**Enterprise Pro-Team 11**

**Group Project Documentation and demonstration of software**

**Interactive, real-time visualisation dashboard**

**Names: UB:**

**Moman Ali 22007066**

**Armaan Hussain 23021587**

**Eisa Hussain 23011721**

**Ibrahim Modak 23017229**

**Mohammed Ihtisham Kayani 23035117**

**Wasi Abbas 23045106**

**Kamran Khan 23019070**

**Project Brief**

Rakusens is a traditional food manufacturer in Leeds. They are upgrading its production with AI and Big Data to stay ahead. As part of this plan, we’re developing an interactive, real-time dashboard to monitor temperature data from sensors across the production line. This tool will help operators quickly adjust temperatures based on data-driven insights, improving product quality and energy efficiency.

**The key features and Tasks**

**Real-Time Data Visualization**

* Display live temperature readings from multiple sensors.
* Update visualizations dynamically (e.g., every few seconds).

**Interactive Dashboard**

* User-friendly interface with charts/graphs (built with Plotly.js)
* Responsive design for desktop, tablet, and mobile use.

**Anomaly Detection**

* Flag unusual temperature trends using a traffic-light system (green/yellow/red).
* Integrate a pre-trained machine learning model to identify outliers automatically.

**Data Access & API**

* Build an API to fetch and parse temperature records from an SQL database.
* Ensure seamless data flow between the database and dashboard.

**User Authentication**

* Secure login system for operators.
* Password reset functionality for account management.

**Goals & Benefits:**

* Improve Production Quality which means operators can react faster to temperature fluctuations.
* Boost Energy Efficiency. This will reduce waste by optimizing heating/cooling processes.
* Replace Manual Checks as there will be automated data monitoring to save time and reduce errors.

**Technical Approach:**

* **Frontend:** JavaScript (Plotly.js), HTML/CSS, responsive frameworks (e.g., Bootstrap).
* **Backend:** API (Python/Node.js), SQL database integration.
* **AI/ML:** Pre-trained model for anomaly detection (provided externally).

By the end of this we shall have a simple but powerful tool that takes random sensor numbers and turns them into useful information making it easier for Rakusens to make production decisions based on reliable data instead of guessing.

**Project plan**

**Project plan**

1. **Project Objectives**

The main project goal consists of developing an industrial sensor monitoring system which tracks and analyses sensor data directly from the source in real-time. The system requires user access management for different roles while including machine learning models for anomaly prediction.

2. **Project Scope**

The system will make use of Flask to develop the backend which serves to ingest data and provide API services.

User interaction components will be designed through front-end elements built with HTML/CSS/JS plus PHP.

The project needs PHP development to establish user authorization functions next to role-based access systems.

The application will store data through SQLite at the local level.

Implementation of Prophet models should be integrated into the system for anomaly detection functions.

The development includes design aspects for dashboards that should function for both users and administrators.

The system accepts CSV files and reproduces live data interactions.

**3. Deliverables**

- Functional API for sensor data retrieval and simulation.

- User authentication system (register, login, logout).

- Admin panel for user management.

- Dashboard with live sensor data and prediction statuses.

- Machine learning integration with forecasting.

- Project documentation and diagrams.

Testing of CSV files alongside pre-trained models constitutes part of the workflow.

**4. Timeline**

Week 1-2: Requirements gathering and system design.

Week 3-4: Backend API development and database setup

Week 5: Frontend dashboard and UI integration

Users will gain authentication features and administrators will receive control systems throughout Week 6.

Week 7: Machine learning model integration

Week 8: Testing, optimization, and documentation

Week 9: Final review and deployment

**5. Team Roles**

The project manager oversees how the work progresses alongside achieving its predefined milestones.

The Backend Developer maintains responsibility for integrating the API alongside the database system along with the ML model.

A frontend developer implements the interface design together with UI/UX features while designing responsive visual components.

The tester executes three functions: quality assurance testing, validation checks and stress testing procedures.

The documentation lead creates project documents as well as prepares reports and diagrams.

**6. Tools & Technologies**

- Python 3.11

- PHP (for user management and session control)

- SQLite (lightweight database for sensor storage)

- HTML/CSS/JS (responsive UI design)

- Virtual Environment and Greenlet for compatibility

- Git for version control

**7. Risks and Mitigation**

Database locking can be achieved through using batch processing together with retry logic.

Model training alongside fallback systems should be properly executed to handle forecast errors.

User Interface performances should be improved through mobile-first CSS and testing.

User input must be validated before saving through any system because password strings should use cryptographic hashing methods.

- File format mismatch: Define strict CSV schema for ingestion

## **Coherent System and Data Description**

## 1. System Functionality Overview

## Your project is a start-to-finish as in sight manufacturing sensor monitoring platform that is made up of:

## - Real-time sensor data visualization

## - Anomaly detection with ML

## - User management and authentication

## - Responsive web interface

## 2. Sensor Data Handling

## Data Format (from CSV: line4.csv, line5.csv)\*\*

## The raw data returned from Line 4 and Line 5 sensor is in CSV format, for example:

## Line 4 CSV Structure:

## timestamp, r01, r02, ..., r08

## 2025-04-01 12:00:00.000, 15, 12, ..., 18

## Line 5 CSV Structure:

## timestamp, r01, r02, ..., r17

## 2025-04-01 12:00:00.000, 10, 13, ..., 19

## -These are batch loaded into the database (line4\_data, line5\_data).

## -Timestamps are cleaned, time zones are stripped off, and then parsed into datetime objects.

## -Both for speculative purposes in the past and seeding the ML prediction models.

## 3. Machine Learning: Predict & Status Classification

## Forecast Logic:

## -Prophet models predict expected values based on time series values.

## Downstream, the function forecast\_with\_prophet in model.py captures single point forecast.

## Status Logic:

## -All predicted values have been classified as:

## -Green: within threshold (10–20)

## -Red: out-of-threshold

## -Amber: edge/fallback

## Model Files:

## - 25 Prophet models (8 Line 4, 17 Line 5)

## - Loaded via at runtime.

## - The compiled .pyc file.

## 4. Environment & Dependencies

## Virtual Environment:

## -Created with Python 3.11.3

## -Probably has flask

## Greenlet Integration:

## -Submodule of greenlet package. Used for coroutine switching.

## -Supports async frameworks

## 5. Full-Stack Architecture

## Frontend: HTML, CSS, JS – responsive UI with forms, navbar, dashboards

## Backend API: Flask – routing, ML, CSV ingest.

## ML: Prophet – time-series anomaly detection

## Database: SQLite – sensor data storage

## User Auth: PHP – login/register system with session and admin panel

## Deployment: Localhost

## 6. User Roles & Experience

## Normal User:

## - Registers, logs in

## - Redirected to user home

## - Views dashboards

## Admin:

## - Access to admin page and admin users

## - Edit user accounts and authorize data from sensor

## Wrap-Up

## This system is a robust smart and well architecture monitoring platform for smart manufacture or industrial IoT. It offers:

## - Real-time + historical tracking

## - Machine learning-based anomaly alerts

## - Admin/user segregation

## - Modular, extensible backend

## - Clean, responsive frontend design

System Requirements and Specifications

1. Coherent Overview

This project is, industrial monitoring system, full stack, capture, and analyse data from 2 manufacturing lines. It provides real time, and historical data to view, using machine learning abstraction reporting anomalies, user identity authentication, role-based dashboards as well as administrative features of access control.

2. Functional Requirements

- Read sensor data from the CSV files of Line 4 and Line 5 to SQLite.

- Create restful API endpoints to retrieve pagination data with the optional filtering by date.

- Use existing records, increment times and generate synthetic, real-time sensor data by sampling.

- Use of Prophet-based ML for time-series forecast per sensor.

- Choice Predictions Vs Threshold to get status (Green, Red, Amber).

- Display sensor values and status in interactive dashboard.

- Allow user registration, user login, user logout and role redirection (user vs admin).

- Give admin view registered users and managing it.

3. Non-Functional Requirements

- Performance: The system should be able to recognize and make predictions very quickly.

- Scalability: Handles inflating data sets for the batch in addition to optimized queries.

- Reliability: Does retry on database locks and is nice to failed predictions.

- Usability: Clean, intuitive, mobile-responsive user interface with clear visual feedback.

- Security: Uses session-based authentication. Passwords should be password hashed.

- Maintainability: Separated role and function-based python and php code base.

- Compatibility: Built with Python 3.11, Flask, Prophet, PHP, and SQLite.

4. System Constraints

- SQLite is being used which is file based and might have limitation of superior performance for production environments

- ML models (Prophet) need the signature pre-trained files and enough historical data.

- The current sensor data is required to be conform to predefined CSV format.

- Frontend has only html/css/js compatibility issues.

- Real time simulation comes from the random data, not real hardware integration.

5. Specifications Summary

- Database: SQLite

- Backend: Flask, PHP

- ML Library: Prophecy for forecasting, for model save/ load

- UI: HTML, CSS, JS

- Data Input: Time stamped sensor read between CSV file.

External Admins

Interface

Admins

Operator

Interface Design and Functional Correlation

1. Login & Registration Interfaces

Functional Purpose:

- Authenticate users securely.

- Allow new user registration.

Interface Features:

-Clean Css form layout with input labels (username, email, password).

- Blue-based UI with rounded box styling and mouse on effects.

- Inline Non-Valid Submission alerts.

- Links login/register.

Functional Correlation:

- Automatically voted with PHP scripts (login\_form.php, register\_form.php).

- In submit, credentials are checked against the records in user\_db.sql

- Session creation and redirection logic (admin\_page.php or user\_home.php).

2. Dashboard Interface (User View)

Functional Purpose:

- Show real time sensor from Line 4 and Line 5.

- Display real-time anomality status (green, red, amber).

- Allow historical data retrieval.

Interface Features:

- Centrally located cards or charts.

- Color-coded indicators for the prediction status.

- Navigation bar with user greeting and logout button.

- Media queries using styles.css (responsive design is implemented on page zoom.

Functional Correlation:

- Fetch sensor data from Flask API /api/data.

- Data is processed and laid out via JavaScript/HTML.

- Response prediction status from model.py.

- Periodically or on user request refreshes.

3. Admin Panel Interface

Functional Purpose:

- Give access to user management.

- make view of all registered user, and account role

- Admin-only control panel interface.

A screenshot of a login form

AI-generated content may be incorrect.

A screenshot of a login form

AI-generated content may be incorrect.#A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a calendar

AI-generated content may be incorrect.

**Team minutes**

|  |  |  |  |
| --- | --- | --- | --- |
| Date-Duration | Participants | Contents/task | Progress |
| 13.03.25- team meeting- 31 minutes | Armaan Hussain  Eisa Hussain  Ibrahim Modak  Wasi Abbas  Kamran Khan  Moman Ali  Ihtisham Kayani | After the prototype demonstration, we received feedback on what was missing from the prototype. As a group we assigned tasks to each other using the feedback that was given. Ibrahim was given the role to add filtering to the dashboard. Kamran was given the task to add the traffic light system. Armaan,Wasi and Moman was given the task to do the unit testing and acceptance testing. Ihtisham and Eisa was given the task to help with the filtering and traffic light system alongside completing the workplan. | We made a deadline for these tasks to be completed in 1-2 weeks so the testing can be done. |
| 21.03.25- Team Meeting- 27 minutes | Armaan Hussain  Eisa Hussain  Ibrahim Modak  Wasi Abbas  Kamran Khan  Moman Ali  Ihtisham Kayani | In this group meeting each person shared their screen and showed what progress was made in the previous week. | The traffic light system was nearly completed and needed 3-4 days for it to be fully completed. Ibrahim managed to complete the filtering. Armaan and Moman and Wasi managed to do acceptance testing on the features that are working. Ihtisham and Eisa completed the work plan and helped with the traffic light system and filtering. Next week we are hoping to have everything completed so all the testing can be finished. |
| 28.03.25- Team Minutes- 30 minutes | Armaan Hussain  Eisa Hussain  Ibrahim Modak  Wasi Abbas  Kamran Khan  Moman Ali  Ihtisham Kayani | Everyone attended this meeting and completed the tasks that were given to them apart from some of the tests. In this meeting we spoke about what needs to be finished and what could have been improved in the next few days. | Traffic light system was completed but had a flaw where only green and red was showing and the filtering was completed. The progress that needs to be made for the next few days is the remaining tests and traffic light system to be fully working where it shows amber alongside with green and red. |
| 02.04.25- In person meetup- 60 minutes | Armaan Hussain  Eisa Hussain  Ibrahim Modak  Wasi Abbas  Kamran Khan  Moman Ali  Ihtisham Kayani | We met up in person and had everything completed in full working order and practised the demonstration and ensured everything was working without any issues. We ensured it works because on the day of prototype demonstrations there was issues loading the code and dashboard. | As a group we keep going through the work to see if anything else needs to be added and agreed everything is at a high standard. |

**Peer review**

**Peer review:**

**Muhammad Armaan Hussain: Login Form and testing**

Armaan was responsible for designing and implementing the login form for the dashboard portal, ensuring it was both secure and user-friendly. He also conducted thorough unit and acceptance testing on the dashboard, playing a key role in identifying and resolving bugs to meet the project’s requirements. His contributions were consistent and technically solid, significantly enhancing the stability and usability of the system. Based on his involvement and performance, Armaan is awarded a contribution score of 9 out of 10.

**Moman Ali: Login form, registration form, testing**

Moman Ali was responsible for developing the registration form for the dashboard portal and conducting testing to ensure its proper functionality. He successfully created a user-friendly and accessible registration interface that integrated well with the rest of the system. His testing efforts contributed to identifying and resolving key issues, improving the overall user experience. Based on his contributions and performance, Moman Ali is awarded a contribution score of 9 out of 10.

**Kamran Khan: LSEP, Dashboard, Admin user page and traffic light system**

Kamran Khan contributed significantly to the project by working on the LSEP integration, the main dashboard, the admin user management page, and the traffic light system. He played a key role in designing and implementing the overall dashboard layout, ensuring real-time data was effectively presented. His work on the admin user page allowed for smooth user role management and secure access controls. Additionally, Kamran developed the traffic light system to visually represent system status and anomalies, which enhanced the user experience and monitoring capabilities. His contributions were technically strong and essential to the core functionality of the project. Based on his efforts and impact, Kamran Khan is awarded a contribution score of 9.5 out of 10.

**Wassi Abass: UML, Testing and code justification**

Wassi Abass was responsible for creating UML diagrams, conducting testing, and providing code justification throughout the development process. His UML work helped clarify the system architecture and workflow, supporting effective planning and communication among team members. He also contributed to testing by identifying issues and validating functionality across multiple components. Additionally, Wassi provided clear and thorough code justifications, making the logic and structure of the implementation easy to understand for both developers and reviewers. His work added clarity and quality assurance to the project. Based on his contributions, Wassi Abass is awarded a contribution score of 8 out of 10.

**Eisa Hussain: Admin Panel, traffic light system**

Eisa Hussain contributed to the development of the admin panel and the traffic light system for the project. He played a key role in implementing features within the admin panel that allowed for effective user and system management, enhancing administrative control and usability. Eisa also worked on the traffic light system, helping to design and integrate a real-time visual indicator for system status and anomalies. His contributions improved both the functionality and user experience of the dashboard. Based on his work and involvement, Eisa Hussain is awarded a contribution score of 8 out of 10.

**Ihtisham Kayani: Data Simulator, workplan, filtering for dashboard**

Ihtisham Kayani was responsible for developing the data simulator, creating the project workplan, and implementing filtering functionality for the dashboard. His data simulator played a crucial role in generating synthetic real-time data for testing and validating dashboard features. He also prepared the workplan, helping the team stay organised and on track with clear task distribution and timelines. Additionally, Ihtisham implemented data filtering features in the dashboard, allowing users to view and analyse data more efficiently. His contributions were essential for both planning and technical functionality. Based on his input and performance, Ihtisham Kayani is awarded a contribution score of 9 out of 10.

**Ibrahim Modak: registration form, filtering for dashboard**

Ibrahim Modak contributed to the development of the registration form and implemented filtering functionality for the dashboard. He assisted in building a user-friendly registration process that aligned with the overall system design, ensuring a smooth onboarding experience. His work on the dashboard filtering allowed users to narrow down and interact with data more effectively, improving usability and performance. Ibrahim’s efforts supported key areas of the user interface and data handling. Based on his contributions, Ibrahim Modak is awarded a contribution score of 9 out of 10.