route is accessed, the function **get_departments_name** is executed. This function retrieves all the departments from the database using **Department.query.all()**, then iterates over each department to create a dictionary containing the department type name. These dictionaries are appended to a list, **departments_list**. Finally, the function returns this list in JSON format using **jsonify(departments_list)**. This effectively provides a JSON response containing the types of all departments when the **/get_departments_name** route is accessed.

5.2 User Acceptance Testing

During the pilot run of the project, we implemented User Acceptance Testing (UAT). This testing method is our main way of collecting feedback. We invited actual users to experiment with our products to collect their user experience and feedback. This stage is extremely important because it allows us to view the product from the user's perspective, identify potential problems and bugs that may be overlooked by the development team, and optimise the product accordingly. Our test users actively participated in this process and provided us with valuable suggestions and feedback, enabling us to better improve our product.

6 Deployment and Maintenance

Based on our programming languages and development environment (Flask+MySQL+HTML+JS+CSS), we use the following methods to deploy our code to the server:

6.1 Basic Environment

For the Python runtime environment, since Ubuntu 22.04 inherently includes Python allowing for execution of Python commands, there is no need to install an additional Python runtime environment. The development team has deployed a basic Flask project.

6.2 Running Environment

Due to Python's extensibility, the project itself requires certain non-standard libraries for functional extension. To better manage software packages, a virtual environment is necessary to avoid installing packages directly on the local machine.

For the software packages used, virtual environments are also created on the local machine using **Anaconda**. The packages and their corresponding versions are saved using the command:

```
pip list --format=freeze > requirements.txt
```

On the server side, the packages are installed using:

```
pip install -r requirements.txt
```

6.3 Nginx and gunicorn Environment

After completing the basic deployment, as the executable files are defined, we proceed with configuration following the tutorial provided by the professor, replacing the directories in gunicorn with those of the virtual environment and the path to the executable files (special thanks to the professor for the guidance).

6.4 Database Environment

As the project uses MySQL as the backend database, the development team needs to configure the MySQL server on the UCD server, initialising and setting passwords under the root environment. Simultaneously, the password for connecting to the database in the project code is updated.

6.5 Data Importing

Since there are no actual data to run the project initially, basic data are required for the operation, necessitating data transfer on the UCD server. The local database uses Flask's built-in migration feature to migrate and establish initial tables, with specific commands as follows:

```
flask db init
flask db migrate
flask db upgrade
```

After setting up the database and populating it with the necessary data, the `mysqldump` command is used to export the .sql file. The project files are then uploaded to GitHub. After cloning, a database with the same name is created in the MySQL command line, and the data is imported using the `source` command.

6.6 Maintenance

After successfully deploying the initial code and getting it to run, it is necessary to gradually add new features. Three days before critical milestones, all functionalities must be tested locally to ensure they are error-free. Following the updates of the required data and approval from the functional manager, the changes should be merged into the main branch. Then, pull operations are performed on the server to update the code, and comprehensive testing is conducted on the server to ensure that the new features do not interfere with the existing functionalities and are working correctly.

If any issues arise, the code should be rolled back to ensure that it remains usable by others, and an issue should be submitted. If specific environment testing on the server is needed (such as testing if a Chinese API can be used in Ireland), it is necessary to consult with the manager. The code should be uploaded to a branch for testing in a separate environment. After testing, any code on the server used solely for testing purposes should be removed, keeping only the code that has been delivered and accepted.

Regardless of whether the code has been successfully tested, it is necessary to restart Nginx and gunicorn whenever changes are made to ensure that the modifications take effect.

6.7 Problem solving

The development team also encountered some issues related to server deployment.

- Case Sensitivity Issues in Ubuntu
  When deploying MySQL, there are differences in default settings between Windows and Ubuntu. Ubuntu is case-sensitive by default, while Windows is not. In the code, case insensitivity is sometimes required (as enumeration values are used). However, on Ubuntu, case sensitivity led to errors. After realising this, the development team modified the relevant parts of the code to standardise the format and strictly defined usage norms. Specifically, enumeration classes use uppercase, and MySQL-related identifiers are all in lowercase. This ensures stability across different systems.
- Viewing gunicorn
  Because Nginx proxy and gunicorn are used to launch Flask, errors do not directly appear in the command line. The following command is needed to view the logs:

```
sudo journalctl -u gunicorn
```

  Checking the logs helps in pinpointing issues.

7 Teamwork and Project Management

Following is the deliverable schedule:

7.1 Developing Plan

Milestone	Description	Week
Milestone 1	Establish topic, clarify problem statement and complete proposal, complete requirement analysis, establish requirements documentation, confirm the development model as the agile model.	5
Milestone 2	Establish technical stack, database establishment, implement basic website functions such as login, registration, etc.	6
Milestone 3	Complete 60% of the basic features, including order management, data visualization, etc.	10
Milestone 4	Overcome the final 40% of enhanced features, including image recognition, and use machine learning algorithms for data analysis.	12
Milestone 5	Delivery phase, complete project testing, open trial to users, fix bugs.	13

Table 4: Project Milestone Plan

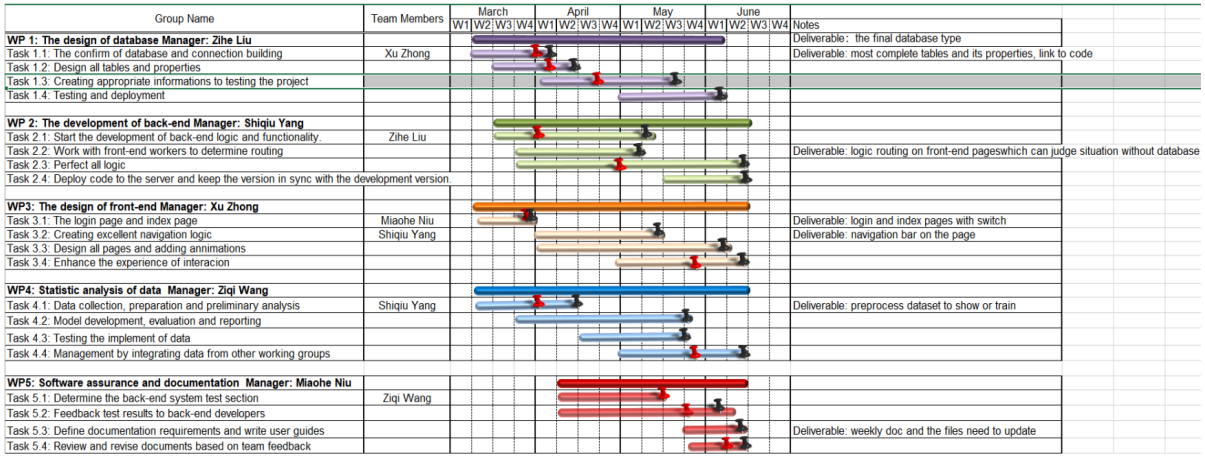


Figure 28: Gantt Overview

7.2 Team Member Roles

We utilize GitHub as our primary platform for code management, fostering seamless collaboration and ensuring version control integrity within our team. Below is the division of roles within our project.

- **Zihe Liu - Backend Development**: On Backend Development, Zihe Liu is dedicated to creating, managing, and optimizing the database and critical functions of the system, ensuring smooth communication between front and back ends. Equally important is Liu's role in providing guidance. Liu assumes responsibility for guiding the team in understanding and implementing the backend logic, ensuring all members are aligned and aware of how the data is managed and used effectively.
- **Xu Zhong - Front-end Development**: Xu Zhong, leading the front-end development, excels at creating engaging UI and implementing animations, providing an intuitive and dynamic digital environment for users. As a guiding light, Zhong takes responsibility for sharing his technical expertise with the development team, enhancing overall productivity and ensuring the effective design and function of the front-end system.
- **Miaohe Niu - Front-end Development**: For the front-end Development, Miaohe Niu ensures web pages' aesthetic and functional quality, while also taking a hands-on role in using JavaScript functions to maximize user interactivity. Niu actively seeks out areas in the front end that need refining, identifying and implementing system improvements to streamline the overall user experience.
- **Shiqi Yang - System Improvement; Documentation and Design**: Shiqiu Yang, with the crucial responsibility of System Improvement, excels in refining documentations, UIs, as well as identifying potential enhancements within the system. Yang holds a key role in Documentation, creating clear and concise system documentation for maximum transparency. Along with these responsibilities, Yang participates in Design, collaborating on the aesthetic and functional elements of the system.
- **Ziqi Wang - Documentation and Design**: Simplifying complex concepts through Documentation, Ziqi Wang ensures clarity and detail are provided throughout all system documents. Wang's work helps elucidate the system to everyone who reviews the documents. Along with documentation, he plays an important role in the System's Design, contributing creative ideas and offering practical solutions.

7.3 Problems during development

• **Database conflicts**: In our development, database conflicts often occur after a project version changes. As a result, login error or test data can not be loaded. Discrepancies between deploying the server and testing locally are often due to a poorly structured database, and sometimes the table names don't match names in the route functions.

• **Issues in Deployment**: During the deployment of our project on the server, we encountered many technical problems, such as the connection between the server and the github project repository was not successful at the beginning, the deployment was not fully kept up with the deployment, there were some problems at the beginning of the deployment, and the login data could not be received after the deployment, so the user could not log in.

• **Data using in machine and deep learning**: machine learning need the Strict data limitation, so we need to change order attribution and data to be a suitable input for this models.

• **Animation design and Implement**: In the front-end design, there will be a conflict of opinion between the designers and the technical developers. In the process of implementation, some libraries were called that did not fulfil our requirements well, and the layout and testing of some components made developers very confused.

• **Schedule mismatch**: Some parts of the project progress faster than others, leading to coordination problems Task prioritization: Difficulty in prioritizing tasks and what to focus on next.

• **Communication gap**: The occasional lack of communication that leads to misunderstanding.

7.4 Solutions

• **Ask help for Eddie and TA**:
We asked our TA for advice, asked her experience in developing projects during meetings, and asked her to give us some suggestions for our envy. Ask Eidde for technical advice. Resolved the deployment server and some database conflicts.

• **Strengthen WeChat meetings:**
Hold short team meetings to align progress and adjust plans. Prioritise tasks and set clear goals for each phase. Synchronise the function name and database.

• **Multi-person collaboration**: When two or more people work on a work package, it greatly speeds up the development schedule of the project. It also reduces errors at the same time.

8 Change Log

The change log keeps track of all the changes made to the system, including bug fixes, feature enhancements, and updates.

For detailed changes and commit records, please refer to the GitHub repository's commit history:

GitHub Repository: `https://github.com/lzhhe/COMP3030J-Software-Engineering-Project2`

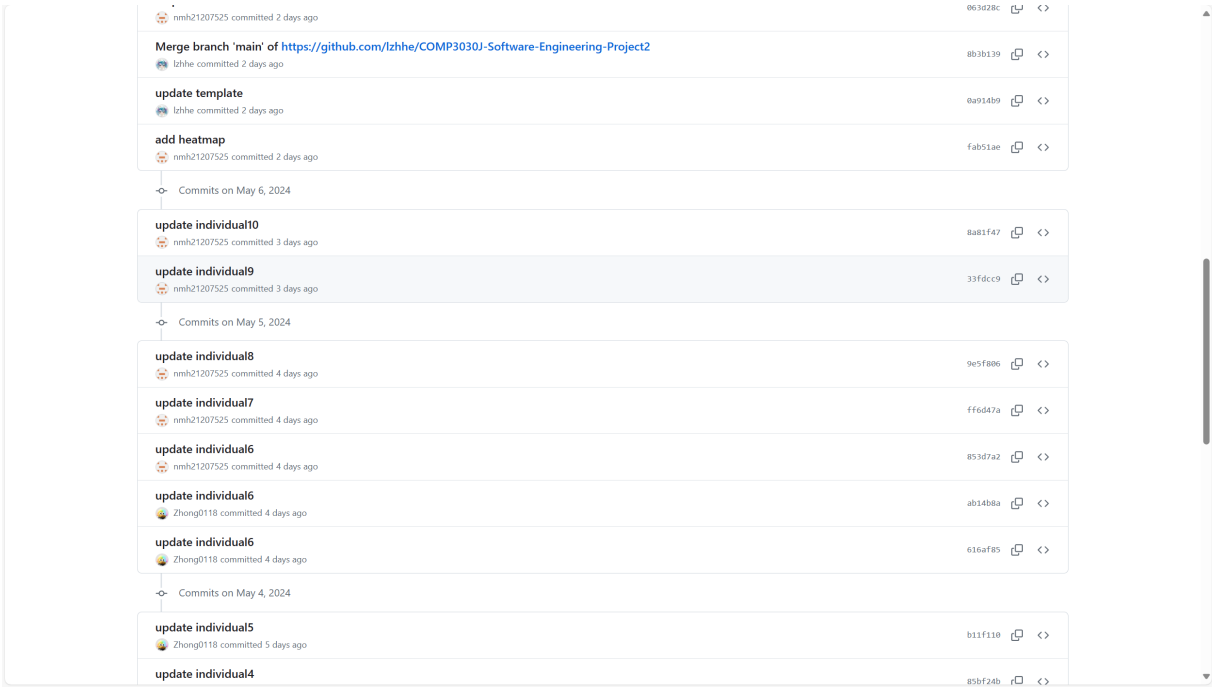


Figure 29: History Logs

In this project development process, we have chosen GitHub as our primary platform for code hosting and version control for several key reasons:

- Enhanced Version Control Capabilities: GitHub provides a robust version control system that meticulously records every change made to the code, including timestamps and information about the changer. This allows us to easily track the history of each code commit, preserving the differences between versions, thereby facilitating version management and progress tracking. This aids us in adhering to the set Gantt chart for development.
- Support for Concurrent Development With its branching and merging strategies, GitHub supports concurrent work on different functionalities by team members without interference. Front-end and back-end development can proceed simultaneously without affecting each other. Additionally, new features are developed in new branches to ensure the usability of the original version.
- Simplified Deployment Process GitHub allows the entire project to be cloned through the clone command without loss of files, which assists us in deploying code to UCD servers.

- Disaster Recovery and Code Rollback In the inevitable event of encountering irreparable severe bugs and logical confusion during code development, GitHub's distributed version control capabilities enable us to quickly rollback to a previous stable version. This rapid recovery maintains development progress by discarding problematic code and also assists in problem identification.
- Project Transparency and Monitoring GitHub provides comprehensive project management tools such as issue tracking, project boards, and milestone settings. These tools help maintain transparency within the project team, ensuring that all members can see the latest progress and status in real-time. Additionally, the submission of issues by testers and developers alerts the development team to additional problems, contributing to the stability of the project.

  .

Through these features, GitHub not only supports our technical needs but also optimises workflow and enhances team collaboration efficiency, making it a critical tool for the success of our project.

This translation conveys the same detailed reasoning and advantages of using GitHub for your project as mentioned in your original text.

9 Conclusion and Acknowledgements

In this project, our team utilized the method of agile development, which greatly boosted our work efficiency and patience. At times, we faced challenges and disagreements amongst team members, but the flexibility of agile development and our own spirit of teamwork led us to overcome these challenges collectively. Through this process, we enhanced soft skills such as leadership, organisation, and problem-solving abilities among every member of our team. We discovered that communication and the rapid adapting to changes are key to successful agile development. By intensifying routine meetings, we ensured consistent and efficient communication amongst team members, establishing a robust and real-time feedback mechanism, enabling us to quickly respond to changes in project requirements.

Lastly, we extend our heartfelt thanks to our TA, DU Wanxuan, who tirelessly provided us with various forms of help, holding at least one meeting a week to assist us in overcoming diverse difficulties encountered in the project and teamwork. We are grateful for Eddie for offering us technical support and guidance, and to Professor Catherine for offering us this opportunity for team collaboration. This experience has honed our abilities in multiple aspects, bringing us one step closer to becoming competent software engineers and an excellent development team.

10 Technical support

This system document is the edited in June 6 version 2.0. If you have any questions during testing or encounter any technical issues in the documentation, please contact `zihe.liu@ucdconnect.ie` For some functions, you can watch the video of final presentation, it can be a very good using instruction for this system.

Thank for your reading and testing —- All members of Group6 Clash in COMP3030J