Software Requirements Specification (SRS)

1. Introduction

1.1 Purpose

The purpose of this document is to define the requirements for the Interactive, Real-Time Visualization Dashboard for Rakusens. This system will provide a web-based solution to monitor and visualize sensor data in real time.

1.2 Scope

The system will include:

* A web-based dashboard for visualizing real-time and historical sensor data.
* Data aggregation and anomaly detection using an ML model.
* User authentication with admin, operator, and general user roles.
* Integration with a PostgreSQL database for historical data storage.
* Traffic-Light System for anomaly detection.
* Real-time alerts and notifications.

1.3 Team Expertise

Our team consists of individuals with expertise in:

* Front-end: HTML, CSS, JavaScript, PHP.
* Back-end: Python, Node.js.
* Database: SQL, PHPMyAdmin, XAMPP.

1.4 Rationale of Topic Choice

* Increase Product Quality – Enables operators to adjust temperatures in real-time.
* Enhance Operational Transparency – Allows users to view historical data and track trends.
* Integrate AI-Driven Anomalies – Uses machine learning to detect temperature deviations.
* Increase Energy Efficiency – Reduces excessive heating/cooling based on real-time detection.

1.5 References

* [Client-provided documentation] [To be filled]
* [Project proposal] [To be filled]

2. Functional (and Non-functional) Requirements

2.1 Actors

The Rakusens Dashboard (RD) will have two types of users:

* Administrator – Manages user accounts and system configurations.
* Operators – Monitor temperature data, receive real-time alerts, and make adjustments accordingly.

2.2 Functions and Their Description

User Authentication

* Login/logout for administrators and operators.
* User registration and authentication by administrators.
* Password reset functionality.

Real-Time Temperature Data Monitoring

* System retrieves real-time temperature readings from multiple sensors.
* Data is stored in an SQL database and accessed via an API.

Interactive Visualization

* Users can view temperature data through interactive graphs and dashboards (e.g., Plotly.js).
* Data is aggregated by sensor and displayed with historical trends.

Traffic-Light System for Anomalies

* ML model flags temperature anomalies using a color-coded system (green, yellow, red).
* Alerts are triggered for deviations from normal trends.

Real-time Alerts and Notifications

* Operators receive alerts via web notifications, emails, or SMS (optional).

Historical Data and Reports

* Users can generate reports on past temperature trends (CSV, PDF, Excel export options).

System Performance and Maintenance

* Admins can monitor system health, manage storage, and optimize data retrieval.

2.3 Non-functional Requirements

* Scalability – Handle increasing sensor data efficiently.
* Security – Implement encryption and role-based access control (RBAC).
* Cross-Platform Compatibility – Responsive design for desktop, tablet, and mobile.
* Performance – Real-time updates with minimal latency.

3. Data Description

3.1 Database Tables

1. Temperature Readings
   * Sensor ID (Primary Key)
   * Timestamp
   * Temperature Value
   * Status (Normal/Warning/Critical)
2. Users
   * User ID (Primary Key)
   * Username
   * Password (Encrypted)
   * Role (Admin/Operator/General User)
3. Alerts
   * Alert ID (Primary Key)
   * Sensor ID
   * Timestamp
   * Alert Level (Low/Medium/High)
   * Action Taken

4. Interface

4.1 Rakusens Dashboard Main Menu

* Administrator
* General User

4.2 Administrator Panel

* User Management (Add/Edit/Delete Users)
* System Settings
* View Reports
* Logout

4.3 General User Dashboard

* View Real-time Temperature Data
* Acknowledge Alerts
* Generate Reports
* Logout

5. LSEPI and Risk Assessment

5.1 Legal, Social, Ethical, and Professional Issues

* Data Privacy – Ensure GDPR compliance for user authentication and data storage.
* Security Risks – Prevent unauthorized access with encrypted credentials and/or multi-factor authentication (MFA).
* Operational Risks – Implement a robust backup strategy to prevent system failures that may affect temperature monitoring.

6. Work Plan

A detailed Gantt Chart will outline the project timeline, including milestones such as API development, front-end design, database setup, and system testing.

7. GitHub

* GitHub Repository for the Prototype – [https://github.com/Hedgeturd/ep-team-repo.git]
* GitHub for Meeting Minutes – [https://github.com/Hedgeturd/ep-team-repo/tree/9361881878c1c3f48f9e82044f8f8e7a1dd54cc0/team]
* GitHub for NDA – [https://github.com/Hedgeturd/ep-team-repo/blob/28ea73b79c7c3b3008ef5b638192885d31435fac/docs/NDA%20student-CEC%20-%20Ehtasham%20Afzal.pdf]

8. Peer Review

Each team member’s contribution will be evaluated based on assigned tasks, achievements, and effort, using a grading scale of 1 to 10 (where 1 = minimal contribution, 10 = outstanding contribution).