TECHNICAL DOCUMENT

Version 1.0

Project Code: EBIW\_Industry-4.0\_2022

Document Version: 1

Date: **3/1/2022**

Tracking of changes

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Changes** |  | **Verified By** | **Validated by** |
| 1.0 |  | EBIW/ | Initial release |  | EBIW/ | -- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Related Documents

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version** | **Serial**  **no** | **Author** | **Document name** | **Date** | **Change Done By** |
| 1.0 |  |  |  |  | -- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

| ***Name*** | ***Roles*** |
| --- | --- |
| Mainul Alam Badhon | Full stack web implement |
| Rafiul Ahmed | AWS Services |
| Ashutosh Kumar Roy | Anomaly and aws sagemaker |
|  |  |
|  |  |
|  |  |
| Dhriti Majumdar | QA & Technical Document |
| Md. Shahriar Alam | QA & Technical Document |

Contents

[**1** **INTRODUCTION** 4](#_Toc98174809)

[***1.1*** **PURPOSE** 4](#_Toc98174810)

[***1.2*** **TARGET AUDIENCE** 4](#_Toc98174811)

[***1.3*** **SPECIFIC OBJECTIVES** 4](#_Toc98174812)

[***1.4*** **SCOPE** 5](#_Toc98174813)

[***1.5*** **GLOSSARY OF TERMS** 5](#_Toc98174814)

[**2** **GENERAL DESCRIPTION** 6](#_Toc98174815)

[**3** **MODULES** 7](#_Toc98174816)

[***3.1*** **Oracle Machine Learning** 7](#_Toc98174817)

[***3.2*** **MQTT** 7](#_Toc98174818)

[***3.3*** **Oracle autonomous database** 7](#_Toc98174819)

[***3.4*** **Node.js** 7](#_Toc98174820)

[***3.5*** **Streaming Data Sensors** 8](#_Toc98174821)

[***3.6*** **Flask API** 8](#_Toc98174822)

[***3.7*** **Socket.IO** 8](#_Toc98174823)

[**4** **Design of The Solution** 9](#_Toc98174824)

[***4.1*** **Backend Project Architecture** 9](#_Toc98174825)

[***4.2*** **Database Diagrams** 9](#_Toc98174826)

[**4.2.1** **Table Structure** 10](#_Toc98174827)

[**4.2.2** **Sample Table** 10](#_Toc98174828)

[**5** **Sequence Diagram** 11](#_Toc98174829)

# **INTRODUCTION**

## **PURPOSE**

The purpose of this document is to develop the detailed design of the requirements requested by Project Industry 4.0.

This document provides a description of the technical design for Industry -4.0. This document’s primary purpose is to describe the technical vision for how business requirements will be realized. This document provides an architectural overview of the system to depict different aspects of the system. This document also functions as a foundational reference point for developers. Please note that this is a baseline document and may be updated as development progresses.

Industry-4.0 Solution from eBIW enables industry intelligence through industry 4.0 IOT that facilities customers in

Achieving:

* + Increased Industry Productivity.
  + Improved Industry Quality.
  + Reduced Operational Cost.
  + Reduced Industry Machine Downtime.
  + Implementing smart systems.

## **TARGET AUDIENCE**

This document is targeted to welding companies.

* Development Team
* IT Management
* Support Staff

## **SPECIFIC OBJECTIVES**

Achieve a successful and satisfactory working product of the Interactive interview platform. Comply in scope, time and cost, with the project planning. Provide the organization with the desired platform.

## **SCOPE**

* Implementation of login and logout module with proper authentication.
* Implementation of postgres db for the database of the solution through AWS EC2.
* Implementation of AWS EC2 for the deployment of the solution.
* Implementation of node.js.
* Implementation of socket.IO.
* Implementation of Amazon Kinesis DataStream service.
* Implementation of Amazon Sagemaker Machine Learning service.
* Implementation flask API for data fetching and posting.
* Implementation of JavaScript.
* Implementation of html, CSS, JavaScript for Interactive graphical dashboard.

## **GLOSSARY OF TERMS**

The following terms are used in this document:

|  |  |
| --- | --- |
| Term | Definition |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# **GENERAL DESCRIPTION**

The advent of industry 4.0 and smart manufacturing has raised a demand for consistent and visible quality of production. The basic manual welding is a largely unattended area where quality and production control is not automated. The coming age of industrial IoT and various communication technologies has helped and design a cost-efficient tool for quality, productivity, costing and condition monitoring of the inverter-based welding machine.

Industry-4.0 is an electronic device to value the make. The inverter-based welding machine IoT enabled. It is installed on the inverter end of the machine and captures the operating parameters from the welding machine; performs edge intelligence and then transmits the information to the cloud using established IoT protocol. Welding machines provided along with control limits are settable on the machine from the cloud. Data is available in real time in the cloud where a dashboard and reports are provided for productivity, quality, costing and condition monitoring.

Productivity is very important as the industry involves high cost and can be considered as a large-scale industry. Therefore, time and budget should be kept minimal to avoid loss to the companies involved. An example of lack of productivity in the industry is there are many complaints in the industry that welders do not perform their job on time and their quality of job differs on a huge scale, proper training is required based on welder performance. Therefore, a monitoring system is required that can be used to monitor these welding stations. This system is important because it can help supervisors track the welding works from afar or anywhere using the internet of things (IoT). To achieve that, a system must consist of hardware and software that are capable of connecting to the internet and monitor the welding works. A product is required that can be used for data processing, can connect to the internet, monitor voltage, current, gas-flow, RPM, etc. using sensors to detect different parameters of the welding machine and a dashboard to show the processed data and can be monitored throughout the globe. Other than that, this system should be able to warn the welder and track the welder based on their quality of job on the welding machine. Thus, the system should solve the problem of welders not performing their job on time. Supervisors should also be able to monitor the job of welders to ensure maximum productivity. A product should work as a welding station monitoring system able to detect welding usage, measure voltage, current, RPM, Gas-flow, and other values of welding and send the data through IoT for monitoring.

Our Industry-4.0 device is used to help the business users to get a detailed insight from the welding machines with industry defined KPIs and dashboards. It helps to have a check on the critical control parameters and alert before any sort of untoward incident happens.

# **MODULES**

## **Oracle Machine Learning**

With Oracle Machine Learning, Oracle moves the algorithms to the data. Oracle runs machine learning within the database, where the data reside. This approach minimizes or eliminates data movement, achieves scalability, preserves data security, and accelerates time-to-model deployment.

## **MQTT**

MQTT is an OASIS standard messaging protocol for the Internet of Things (IoT). It is designed as an extremely lightweight publish/subscribe messaging transport that is ideal for connecting remote devices with a small code footprint and minimal network bandwidth. MQTT today is used in a wide variety of industries, such as automotive, manufacturing, telecommunications, oil and gas, etc.

## **Oracle autonomous database**

Autonomous Data Warehouse is a cloud database service optimized for analytical processing. It automatically scales compute and storage, delivers fast query performance, and requires no database administration.

An autonomous database leverages AI and machine learning to provide full, end-to-end automation for provisioning, security, updates, availability, performance, change management, and error prevention.

## **Node.js**

Node.js is an [open-source](https://en.wikipedia.org/wiki/Open-source_software), [cross platform](https://en.wikipedia.org/wiki/Cross-platform), [backend](https://en.wikipedia.org/wiki/Front_end_and_back_end) [JavaScript](https://en.wikipedia.org/wiki/JavaScript) [runtime environment](https://en.wikipedia.org/wiki/Runtime_system) that runs on the [V8 engine](https://en.wikipedia.org/wiki/V8_(JavaScript_engine)) and executes JavaScript code outside a [web browser](https://en.wikipedia.org/wiki/Web_browser). Node.js lets developers use JavaScript to write command line tools and for [server-side scripting](https://en.wikipedia.org/wiki/Server-side_scripting) running scripts server-side to produce [dynamic web page](https://en.wikipedia.org/wiki/Dynamic_web_page) content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying [web-application](https://en.wikipedia.org/wiki/Web_application) development around a single programming language, rather than different languages for server-side and client-side scripts.

Node.js has an [event-driven architecture](https://en.wikipedia.org/wiki/Event-driven_architecture) capable of [asynchronous I/O](https://en.wikipedia.org/wiki/Asynchronous_I/O). These design choices aim to optimize [throughput](https://en.wikipedia.org/wiki/Throughput) and [scalability](https://en.wikipedia.org/wiki/Scalability) in web applications with many input/output operations, as well as for [real-time Web](https://en.wikipedia.org/wiki/Real-time_Web) applications.

Node.js is calculating UCL, LCL, operations and also transferring data to Sagemaker for performing the respective operations.

## **Streaming Data Sensors**

The data which we are getting from different sensors (real-time data) for example temperature sensor, pressure sensor, voltage sensor etc. Further additional sensors can also be used for expansion i.e., IOT sensors.

## **Flask API**

Flask-RESTful is an extension for Flask that adds support for quickly building REST APIs. It is a lightweight abstraction that works with your existing ORM/libraries. Flask-RESTful encourages best practices with minimal setup. If you are familiar with Flask, Flask-RESTful should be easy to pick up.

Through flask api anomaly score, id, kpi name, lcl, ucl data is being passed into the database. When data is called in the frontend through flask api socket io and js.client passes the data to flask api which is then shown in the live dashboard. All the communication part is done by flask api.

## **Socket.IO**

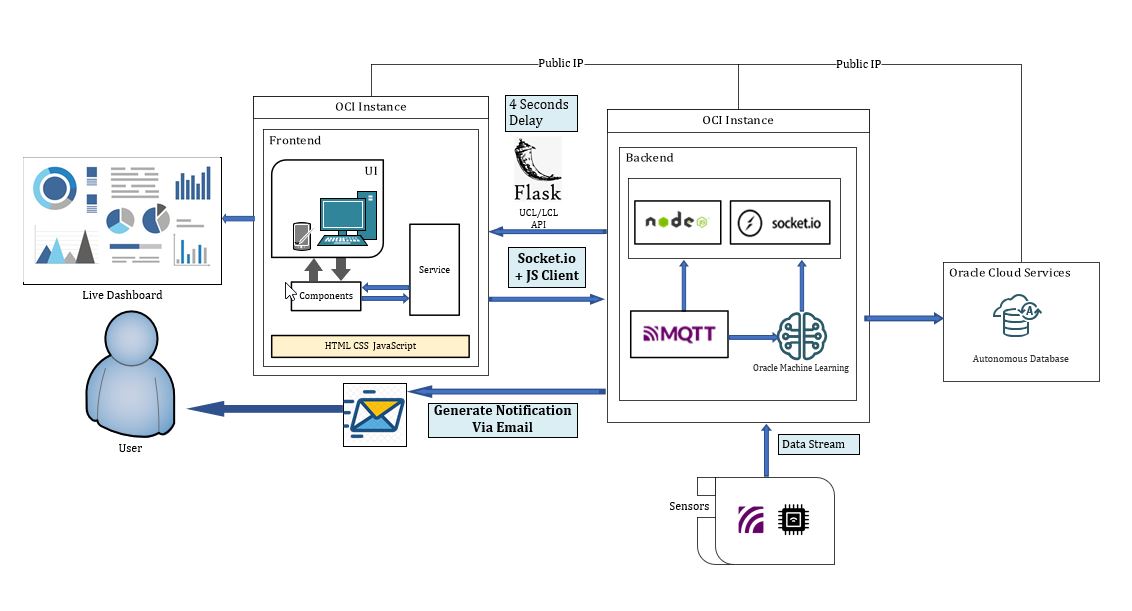
Socket.IO enables real-time bidirectional event-based communication. It consists of:

* a Node.js server (this repository)
* a [JavaScript client library](https://github.com/socketio/socket.io-client) for the browser (or a Node.js client)

The UI based connection is done by socket io. The live stream data is done by js client.

# **Design of The Solution**

## **Backend Project Architecture**



## **Database Diagrams**

### **Table Structure**

### **Sample Table**

# **Sequence Diagram**