

Department of Computer Science - The University of Bradford

## Requirements Specification Document

### Rakusen's Interactive, real-time visualisation dashboard

Team 17

Yahya Ashfak 23011246 [yashfak@bradford.ac.uk](mailto:yashfak@bradford.ac.uk)  
Adam Ahmed 23018946 [a.ahmed233@bradford.ac.uk](mailto:a.ahmed233@bradford.ac.uk)  
Barirah Irfan 23043808 [b.irfan@bradford.ac.uk](mailto:b.irfan@bradford.ac.uk)  
Saiful-Islam-Waheed 23018692 [s.waheed11@bradford.ac.uk](mailto:s.waheed11@bradford.ac.uk)  
Osman Talib 23017617 [o.talib@bradford.ac.uk](mailto:o.talib@bradford.ac.uk)  
Mohammed Ramzan Iqbal 23016117 [r.iqbal15@bradford.ac.uk](mailto:r.iqbal15@bradford.ac.uk)  
Harris Asghar Khan 23011221 [h.khan149@bradford.ac.uk](mailto:h.khan149@bradford.ac.uk)

#### Contents

1. Introduction.....	2
1.1 Project Brief.....	2
1.2 Team Expertise.....	2
1.3 Rationale of Topic Choice.....	2
2. Requirements .....	3
2.1 Functional Requirements.....	3
2.2 Non-Functional Requirements.....	4
2.3 Actors .....	5
2.4 Use Case Diagram.....	7
3. ERD Diagram .....	8
4. Data description .....	9
5. Class Diagram .....	10
6. Interface.....	11
7. Work Plan .....	14
8. Legal Social Ethical and Professional Issues .....	15
8.1 Legal Issues.....	15
8.2 Social Issues.....	15
8.3 Ethical Issues.....	15
8.4 Professional Issues.....	16
9. Peer Review.....	17
10. Code Implementation .....	18

# 1. Introduction

## 1.1 Project Brief

Rakusen's, a traditional food manufacturer based in Leeds, aims to enhance efficiency and adopt a more modernised, innovative approach to its processes through the integration of AI and Big Data technology. To support this initiative, an interactive, real-time visualisation dashboard will be developed to monitor the temperature data from sensors installed across two production lines. The insights and statistics derived from this system will be invaluable, enabling operators to fine-tune production temperatures with greater precision, thereby enhancing product quality and optimising energy usage.

The platform incorporates an API designed to access historical sensor data, enable time-based data filtering, and create simulated real-time sensor data. The user interface will be responsive, ensuring a smooth experience across desktops and tablet devices. Secure access is provided through a login system with email/password authentication and password reset functionality, with the API managing the authentication process.

Two distinct user roles will be supported: regular users (production operators) and admin users (managerial staff), each with tailored access privileges. The dashboard will provide dynamic visualisations, such as sensor charts and detailed statistical insights, allowing users to examine data aggregated by the sensors. A pre-trained Machine Learning model can then be integrated into the API's backend to forecast expected temperature readings for a specific timestamp and sensor. This will work in conjunction with a traffic-light system to actively highlight data points that deviate from the forecasted temperatures. Color-coded alerts will indicate anomalies based on the model's predictions, allowing for quick identification of significant variations from expected trends.

## 1.2 Team Expertise

Our team's extensive expertise enables us to design and develop Rakusen's visualisation dashboard with precision and efficiency. With strong background knowledge on software systems security, we understand the importance of securely handling data, ensuring that CSV files are safely imported into the database while maintaining integrity and protection. Our experience in designing well-structured system architectures, using UML and ERD diagrams, enables us to clearly define and map out the system's design. Our proficiency in languages such as HTML, CSS, and JavaScript means we can efficiently design a visually engaging web application. Our expertise in Python, enhances our ability to develop the back-end system incorporated with the machine learning model. This is further supported by our strong knowledge of SQL and PHP, ensuring efficient interaction between the API and the database. This combination ensures a seamless integration of front-end and back-end functionalities, resulting in a cohesive and high-performance platform. Moreover, our team's experience with a wide range of AI models ensures a smooth and efficient integration process, as we are well-equipped to comprehend and work with the provided pre-trained model.

## 1.3 Rationale of Topic Choice

Our team's approach is driven by a commitment to delivering a solution that not only meets Rakusen's technical requirements but also aligns with their goals of innovation and efficiency.

With our strong understanding of various AI models and experience working with databases, we are well-positioned to deliver a reliable and scalable dashboard that will support the company's transformation. By adopting a user-centred design approach, we aim to equip Rakusen's operators and managers with valuable data-driven insights that will enhance operational efficiency.

## 2. Requirements

*The following section outlines the functional and non-functional requirements for our project*

### 2.1 Functional Requirements

#### 1. Dashboard Display

- 1.1. Sensors should be organised logically for better readability and visualisation.
- 1.2. Display live sensor data and historical data.
- 1.3. Display filtering options for the user to apply on the historical sensor data.

#### 2. Display Statistics

- 2.1. When specific sensors are clicked on, specific sensor data should be displayed to the user.
- 2.2. Statistics include sensor charts, , minimum, maximum, average values, and anomalies for each sensor.

#### 3. Implement Signup Process

- 3.1. Upon registration, the user enters their email, temporary password assigned by the admin, password, and confirmation of password.
- 3.2. After completing registration, admins are alerted and must approve accounts to allow users to login

#### 4. Implement Login System

- 4.1. Login: Users should be able to log in using email and password after being approved by the admin.
- 4.2. Password Security: Implement a hashing technique to securely store passwords in the database.
- 4.3. Reset Password: Users should be provided with the option to reset their passwords if they have forgotten theirs.

#### 5. Define User Roles and Access Privileges

##### 5.1. Production Operators:

- 5.1.1. Must first register to access the system.
- 5.1.2. Can view historical data.
- 5.1.3. Can apply filters to refine sensor data displayed to them.
- 5.1.4. Can view live sensor data.
- 5.1.5. Can view statistical analysis for sensor data.
- 5.1.6. Can access the forgot password functionality.
- 5.1.7. Can access the user help guide.
- 5.1.8. Can apply light/dark mode.

##### 5.2. Production Managers:

Has the same access privileges as production operators but also have admin functionalities including:

- 5.2.1. Approving/declining new user registrations.
- 5.2.2. Optionally removing/adding sensors.
- 5.2.3. Assigning temporary passwords upon registration.
- 5.2.4. Edit details of production operators – this includes their names and temporary passwords
- 5.2.5. Resetting passwords for users who have forgotten their passwords.

## **6. API Development**

### **6.1. Interact with database:**

- 6.1.1. The system must be able to read and process historical sensor data from CSV files stored in the database.
- 6.1.2. The system will authenticate user logins by verifying the credentials and existence of the user stored in a separate database table, which includes emails and hashed passwords.

### **6.2. Simulate Real-Time Data:**

- 6.2.1. The API should simulate real-time data, either using synthetic data, replaying historical data, or implementing a random number generator. The most suitable approach will be selected based on its effectiveness for the system.
- 6.2.2. Simulated real-time data must be displayed and updated live on the dashboard at specific time intervals.
- 6.2.3. Once live data is displayed, it effectively becomes historical. Upon the next update, this data will be stored in the database and displayed as historical data.

**6.3. Filtering:** Allow filtering of data by specific time ranges to refine displayed data.

### **6.4. Implement Traffic Light System (Separate functionality):**

- 6.4.1. The Traffic Light System should be integrated as a separate functionality within the API.
- 6.4.2. Anomaly Detection: Flag sensor data points with a traffic light system (green, yellow, red) based on deviation from expected values.
- 6.4.3. Anomalies will be detected either using static thresholds derived from the statistical analysis of the sensor data, or by integrating the ML model.

## **7. Web Application Responsiveness**

- 7.1. Device Compatibility: The web application must be responsive and accessible on both desktop and tablet devices.

## **8. Machine Learning Model Integration (Optional)**

- 8.1. Predict Sensor Values: The provided pre-trained ML model will predict the expected sensor values for a given timestamp, with these predictions being used internally within the traffic light system. They will not be presented to the user as actual sensor data but will instead serve as a reference for system operations.
- 8.2. Integration: The model should be integrated into the backend via an API or Python code.

---

## **2.2 Non-Functional Requirements**

### **1. Light/Dark Mode (Optional)**

- 1.1. Mode Toggle: Implement a toggle to allow users to switch between light and dark modes.

### **2. Flexibility for Future Sensor Additions (Optional)**

- 2.1. Scalable Model: The system must be able to accommodate the addition of new sensors without requiring major changes or experiencing a drop in performance.

2.2. Dynamic Sensor Display: The frontend must dynamically adjust to new sensor types without hardcoded changes.

### **3. Performance**

3.1. Fast Response Times: The web application should load quickly and update sensor data in real-time, regularly and without delays.

### **4. Reliability**

4.1. Consistent Data: Simulated real-time data displayed on the dashboard should be consistent with the actual sensor readings and historical data.

### **5. Usability**

5.1. User-Friendly Interface: The application should have a clear, intuitive interface that is easy to navigate.

5.2. Accessibility: Ensure the application follows best practices for accessibility (e.g., colour contrast, font readability).

5.3. Help Sections: Provide user guidance through a help section to support new or less-technical users.

### **6. Maintainability**

6.1. Modular Codebase: The code should be modular and easily maintainable, with clear separation of concerns.

6.2. Flexible Configuration: The system should allow easy configuration and addition of new sensors or features without extensive rewrites.

6.3. Adherence to Coding Best Practices: Standard coding practices, such as using clear and descriptive variable names, should be consistently applied to ensure ease of collaboration.

### **7. Data Privacy**

7.1. Anonymity of User Data: Store only the necessary user data (e.g. hashed passwords) and avoid storing sensitive personal information.

## **2.3 Actors**

### **1. Production Operators:**

Production Operators will access the system after registering and receiving approval from the admin. They can view sensor data, including both real-time and historical data, as well as statistical analysis for individual sensors. They will be able to apply filters to refine the data based on time intervals. Operators can also access the help guide to understand the system's features and reset their passwords if necessary. Additionally, they can toggle between light and dark modes for better user experience. However, their privileges are limited, and they cannot add or remove sensors, nor approve new user registrations.

### **2. Production Managers:**

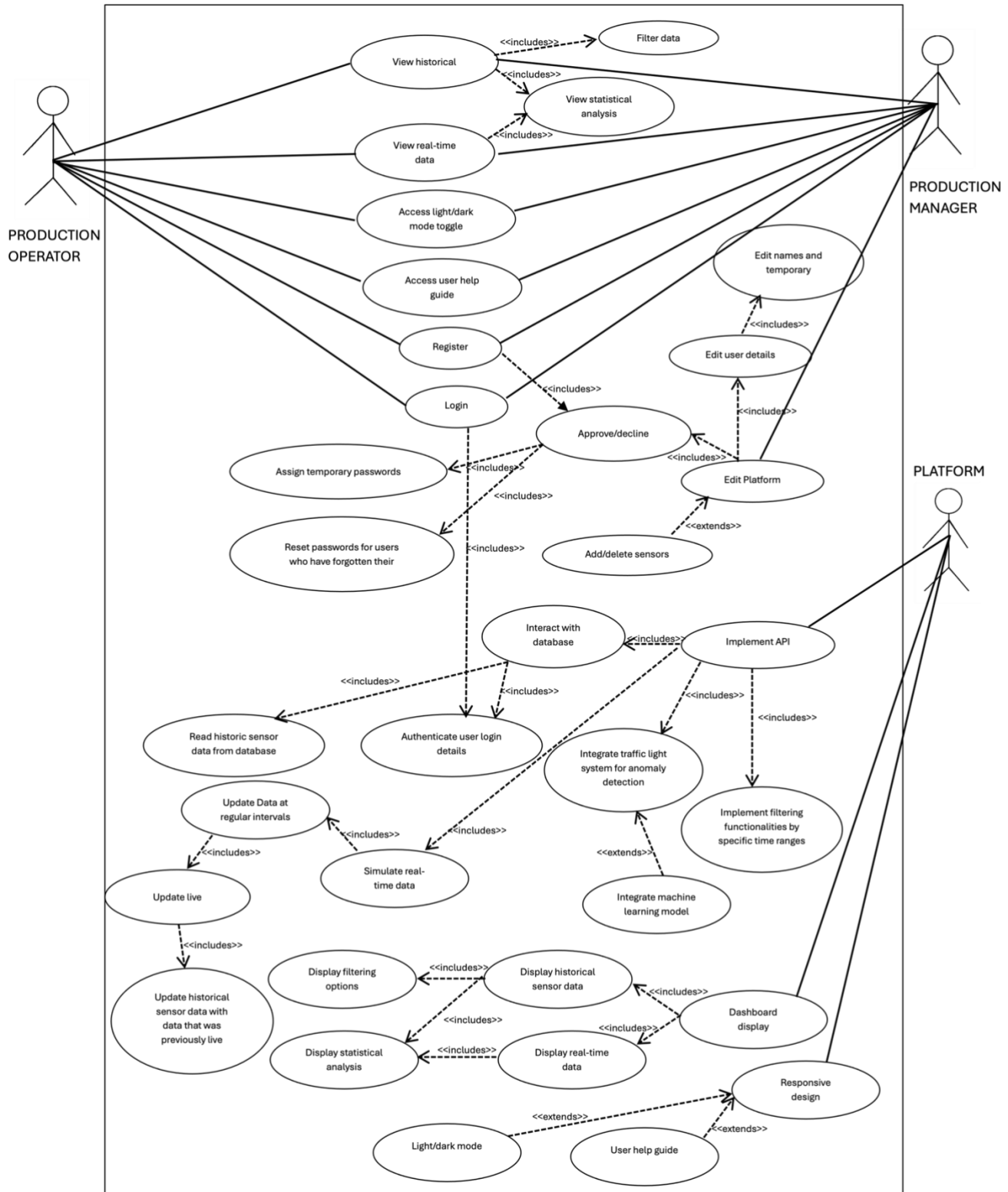
Production Managers have all the privileges of Production Operators, with the added responsibility of overseeing user registrations. They have the authority to approve or decline new user accounts and can also manage sensor configurations by adding or removing sensors, although this is not a mandatory functionality. Admins also can assign temporary passwords for

new users and reset passwords for existing users. Additionally, they can edit user details, such as their names, their account status and role. Their elevated privileges allow them to configure the system more flexibly while maintaining user access control, ensuring the system can be adapted to operational needs.

### 3. Platform (Web Application):

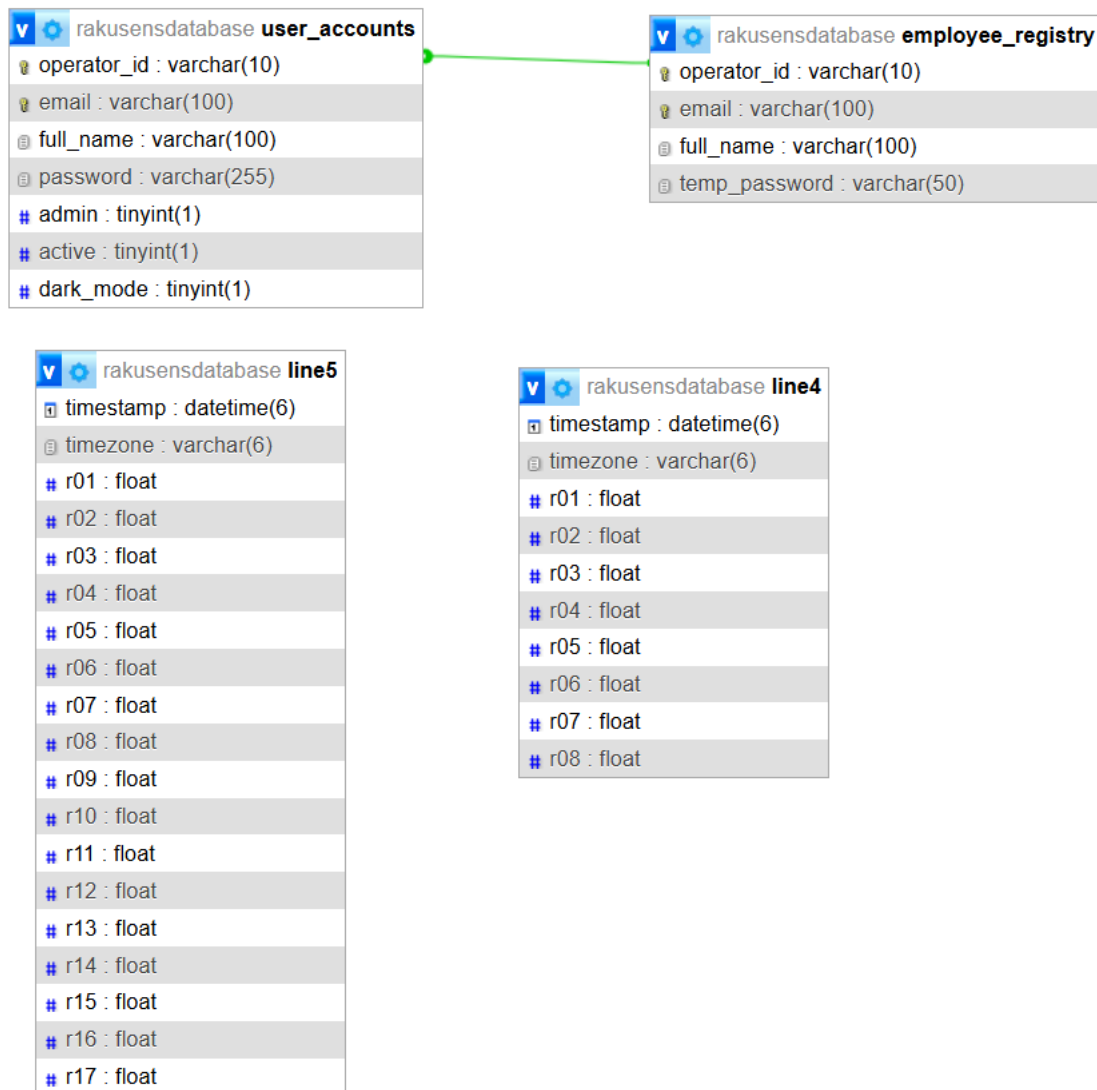
The platform is a user-friendly, web-based application that acts as the central hub for visualising and interacting with sensor data. It features a dynamic and responsive dashboard where users can view both live and historical data from sensors, along with comprehensive statistical analysis. The platform allows users to filter historical data based on specific time intervals to refine the information they see. It supports statistical visualisations as well as anomaly detection through a traffic light system, which flags data points based on deviations from expected values. Additionally, the platform is designed to be fully responsive, ensuring seamless access on both desktop and tablet devices. It also offers a light/dark mode toggle for user preference and a help guide to assist users in navigating the system. The platform communicates with the backend API to fetch and display real-time data, process filtering options, and update data (this includes both live data and historical data) at specified intervals. It ensures that users have a smooth and informative experience while interacting with the sensor data.

The following use case diagram depicts the systems core functionality from different actors' perspectives, illustrating their interactions with the different aspects of the dashboard



### 3. ERD Diagram

*The ERD diagram illustrates the relationships between key entities in the system, outlining how data flows and is organised within our database structure*





## 4. Data description

*This section provides a description of the sensor data across both production lines and the pre-trained machine learning model provided by the client, as well as the data within our database.*

notes.md – main documentation file – project requirements and specifications

- Data
  - raw\_uncleaned
    - line4.csv – Original, unprocessed sensor readings from production line 4
    - line5.csv – Original, unprocessed sensor readings from production line 5
    - line4.csv - processed sensor readings from production line 4
    - line5.csv - processed sensor readings from production line 5
  - machine-learning
    - example.html – html copy of example notebook of ran prophet model
    - example.ipynb
    - notes.md
    - requirements.txt
  - models
    - line4
      - prophet\_r01.pkl - prophet\_r08.pkl (8 model files)
    - line5
      - prophet\_r01.pkl - prophet\_r17.pkl (17 model files)

Sensor data files – line4.csv and line5.csv contain the temperature readings for 8 and 17 sensors respectively of active baking machines taken at roughly 30 second intervals.

### **SensorReading - Line 4 / line 5.csv**

timestamp: DATETIME - ISO 8601 format with 6-digit microsecond precision and timezone

temperature: FLOAT - Temperature reading in Celsius (°C)

sensor\_id: STRING(4) - Format: 'r01' through 'r17'

production\_line: STRING(5) - Format: 'line4' or 'line5'

### **ProphetModel**

model\_id: STRING(20) - Unique identifier for model – (prophet\_r08.pkl)

sensor\_id: STRING(4) - Format: 'r01' through 'r17'

production\_line: STRING(5) - Format: 'line4' or 'line5'

last\_updated: DATETIME - Timestamp of last model update

### **User**

employee\_id: STRING(8)

first\_name: VARCHAR(50)

last\_name: VARCHAR(50)

password\_hash: STRING(64) – Hashed password

email: VARCHAR(100)

status: ENUM - Values: 'ACTIVE', 'INACTIVE', 'PENDING'

contact\_number: STRING(15)

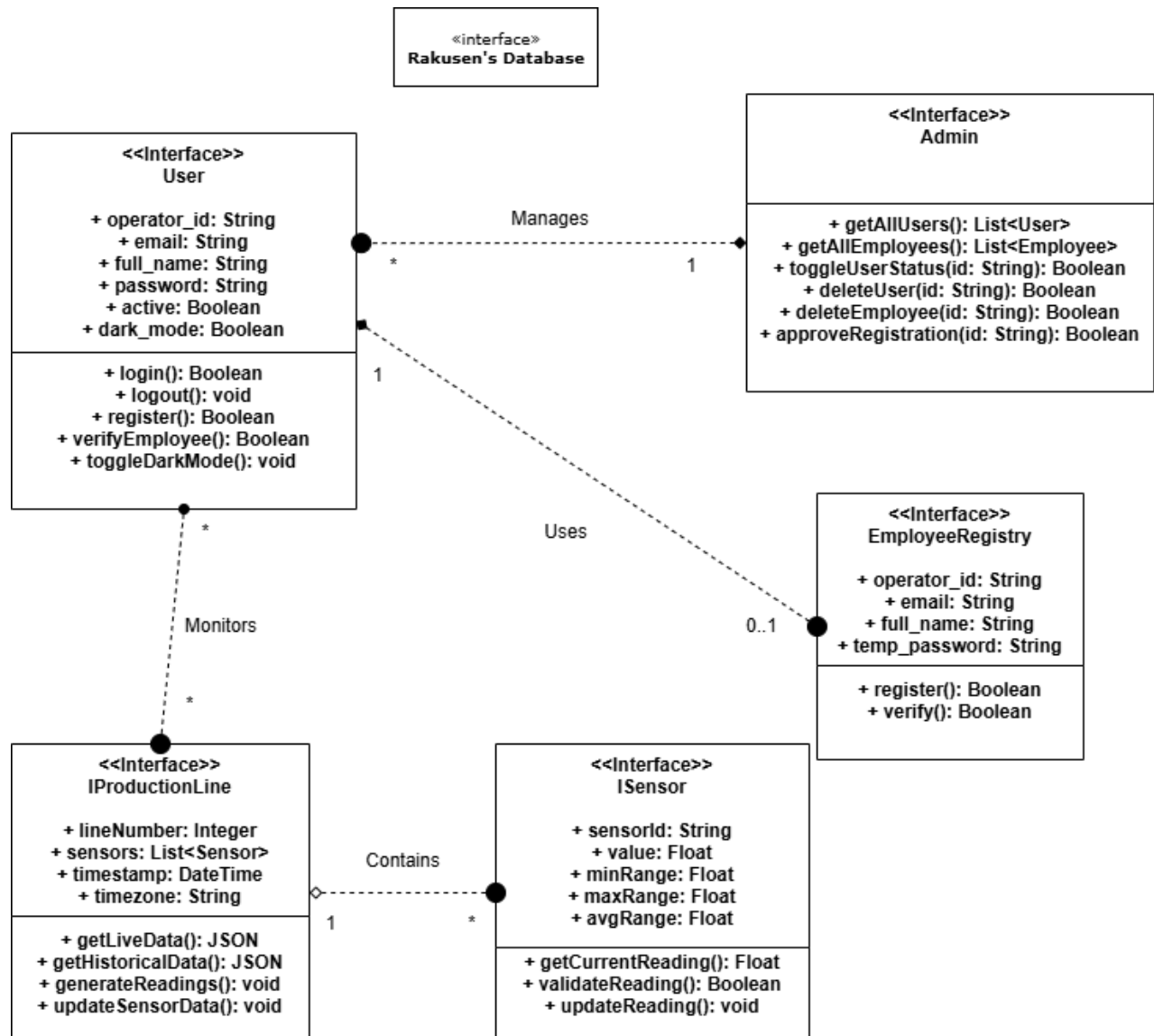
created\_by: STRING(8) - EmployeeID of approving admin

dark\_mode: BOOLEAN - True or false

access\_level: ENUM - Values: 'OPERATOR', 'ADMIN'

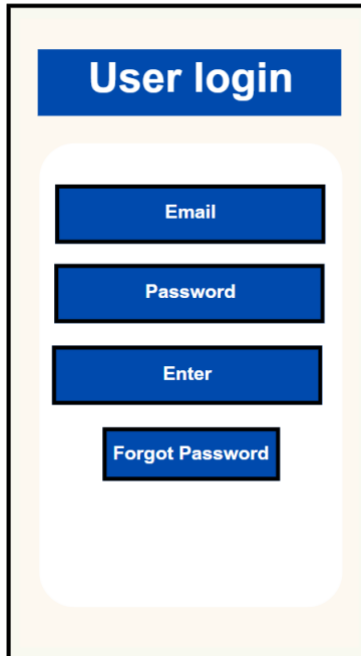
## 5. Class Diagram

The following class diagram illustrates the systems object-oriented structure showing the relationships between each component



## 6. Interface

*The following section showcases the user interface design through mock-ups of key pages, demonstrating user interactions with the dashboard*



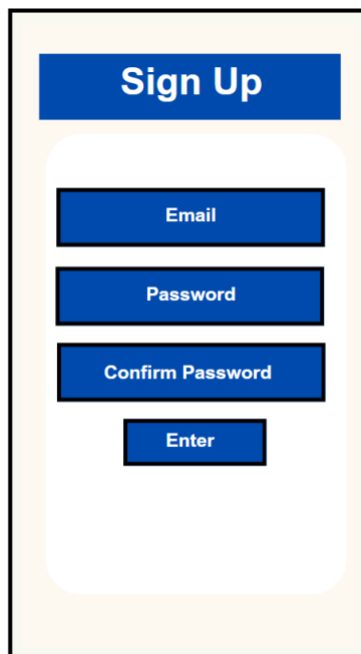
A mock-up of a user login page. It features a blue header with the text "User login". Below the header, there is a white rounded rectangle containing four blue buttons: "Email", "Password", "Enter", and "Forgot Password".

### User login page

When a user opens the website the login page is the first page they are greeted with. The page prompts the user to enter their email and password

If a user has not registered yet, they can choose to go to the signup page.

If a user forgets their password, they can prompt a forget password function which will alert an administrator to reset their password and change their account details accordingly



A mock-up of a sign up page. It features a blue header with the text "Sign Up". Below the header, there is a white rounded rectangle containing four blue buttons: "Email", "Password", "Confirm Password", and "Enter".

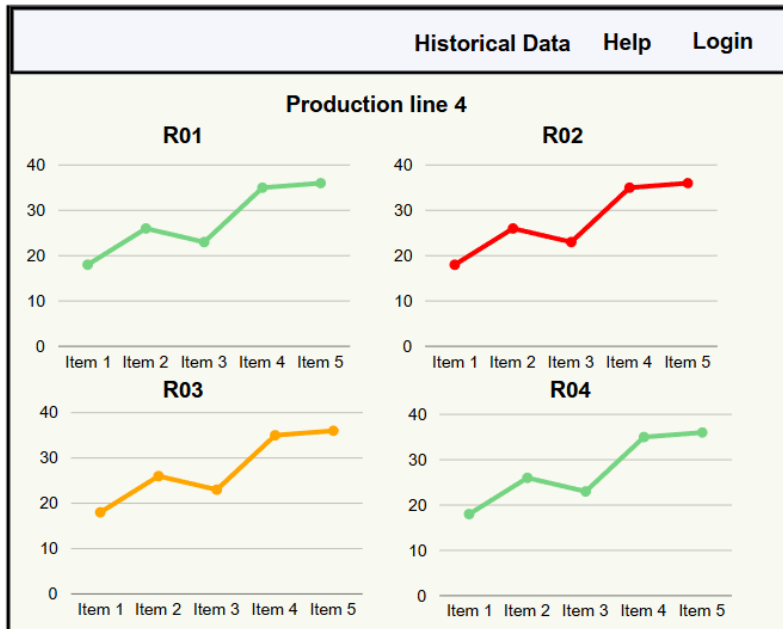
### Signup page

A user accesses this page from the login page if they have not yet signed up.

This page is a signup page but only works with set user details given by administrators

The user is prompted to create a new password to replace the preset details

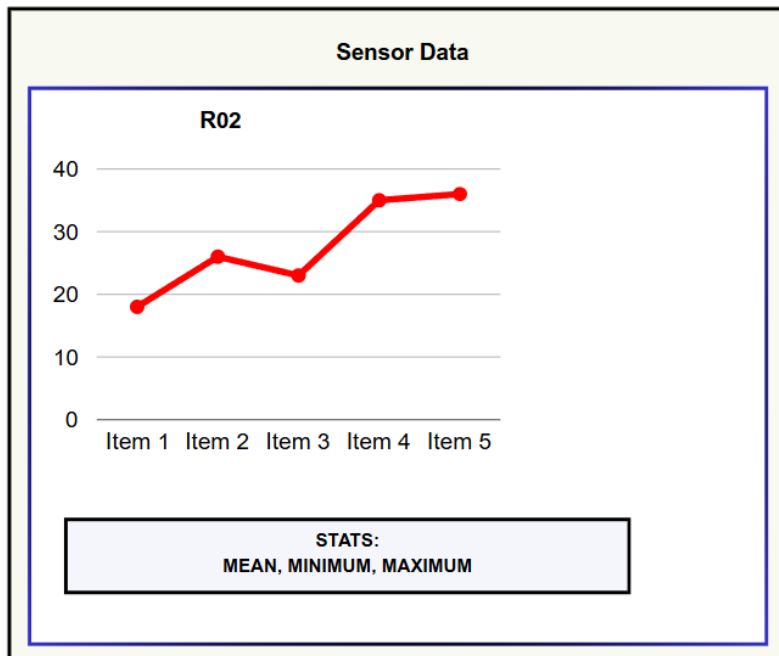
This process will alert an administrator and setup their account in a pending state where an admin will have to approve their account to allow them to log in



## Home page

This is the home page and where logged in users are directed to as the main page of work.

It provides a concise overview of the live temperature data for each sensor across both production lines.



## Sensor data page (single sensor)

This page is accessed from the home page or the historical data page

This page offers detailed information about a specific sensor, including visualisations such as graphs, charts and infographics. It presents key data points like minimum, maximum, and average values, along with any anomalous data.

Here, a more in detailed traffic light system will be displayed, showing the lower and upper bounds visually and the colour adapting dynamically based on the forecasted temperature.

live Data  Help  Login								
Filter								
timestamp	r01	r02	r03	r04	r05	r06	r07	r08

Historical data page

This page is accessed from the navigation bar

This page follows a similar format to the home page but includes a timeline feature, allowing users to filter and view data from specific points in the past based on their selected criteria.

## 7. Work Plan

The Gantt Chart below illustrates how coding tasks were distributed amongst team members, with progress being checked regularly – a copy is also provided in the [GitHub](#) repo /Documents

[illegible]

## 8. Legal Social Ethical and Professional Issues

*The following section outline the Legal, Social Ethical and Professional issues for our project*

There are several legal, social, ethical, and professional considerations that need to be accounted for in the development of this dashboard, ensuring it aligns with best practices and regulations while addressing potential risks.

### 8.1 Legal Issues

Whilst designing an internal system for company use, it is still important to comply with data privacy laws, such as the General Data Protection Regulation (GDPR) in the EU, ensuring that even minimal user data, like hashed passwords, is stored in a way that protects user privacy. Sensitive information should only be stored if explicitly necessary, and data should be retained for the minimum time required. Additionally, when using third-party libraries or software components (e.g., for the API or visualisations), it's crucial to adhere to licensing requirements and ensure proper attribution, particularly when using open-source tools. Ownership of the machine learning model and sensor data should be clearly defined, establishing who owns the collected sensor data and what rights users or employees have in terms of accessing or distributing it.

### 8.2 Social Issues

To ensure accessibility and inclusivity, the dashboard must comply with WCAG (Web Content Accessibility Guidelines), as it will be used by approximately 20 users with no technological background. This requires intuitive navigation, and accessible help pages to facilitate ease of use. A responsive design will ensure the application works seamlessly across various devices, although reliable internet access is essential for optimal performance. Transparency and accountability are critical, ensuring operators, managers, and users can trust the visualisations and data presented, with clear and accurate traffic light systems and anomaly detection to avoid misleading decisions. While the automation and AI-driven insights may reduce the need for manual monitoring, this is a long-term consideration for the company and falls outside the scope of our project.

### 8.3 Ethical Issues

Ethical considerations for this system revolve around ensuring that the pre-trained machine learning model is free from bias, especially in how data points are flagged as anomalous. While our project is focused on integrating the model as per the client's specifications, the company must prioritise validating future models with diverse datasets to mitigate any potential bias. For example, historical data used to train the model might reflect periods when anomalies were more or less common, which could lead to the model being biased toward detecting anomalies. Developing, training, and optimising these models to ensure they are unbiased and validated with varied datasets can be costly. This may involve acquiring additional data or hiring data scientists to fine-tune and validate the models, which could increase operational costs. However, by investing in this process, the company can enhance the model's accuracy and reliability, ensuring it performs well across different conditions.

Moreover, ethical concerns could arise if synthetic data does not adequately reflect real-world conditions, potentially undermining the system's decision-making reliability. Along with ensuring reliability, the system must prioritise the security of personal data. Personal

information, such as login credentials, must be securely stored and used solely for authentication. To enhance security, investments in advanced security protocols, such as two-factor or multi-factor authentication systems, as well as securing cloud storage solutions, may be necessary. These measures could lead to additional operational expenses and increased infrastructure costs. To maintain transparency and respect user autonomy, users should be clearly informed about the data being collected—such as email and password—and how it will be used. The company should also consider offering users the ability to manage their data preferences and request account deletion in the future.

Finally, while not the direct focus of our project, the system should be developed with energy consumption in mind. This approach ensures that the system contributes to ethical and environmentally responsible practices. Moreover, by implementing energy-efficient measures, the system will help reduce overall energy consumption and support the sustainable use of resources, ultimately leading to a lower environmental footprint over time. For example, the use of an AI-powered backend system could require increased computational resources, potentially increasing energy consumption and the environmental footprint. As the company aims to improve platform performance and scalability, it's important to consider this added energy demand. To mitigate such impacts, the company might explore solutions like energy-efficient hardware, optimising AI models for better performance, or offsetting emissions through sustainability programs.

## 8.4 Professional Issues

Quality assurance and testing are critical in this project, especially with real-time data updates and the integration of machine learning models. Professionals must thoroughly test and validate the application to prevent bugs or incorrect results. Additionally, continuous monitoring and maintenance will be necessary to ensure the dashboard remains reliable and accurate as the number of sensors and data points grows. Transparency in the machine learning models is equally important, as users need to trust the system's decisions. The anomaly detection process should be explainable rather than a black-box solution, and clear documentation about model training and limitations will be essential to maintain professional ethics. Furthermore, the system must be adaptable, allowing for the easy addition of new sensors and functionality without excessive difficulty. It should also be designed with scalability in mind, ensuring it can grow seamlessly without compromising performance or security.

*Below is a table summarising potential risks associated with the project and possible solutions to mitigate these risks within the scope of our work.*

Potential risks	A solution to tackle this
Unauthorised Access	Secure passwords with hashing algorithms, potentially implementing 2FA authentication for admin users
Non-working UI on some devices	Ensure responsive design with testing across different screen sizes and devices that will be used in the workplace
System downtime	If the local server implementation fails, consider using a reliable cloud server as an alternative. Ensure that redundancy and backup mechanisms are in place to maintain system reliability and prevent data loss.



User misinterpreting the data that is shown	Provide training and documentation which explains the data that is being visualised
---	---

## 9. Peer Review

Barirah Irfan (10/10)

- Barirah was responsible for developing the admin system and for compiling a comprehensive list of both functional and nonfunctional requirements for the system. She also identified and defined the relevant actors for the system. In addition to this, Barirah designed the use case diagram, to illustrate interactions between the users and the system.

Yahya Ashfak (10/10)

- Yahya was responsible for writing the data description and developing both the user signup and login pages. He also managed the project's GitHub repository, ensuring that the team's code and documentation was organised. Furthermore, Yahya designed the use class diagram to support understanding of the system's architecture.

Aadam Ahmed (10/10)

- Aadam was responsible for the LSEPi section of the requirements document. He also designed the live sensor/home page to ensure that real-time sensor data is generated at specific time intervals, presented smoothly and efficiently in our web application, and also stored within our database.

Mohammed Ramzan Iqbal (10/10)

- Ramzan contributed to the interface section of the report by explaining the system diagrams/functions. He was also responsible for compiling the meeting minutes after each team discussion. For coding, Ramzan designed the traffic light system for anomaly detection, and also designed code for how the provided machine learning model can be integrated within our system to produce a more refined traffic light system.

Osman Talib (10/10)

- Osman was responsible for writing the project brief, which outlined the scope and objectives of the project. He also designed the sensor data page for individual sensors, showcasing detailed statistical analysis for each sensor across both production lines.

Saif-Ul-Islam Waleed (10/10)

- Saif contributed to the overall interface design by designing the interface diagrams and collaborating with Osman to showcase the statistical analysis for individual sensors. He also ensured a smooth integration of this analysis with the backend, effectively connecting the front-end interface with the backend system for smooth data flow and functionality. Saif also worked on producing a machine learning integration method to refine the traffic light system.

Harris Asghar Khan (10/10)

- Harris was responsible for documenting the Team Expertise and Team Rationale and designed the Gantt chart to track project progress. Additionally, he designed the help page and the historical data page for the web application, ensuring a smooth database connection and proper display of sensor data for both production lines on the front-end.

## 10. Code Implementation

*Below is our team's GitHub repository link for our code and documentation*

<https://github.com/Yahyaaa-1/Team17>