University of Colombo School of Computing

SCS4123 - Final Year Project in Software Engineering

Academic Year 2017 – 2018

# Hitech Smart Factory

A generalized software solution for automated production line monitoring


**Authors**

V.D. Liyanage (Index No: 13000659)

H.M.P.M. Karunarathne (Index No: 13000561)

D.D. Mathangaweera (Index No: 13000764)

**Supervisor**

Dr. K.L. Jayarathne


Submitted in support of the degree of

Bachelor of Science (Hons) in Software Engineering

Submitted on 2nd of January 2018

# Declaration

We certify that this dissertation does not incorporate, without acknowledgment, any material previously submitted for a degree or diploma in any university and to the best of our knowledge and belief, it does not contain any material previously published or written by another person or ourselves except where due reference is made in the text. We also hereby give consent for our dissertation, if accepted, be made available for photocopying and for interlibrary loans, and for the title and abstract to be made available to outside organizations.

| Candidate Name | Signature |
|---|---|
| V.D. Liyanage | |
| H.M.P.M. Karunarathne | |
| D.D. Mathangaweera | |

Date: 01/02/2018

This is to certify that this dissertation is based on the work of Mr. V.D. Liyanage, Ms. H.M.P.M. Karunarathne and Ms. D.D. Mathangaweera under my supervision. The thesis has been prepared according to the format stipulated and is of an acceptable standard.

Supervisor Name: Dr. K.L. Jayarathne

……. ……..

Signature of Supervisor Date

# Abstract

Mass production of goods with automated production lines is a commonly used approach today to fulfill the high demand. Automation reduces production time, increases accuracy and eliminates human errors. However, controlling and obtaining the highest throughput from these automated production lines require constant monitoring. Lack of monitoring can result in business failures as well as huge profit losses. Using a software solution for this monitoring process is way beneficial than using human labor.

In this dissertation, a generalized software solution is proposed for monitoring automated production lines, which can be integrated with existing production lines without doing any hardware modifications. To achieve the generalization, a novel approach to production line modeling is being introduced. As the initial step, the proposed solution, Hitech Smart Factory provides a mechanism to model the existing production lines, and that model is then used in further functionalities. Hitech Smart Factory uses sensor data read through PLC units as inputs in order to facilitate the integration of the system into an existing production line without doing hardware modifications.

Users can view the status of the sensors in real time. A data processing platform is used to process and store real-time data to match with various business requirements. Reports can also be generated using these processed data.

Evaluations carried out for the proposed solutions shows that Hitech Smart Factory is capable of meeting the identified requirements to an above satisfactory level. The system is capable of presenting real-time data values in its graphical components within a latency of two seconds. Non-functional requirements such as modularity, Reusability and scalability are also achieved. Promising future study areas have also identified which can be used for further tuning of the proposed solution.

# Acknowledgement

We would like to express our sincere gratitude to our project supervisor, Dr. K.L. Jayarathne, senior lecturer of University of Colombo School of Computing for providing us continuous guidance and supervision throughout the project.

We would also like to extend our sincere gratitude to Mr. Nuwan Pallewela, a senior software engineer at Sysco Labs (Pvt) Ltd and Mr. Buddhika Marasinghe, Director / Engineer of Hitech Solutions Pvt Ltd for the guidance and support provided throughout the year. Furthermore, our sincere appreciation goes to DR. Mindika Premachandra, lecturer of University of Colombo School of Computing and Mr. W.V. Welgama, senior lecturer of University of Colombo School of Computing for providing valuable feedback on our project to improve the outcome. We also take this opportunity to acknowledge the assistance provided by Dr. Manjusri I.E. Wickramasinghe as the final year software engineering project coordinator.

We appreciate the feedback and motivation provided by our friends to achieve our project goals. This thesis is also dedicated to our loving families who has been an immense support to us throughout this journey of life. It is a great pleasure for us to acknowledge the assistance and contribution of all the people who helped us to complete our project.

# Table of Contents

# List of Figures

# List of Acronyms

| | |
|---|---|
| CRUD | Create, Retrieve, Update, Delete |
| RDBMS | Relational Database Management System |
| WSO2 DSS | WSO2 Data Services Server |
| WSO2 IS | WSO2 Identity Server |
| XML | Extensible Markup Language |

# Glossary

**Lambda Architecture** - Lambda architecture is a generic, scalable and fault-tolerant data processing architecture designed for big data handling. It supports both batch and stream (real-time) data processing and works as a cost-effective solution for handling massive quantities of data. Lambda architecture is divided into three processing layers as the batch layer, serving layer, and speed layer. The speed layer uses most recent data for real-time (stream) processing and the batch layer maintains the master dataset for non-real-time processing. Views get created separately on batch as well as real-time data and the serving layer is responsible for merging the views created on batch and real-time layers. [25]

**Modbus protocol** - Modbus or Modbus RTU is a serial communication protocol published by Modicon (now Schneider Electric) in 1979. Initially, they developed this protocol to use with their programmable logic controllers. In abstract, Modbus is a communication protocol designed to transmit information over serial lines. Modbus architecture contains two main components as Modbus Master, which is the device requesting information and the Modbus Slave who supplies the information. In a typical network, there is only one Modbus Master Node present and up to 247 Slaves can be connected. [26]

**PLC** - A programmable logic controller is an industrial digital computer which has been adapted for the control of manufacturing processes, in a rugged environment. These are used to control the machinery in automated production lines. They are pre-programmed to make decisions based on the inputs, mostly the sensors. [16]

**POST** - A request method supported by the HTTP protocol of the World Wide Web. By design, the POST request method expects that a web server accepts the data enclosed in the body of the request message. These data can be an arbitrary amount of any type. A header field in a POST request usually indicates the Internet media type of the message body. [15]

**RESTful web service** - REpresentational State Transfer (REST), is an architectural pattern which provides interoperability between computer systems on the Internet.

Roy Fielding defined REST in his dissertation "Architectural Styles and the Design of Network-based Software Architectures" written for his Ph.D. in 2000. RESTful web service is a web service which follows REST architecture. They are lightweight, highly scalable and maintainable and also very commonly used to create APIs for web-based applications. [17]

**SCADA** - A system of software and hardware elements that allows industrial organizations to maintain efficiency, process data for smarter decisions, and communicate system issues to help mitigate downtime. The basic SCADA architecture begins with programmable logic controllers (PLCs) or remote terminal units (RTUs). PLCs and RTUs are microcomputers that communicate with an array of objects such as factory machines, HMIs, sensors, and end devices, and then route the information from those objects to computers with SCADA software. The SCADA software processes, distributes and displays the data, helping operators and other employees analyze the data and make important decisions. [20]

# Chapter 1

# Introduction

In correspondence with the present economic system of the world, most goods are manufactured in mass quantities. Factories are the establishments where this mass production happens. Most of the time, manufacturing goods only with human labor is not sufficient due to the high demand. Therefore, most of the production companies lean towards introducing machinery equipment to replace human workers to speed up the production processes. Introduction of these machinery is known as automation.

However, replacing humans with machinery does not necessarily solve all problems, it instead emerges new issues to the surface. Designing of the equipment is not straightforward since every detail of the target process has to be understood beforehand. Besides these new systems run at high speeds and volumes making them quite tricky to control. According to Dr. Don Norman, the director of The Design Lab at University of California, San Diego, *"automation is at an intermediate level of intelligence, powerful enough to take over control that used to be done by people, but not powerful enough to handle all abnormalities"* [7]. To tackle these new issues properly, careful monitoring of that machinery is necessary.

## 1.1 Motivation

The world may face the problem of rapidly weakening population growth in coming years due to low birth rate and low life expectancy. Hence, the proportion of people of working age is decreasing while the relative number of those retired is expanding. In order to face this population dispute, most of the jobs will be automated in the future. This will result in an era of new automated production companies emerging more often as well as existing companies getting larger by the day.

When considering a large-scale production company which has several automated production lines, it is necessary to monitor the status of each production line from a centralized viewpoint. Most primary reason for the argument is the fact that without knowing the present status, it is impossible to improve. As a company, it is crucial to

know what is happening in each part of the production process in order to make better decisions.

However, most companies tend to neglect this fact due to various reasons such as high cost of implementing a customized solution, need of modifying or replacing the existing machinery, need of installing new machinery, etc. Not having a methodology to view and monitor real-time data leads to massive amount of profit losses and business failures every day.

As a leading company in automation industry of Sri Lanka, Hitech Solutions (Pvt) Ltd grasp a vast and up to date knowledge on the essential requirements of the domain. While working with over one hundred clients in Sri Lanka, they have identified the tremendous need for a monitoring solution for automated production companies. Even though there are several automation monitoring solutions exist, they have found out that those solutions are hard to apply in Sri Lankan context due to various practical issues. The Director of Hitech Solutions (Pvt) Ltd, Mr. Buddhika Marasinghe emphasized that it will be a huge benefit to have a low-cost automation monitoring solution as a large number of small and medium scale production companies tend to automate their productions day by day.

As Mr. Nuwan Pallewela, Senior Software Engineer and Sysco Labs, said, this project has a promising future in the Sri Lankan automation industry, simply because the existing solutions do not suit for the Sri Lankan context.

## 1.2 Problem Definition

In an era where automation is applied to produce almost everything consumed by the world, controlling such processes and improving is not straightforward. Monitoring those machinery carefully can be seen as a confident approach towards the issue. This will aid in understanding the inner workings of the automated production lines in question and results in better planning and throughput. However, monitoring such automated production lines are as complex as designing those machinery themselves.

## 1.3 Project Goal and Objectives

The primary goal of this project is to provide a cost-effective and generalized monitoring solution which can be applied to any automated production line instantly without doing hardware modifications to the existing system.

The objectives of this research project can be identified as follows.

- Developing of a graphical tool for modeling and representing production lines.
- Developing a web-based tool for production line monitoring.
- Developing a mobile application for production line monitoring.
- Developing a web application to view analyzed data on the system.

## 1.4 Methodology

The Initial step was to study existing similar systems and related research areas to get an overall idea of the domain. Points gathered from this study will be discussed in Chapter 2 in detail. As the next step, a solution is designed considering the ideas acquired from the background study and the requirements gathered.

When considering an automated production line, it has several machinery components which are controlled by one or more PLC units. PLC units use sensors to collect data which represent the statuses of the machines and change the behavior of those machines accordingly. Hence ideally sensors are the minimal units of status representatives, a production line can have. Therefore, Hitech Smart Factory uses the sensor data read by PLC units to represent the status of the machines in a production line.

These data are used to fulfill two purposes. They are directly used to show the real-time status of the sensors to the users, meanwhile streaming through a processing engine which processes these real-time data and stores them in meaningful ways for long-term uses including report generation and maintenance.

The primary challenge was to develop a generalized solution which can be applied to any production line without doing any hardware modification. Accordingly, a novel approach is being introduced where users can model existing production lines using a toolset provided by the system as the first step and then use that model for further activities such as pushing data to databases, showing real-time status and report generation. Design and implementation consideration taken into account will be discussed in detail in Chapter 3 and Chapter 4 respectively.

## 1.5 Scope

As mentioned in Section 1.4, Hitech Smart Factory uses the sensor data read through PLC units. The software component used for this purpose is called Data Publisher module, which will be further elaborated in Subsection 3.5.1. Since the project is aimed at any automated production line in the industry, there can be different brands and makes of PLC units involved in reading sensor data with different communication protocols.

Concern arose in first project defenses about the feasibility of implementing support for all PLC units available in the market as mentioned above. Hence it is decided to build the product focusing a single PLC unit in this context, yet preserve the capability of supporting various units in the future.

Accordingly, a XINJE branded XC3 series PLC was used with the model number of XC3-60RT-E[1], considering the availability of the hardware. A commercial view of this PLC is shown in Figure 1.1. XC is a general PLC series that has features such as Support two kinds of program languages, Rich essential functions, Offset function (Indirect addressing), Single phase or AB high-speed counter, etc. XC3 is the standard type variation of the XC series that is capable of fulfilling most industry requirements identified in the present [25]. According to Mr. Buddhika Marasinghe, Director / Engineer of Hitech Solutions (Pvt) Ltd., XC3 series PLCs are used with high popularity within the Sri Lankan automation industry.

[1] http://www.xinje.com/en/ProductView.asp?ID=114&SortID=143

***Figure 1.1**: A Commercial View of XC3-60RT-E PLC*

## 1.6 Outline of the Dissertation

The dissertation is structured as follows. Chapter 2 explores similar systems and related research areas to automated production line monitoring. Chapter 3 describes the design of the proposed solution, Hitech Smart Factory. Chapter 4 demonstrates the implementation details of the design elaborated in Chapter 3. Chapter 5 presents the evaluation model and the evaluation results of Hitech Smart Factory. Last chapter, Chapter 6, presents the summary of the dissertation and the project along with possible future extensions.

## 1.7 Summary

This chapter laid the foundations for the dissertation. It introduced the project, motivation behind the project, problem definition and the goals and the objectives. Then, the methodology was briefly described, project scope is specified and the dissertation was outlined. On these foundations, the dissertation will proceed with a detailed description of the project. The next chapter will discuss the background study.

# Chapter 2

# Background Study

## 2.1 Introduction

In this chapter, a review of related work on automated production line monitoring is provided. There are some products available that have tried to address the issue in various manners. The first part of the chapter describes those products, their strengths and weaknesses. Then points taken into consideration by studying those similar systems are discussed. Next part of the chapter reviews research areas that were proved to be useful in developing a monitoring system, especially in handling a vast amount of data.

## 2.2 Existing Solutions

### 2.2.1 WimFactory by UlalaLabs

WimFactory, an IoT smart factory platform, using smart sensors called 'WICON' which can apply to production facilities under any environment with a smooth and efficient application [22] [9]. The new sensor module introduced with WimFactory, which are called WICON is capable of measuring various data in a factory such as coolant input and temperature, volume of mold, temperature of emissions, and humidity. According to customer needs, UlalaLabs are willing to provide other sensors such as magnetism force, vibration, light, acceleration, distance, pressure, etc. as well. Once installed in a factory, WICON sensors measure data in real-time and send them to the cloud server for big data analysis. According to UlalaLabs, WimFactory can be installed in a facility within 15 to 25 days.

The main issue that was identified in WimFactory is the need of purchasing and installing third-party sensor modules to an existing system.

### 2.2.2 WinTr by FulTek

By definition, WinTr is an advanced SCADA software framework developed for monitoring and saving data of manufacturing processes [10]. By applying WinTr, devices managed from a single station can be connected with several PLCs using several protocols the software supports such as Profinet (S7 1200), Modbus RTU, Modbus TCP/IP, etc. Historical data related to processes are saved into the databases and report generation is also included in the package. WinTr is developed using .NET, C# and VBScript and uses SQL server 2005 as the database. [6]

One of the main issues that were identified was the complexity. WinTr is not a complete solution but a framework for developing monitoring systems. Everything has to be designed and setup using the tools provided by a person who has an excellent knowledge of the software. Another issue was the difficulty of getting support and maintenance since the company is located in Turkey.

### 2.2.3 Status Enterprise by B-SCADA

B-Scada's Status Enterprise is a modern, information model based SCADA system designed to connect plant floor processes with the rest of the enterprise. It aggregates and organizes data from local or geographically dispersed assets and systems into a consolidated, well-organized information model for alarming, archiving, analytics and real-time visualization [5].

Functionalities provided:

- Real-time and Historical Data
- Logging
- Calculations
- Reporting
- Form Design
- User Roles and Workspaces
- Web and Mobile Access

One of the main issues identified with this system was the intricate architecture of the system. It acts as a counter element in applying modifications and maintaining the

software. Another issue is the difficulty of obtaining support and maintenance since the company is located in the USA.

After studying existing attempts to address the issue, several ideas were taken into consideration for the design of the proposed solution. It is noticed that instead of developing a standalone solution similar to WinTr by Fultek, a cloud-based product similar to WimFactory would result in high availability, ease of access and ease of maintenance. Although reading sensor data using existing components such as PLC units were identified as a better solution rather than introducing new equipment that needs to be installed, similar to the work done by WimFactory with their WICON sensors.

When considering the automation industry of Sri Lanka, a fully developed system that is capable of plugging into a factory would serve the clients better than a framework which needs further modifications or adjustments. Hence the idea of such framework similar to the WinTr system is disregarded. Providing a mobile application to view the statuses of production lines on the go, similar to the Status Enterprise by B-SCADA, was seen as a viable option considering the availability of smartphones in the present. Other points that were taken into consideration from Status Enterprise were to store both real-time and historical data in system databases and to have user roles for ease of user management.

## 2.3 Related Research Areas

### 2.3.1 Big Data Handling

Hitech Smart Factory can be considered as a system belongs to SCADA architecture and it is required to handle a vast amount of real-time and historical data. Therefore, a study was conducted to determine the optimal approach to handle big data.

The research paper “Algorithm and Approaches to Handle Big Data” by Shafaque, Uzma, and Parag D. Thakare[23] reviews various algorithms and approaches such as clustering and Apriori algorithm, decision tree algorithm, random forest algorithm and K-means algorithm necessary for handling big data and describes methods of different

approaches used to handle such large datasets [23]. Moreover, it gives an overview of architectures and algorithms used in large data sets.

The research paper “Big data analysis using Hadoop cluster” by A. Saldhi, D. Yadav, D. Saksena, A. Goel, A. Saldhi and S. Indu describes techniques and technologies to store, distribute, manage and analyze large-sized data sets with high-velocity [4]. Big data can be structured, unstructured or semi-structured, resulting in incapabilities of conventional data management methods. Data are generated from various sources and can arrive in the system at various rates. In order to process these large amounts of data in an inexpensive and efficient way, parallelism is used. Hadoop is the core platform for structuring Big Data and solves the problem of making it useful for analytical purposes. Hadoop is an open source software project that enables the distributed processing of large datasets with a very high degree of fault tolerance.

### 2.3.2 Big Data Processing Architectures

The data generated by Hitech Smart Factory belongs to the category of big data. Several data processing architectures are used today by the professionals to process big data, such as Lambda architecture, Kappa architecture and Zeta architecture. Therefore, another study was conducted on these architectures to determine the optimal big data processing architecture for SCADA systems.

The research paper “Lambda Architecture for Cost-effective Batch and Speed Big Data processing” by M. Kiran, P. Murphy, I. Monga, J. Dugan and S. S. Baveja is about Lambda Architecture for cost-effective batch and speed big data processing [14]. This paper presents Lambda Architecture as a software design pattern which unifies online and batch processing within a single framework. The pattern is suited for applications where there are time delays in data collection and availability through dashboards, requiring data validity for online processing as it arrives. The pattern also allows for batch processing for old data sets to find behavioral patterns as per user needs.

The research paper “Real-time stream processing for Big Data” by Wolfram Wingerath, Felix Gessert, Steffen Friedrich, and Norbert Ritter describes that Kappa Architecture employs a powerful stream processor capable of coping with data at a far higher rate than it is incoming and a scalable streaming system for data retention [24]. An example

of such streaming system is Apache Kafka [3] which has been specially designed to work with the stream processor Samza[1]. Kappa is a simplification of Lambda Architecture with the batch processing system removed [12]. Although Kappa architecture does not offer both batch and real-time processing it can handle real-time data processing very efficiently with a single stream processing engine. Accordingly, the Kappa architecture is used in Hitech Smart Factory in order to process massive quantities of real-time sensor data.

## 2.4 Summary

This chapter mainly focused on providing a review of the existing similar systems and the related research areas. The main functionalities provided by the existing solutions were discussed along with the issues which make it difficult to adapt those solutions into the Sri Lankan context. The ideas inferred from those solutions and from the research papers which were published on the related research areas are also discussed. The next chapter will elaborate the design of Hitech Smart factory.

[1] http://samza.apache.org/

# Chapter 3

# Analysis and Design

## 3.1 introduction

This chapter explicates the proposed solution to production line monitoring, Hitech Smart Factory. At the beginning of the chapter, identified functional and non-functional requirements are described. After comes the technical requirements in Section 3.4 and the proposed system design in Section 3.5. Design of the system is described by discussing each separate architectural component one by one.

## 3.2 Functional Requirements

### 3.2.1 Login/Logout

In Hitech Smart Factory all the activities should only be performed by authorized users. Hence all the users are required to log in to the system providing their usernames and passwords.

### 3.2.2 Production line modeling

The automated production lines differ from factory to factory. They can have different brands and makes of machinery and sensors. Since Hitech Smart Factory is a generalized solution, it requires a mechanism of representing those production lines. Although a production line consists of a large number of sensors, all of those sensors are not required to view the status of the production line. Hence the production line should be represented using only the sensors which are preferred by the engineering and managerial staff of the factory.

Above process of modeling the production line, is an essential task in Hitech Smart factory. Therefore it can be performed by a system administrator (system admin) or a factory administrator (factory admin) only. This process consists of following subtasks.

- Create/ Update/ View/ Delete factory
- Create/ Update/ View/ Delete branch
- Create/ Update/ View/ Delete section
- Create/ Update/ View/ Delete production line
- Model production line

### 3.2.3 View real-time status

Machine operators and engineers of a factory need to view the real-time status of the sensors and components to make sure that the machines are functioning as expected. This functionality is provided in both web application and the mobile application.

### 3.2.4 User Management

A software system typically comprises with various users who can access various parts of the system. Apart from the administrative users of Hitech Smart Factory, all other users are divided into 4 levels.

- A **factory level user** can view all the branches, sections and the production lines of the factory he/she is given access to.
- A **branch level user** can view all the sections and the production lines of the branches he/she is given access to.
- A **section level user** can view all the production lines of the sections he/she is given access to.
- A **production line level** user can view all the production lines he/she is given access to.

Managing these user levels are done by factory admin. This includes adding, deleting, updating and viewing users.

### 3.2.5 Generate reports

Reports are required by managerial staff in order to take managerial decisions such as formulate future production plans, identifying production bottlenecks etc. Engineers require reports for maintenance purposes. These reports can be generated by using the data in Hitech Smart Factory databases.

The use case diagrams of the system are available in Appendix A.1.

## 3.3 Non- functional Requirements

### 3.3.1 Security

In Hitech Smart Factory user authentication will be done through WSO2 Identity Server (WSO2 IS) instance. It includes multi-layer security functionalities as well as features such as Role Based Access Control, which are used in Hitech Smart Factory to ensure the security. Login is required to perform any task within the system for every user. Furthermore, Role Based Access Control is used to manage the different privileges and functionalities of different users within the system.

### 3.3.2 Modularity and Modifiability

Hitech Smart Factory is developed following a modular architecture. Every single component of the system is independent of each other and message passing techniques are used to glue together these loosely coupled units. The main advantage of this architecture is the ability to modify any component at any given time without breaking down the whole system.

As discussed in Section 1.5, the current system has been developed to support XINJE XC3-60RT-E PLC unit. However, with the aid of the modular architecture of Hitech Smart Factory, supporting for another PLC unit is quite simple. Only the Data Publisher module needs to be modified in order to support the requirements of the new PLC. Data Publisher module and its unique implementation details regard to the XC3-60RT-E PLC will be further elaborated in Subsection 4.2.1.

### 3.3.3 Reusability

Modules developed in Hitech Smart Factory are designed to be reused in applicable instances. For example, Data Publisher module is designed to take properties of PLC unit which is used to read the sensor data, as inputs by property files. The particular Data Publisher developed for this context is capable of reusing for all PLC units that

support Modbus RTU protocol. More details about the implementation of Data Publisher module are discussed in Subsection 4.2.1.

### 3.3.3 Scalability

Scaling of a deployed product instance from monitoring small-scale production lines to large-scale production lines have been made easier with the modular architecture of Hitech Smart Factory. Since each component is loosely coupled, each of them can be scaled by adding more resources (typically hardware) separately, depending on the load. With high standard technologies used in each critical component of the system such as influxDB and Kafka, even with its default configurations, Hitech Smart Factory is capable of handling a noticeable load.

### 3.3.5 Easy Integration

As mentioned in Section 1.4, Hitech Smart Factory uses PLC units to get sensor data. Since most of the production lines in the industry already use PLC units, no hardware modifications are needed to integrate Hitech Smart Factory into production lines of a factory.

## 3.4 Technical Requirements

This section summarizes the technologies used to develop Hitech Smart Factory.

- Programming Languages - J2EE, Javascript, Python
- Frameworks - React Native, Spring, Kafka
- Databases - MySQL, InfluxDB
- Servers - Tomcat
- Package Managers - npm, maven
- Platforms - Docker, rancher
- Libraries - JQuery, mxGraph

## 3.5 System Design

The system design will be explained using the system architecture diagram, which is shown in Figure 3.1**.** Descriptions of the components in the diagram will be provided in this section with the intention of serving a high-level idea about each of them. These components will be further elaborated in detail in Section 4.2 below with implementation details. Blue colored lines in the diagram indicate how the user interactions flow and black colored dashed lines indicate data flows. Yellow colored components represent user interfaces while green colored components represent backend servers.

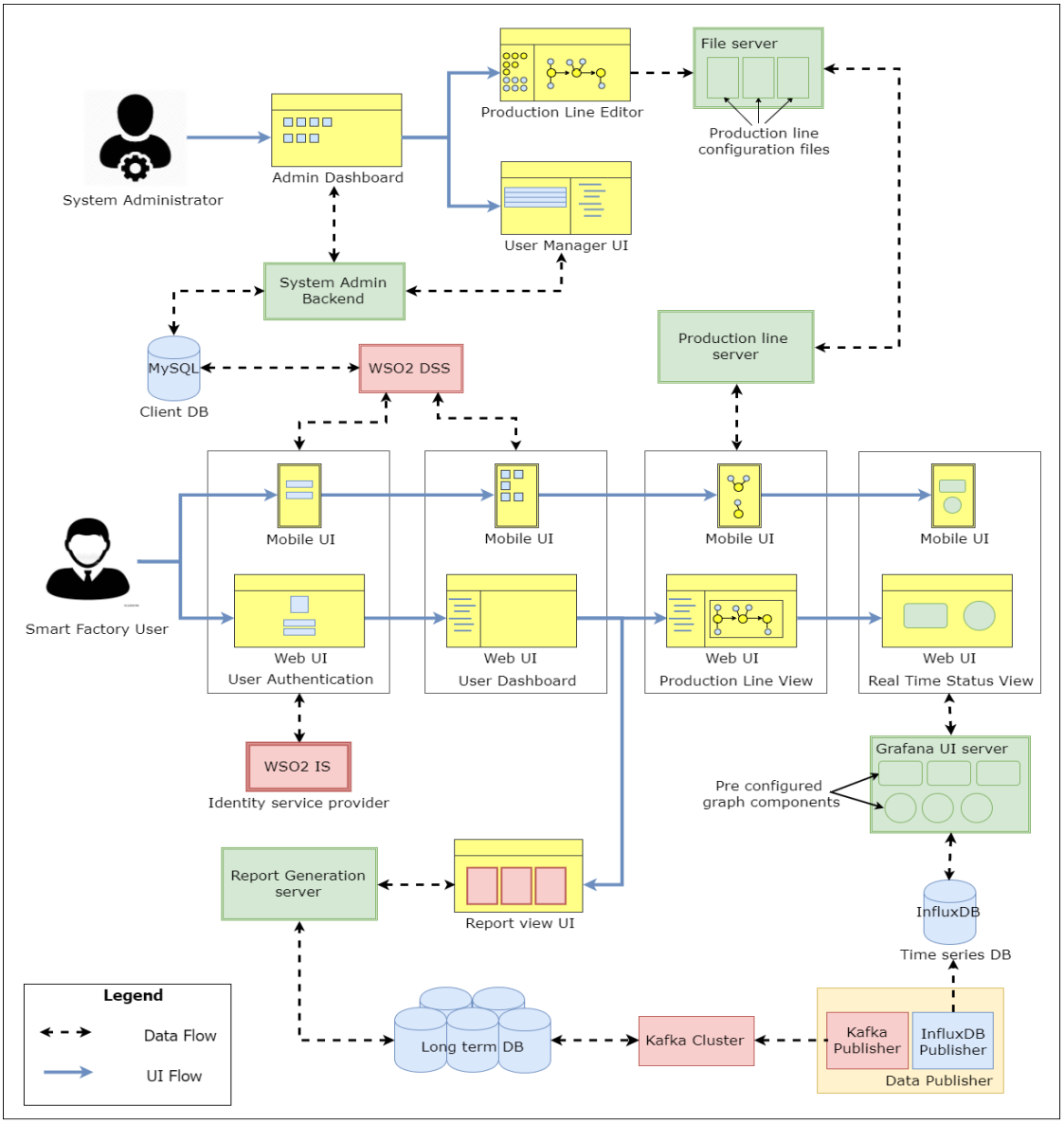


***Figure 3.1**: System Architecture Diagram*

### 3.5.1 Data Publisher

The data flow of the system begins with the Data Publisher module. PLC units in production lines are connected to these modules and these are responsible for reading sensor data from the PLCs. Ideally, each PLC will have a single Data Publisher module. However, it is possible to connect several PLCs to the same Data Publisher module using various communication methods. Data Publisher module has two separate components as Kafka Data Publisher and InfluxDB Data Publisher, which are respectively responsible for pushing real-time data into Kafka Cluster and InfluxDB Time Series Database.

### 3.5.2 Kafka Cluster

In Hitech Smart Factory, Data Publishers push vast amounts of sensor data from different sensors into the system and these data should be processed in meaningful ways in order to make more accurate managerial decisions about the factory. So that, a stream processing platform is needed to process these real-time sensor data. Massive quantities of real-time sensor data pushed by Data Publishers are processed within this stream processing platform which keeps sensor data under different topics and these topics are maintained with a unique tag name. Processed sensor data are pushed into a Long-Term Database for report generation purpose.

### 3.5.3 Long-Term Database

This database consists of data which were processed by the Kafka Cluster. This may contain information of production lines for several weeks or months, depending on the requirements of the factory. These data are used to generate reports for managerial staff and engineers.

### 3.5.4 Report Generation Server

This server consists of several services that can be used to generate reports using the data in Long-Term Database with predefined formats. Whenever the Report View User Interface (will be discussed in Subsection 3.5.5) requests for a particular report, this server query the Long-Term Database generates the report and send it to the user interface as the response.

### 3.5.5 Report View User Interface

Users can view and get printouts of reports about the production lines through this user interface. When a user required to view a particular report, this UI is used to request that from the Report Generation Server discussed in Subsection 3.5.4.

### 3.5.6 Time Series Database

This is the component used to store all the sensor data values in Hitech Smart Factory in real time. Since a sensor produces new data values by the seconds or milliseconds, it is not practical to store them in a traditional Relational Database Management System (RDBMS). Given the fact that one production line can accommodate hundreds of sensors and a factory can have hundreds of production lines, file-based data storing mechanisms are also ruled out. The best solution is to use a Time Series Database. Time series databases are software systems that are specially optimized for data indexed by timestamps. They can handle large amounts of data efficiently as well as executing queries that involve timely conditions. [21]

### 3.5.7 Grafana User Interface Server

Grafana UI server is the component used in Hitech Smart Factory to present sensor statuses to the users in a graphical manner. Once a production line is modeled through Production Line Editor (will be discussed in Subsection 3.5.12), required graphical components such as graphs and gauges can be configured using grafana UI server. The Time Series Database is connected with these graphical components and using that connection, users can view sensor data values as a point in a graph or as a reading in a gauge. When a user requests to view the status of a selected sensor, Real-Time Status View (will be discussed in Subsection 3.5.18) sends a request to Grafana UI server to acquire the graphical component.

### 3.5.8 Client Database

This is used to store the information about the clients of Hitech Smart Factory. These information consist of factories, branches, sections, production lines, the users as well as the details of which users are assigned to which factory, branches, sections or production lines. Data in this database are used to generate the user dashboards in

mobile application or the web application. The ER diagram of this database is shown in the Figure 3.2.

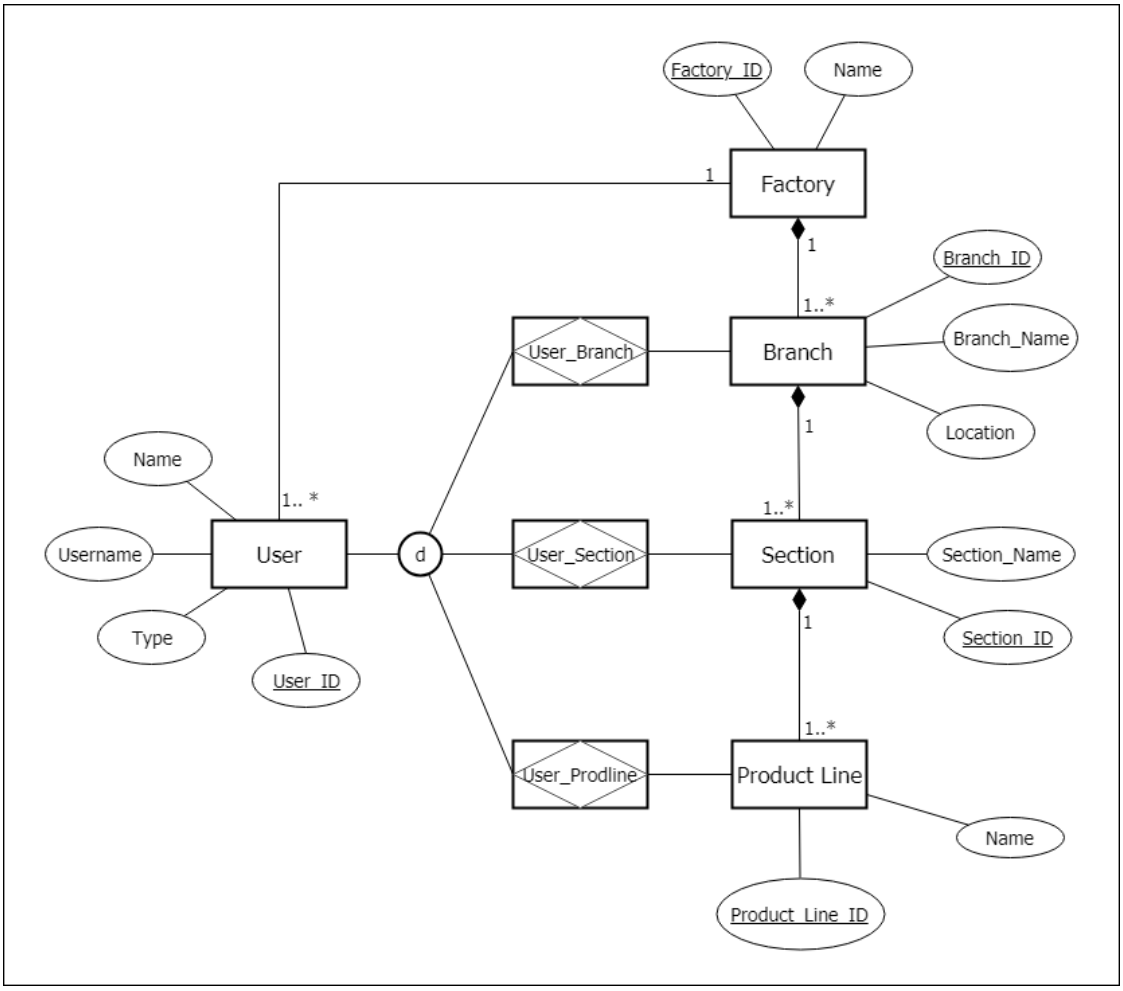


***Figure 3.2**: Entity Relationship Diagram of Client Database*

### 3.5.9 System Administrator Backend

The primary responsibility of this component is to maintain the communication between Admin Dashboard and the Client Database. The class diagram of this component is shown in Figure 3.3. All Create, Retrieve, Update, Delete (CRUD) operations done to factories, branches sections, production lines and users are handled through this component.

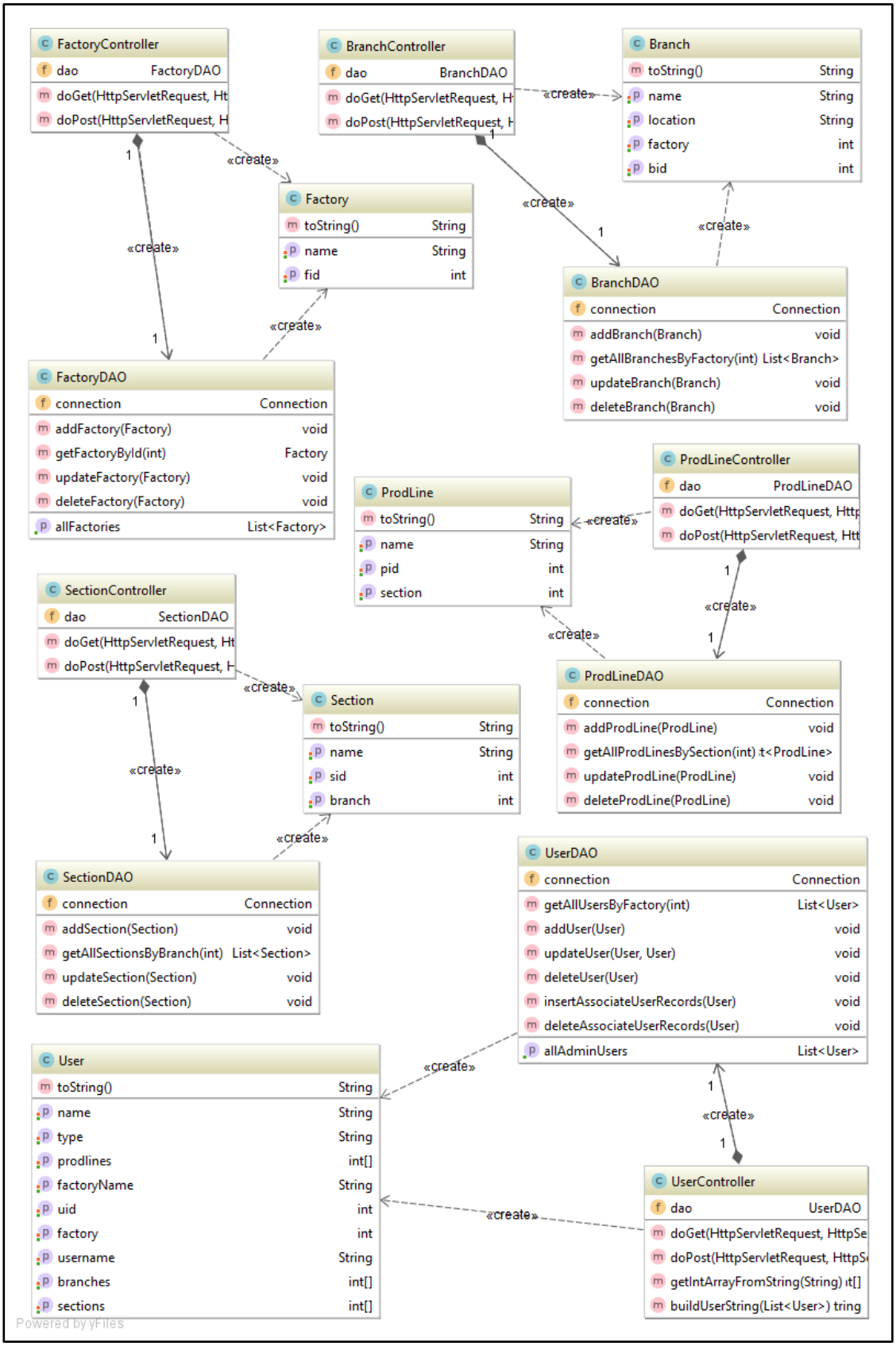


***Figure 3.3**: Class Diagram of System Administrator Backend*

### 3.5.10 Admin Dashboard

System admins of Hitech Smart Factory can access the system using this interface. There are two kinds of admins in the system as *system admins* and *factory admins*. System admins are able to create, update and delete factories, branches, sections and production lines using this particular interface. Factory admins are able to create, update and delete branches, sections and production lines within his respective factory. Once a new client requests the services of Hitech Smart Factory, the system admin can create a new factory instance with respective branches, sections and production lines along with a factory admin user account. After the credentials are sent, factory admin can take control of that factory.

### 3.5.11 User Manager User Interface

System admins are able to manage users of Hitech Smart Factory using this interface. As mentioned in above Subsection 3.5.10, system admins can manage all factory admins and factory admins can manage other users in his respective factory. As mentioned in Subsection 3.3.4, four levels of access management is enforced via this component. Factory admins are able to grant access to several branches, sections or production lines to a user based on the business requirements of the factory.

### 3.5.12 Production Line Editor

Once an admin creates a factory, a branch, a section and a production line, modeling of that production line can be done using the Production Line Editor. It has a toolbar and a canvas area where the user can drag and drop production line components and sensors which can be then connected according to the requirement. The editor generates an Extensible Markup Language (XML) based configuration file for each production line and these files are saved on the File Server.

### 3.5.13 File Server

Production lines which are modeled through the Production Line Editor are stored as XML based configuration files. These files are needed by the production line server, hence a file server is used to serve them upon requests.

### 3.5.14 Production Line Server

When a user requested to view a particular production line, that request comes to the Production Line Server. This component gets the particular production line configuration XML by communicating with the file server and generates a response to send to the user device depending on how the user needs to view it, either in the web UI or in the mobile application.

### 3.5.15 User Authentication

This is where the users of Hitech Smart Factory access the system via mobile application or the web application. Users are asked to provide their username and password which are then authenticated by a third party identity server. Once the user is authenticated successfully, the user is redirected to the user dashboard. In web application, logged in user object is maintained in browser sessions which will be used to identify access privileges of the user in user dashboard.

### 3.5.16 User Dashboard

This component is comprised of a dynamic dashboard generated according to the access privileges of the logged in user. It shows the factory details such as branches, sections, and production lines, where the user can browse through and select one production line to move forward to the Production Line View (will be discussed in Subsection 3.5.17). Factory details are requested through a third party Data Services Server, which sends data by querying the Client Database. The Users who access user dashboard via web UI is also able to redirect to the Report View User Interface as well.

### 3.5.17 Production Line View

Once the user selects a production line from user dashboard, via the mobile application or the web application, the user is redirected to Production Line View. This UI component communicates with the Production Line Server to preview the required production line. The user can select each sensor module in the selected production line to view the real-time status.

### 3.5.18 Real-Time Status View

After the user selects a sensor from the Production Line View, the particular graphical component is loaded with real-time data. To load these components, Real-Time Status View communicates with the Grafana UI Server. Both web application and mobile application is capable of displaying real-time statuses.

## 3.6 summary

This chapter demonstrated the design considerations of the Hitech Smart Factory based on the components of the system architecture. The next chapter will discuss the implementation details of the above-mentioned design considerations mainly focusing on their technical aspects.

# Chapter 4

# Implementation

## 4.1 Introduction

This chapter elaborates the implementation details of each architectural component discussed in Section 3.5 above. Each subtopic describes how each component developed, what technologies used, problems faced during the implementation and essential code snippets if applicable. The chapter ends with a description of a scenario where Hitech Smart Factory can be applied, giving a clear understanding of the overall functioning of the system.

## 4.2 Implementation Details of Architectural Components

### 4.2.1 Data Publisher

As discussed in the Subsection 3.5.1, there are two components in the Data Publisher module as InfluxDB Data Publisher and Kafka Data Publisher. The primary responsibility of the Data Publisher is to read sensor data from PLC units. PLCs store values of each sensor in a given time, in its in-build registers. Since we are using XINJE XC-3 60RT-E PLC in this context, Data Publisher module has to follow the requirements of that PLC. This particular PLC uses Modbus-RTU protocol for communication. Accordingly for the Data Publisher module to support Modbus protocol, Jlibmodbus library[1] written by Vladislav Y. Kochedykov has been used. The small learning curve, user support and the reputation it holds among the community against other Modbus libraries has been considered when choosing this particular library.

After reading the sensor data from the PLC, data has to be pushed into InfluxDB instance and the Kafka Cluster. To separate various sensor values from each other, a special tag value is used when pushing data. This tag value can represent a sensor uniquely among all other sensors in an entire factory. This is generated when modeling

[1] https://github.com/kochedykov/jlibmodbus

a production line (will be discussed in Subsection 4.2.12) and a mapping of PLC register to the tag value needs to be provided for the Data Publisher to work. Currently, a property file is used which feed in above-mentioned mapping as follows.

```
# register_name tag_value
1-fd8 1_1_1_19_4_10
1-d3 3_2_1_3_8_3
1-d2 1_2_2_7_5_9
1-d1 3_8_6_8_6_13
```

Internally to make a tag value unique, sensors path from factory to machinery component is used. The numbers separated by underscores are correspondent to factory id, branch id, section id, production line id, machinery id and sensor id respectively.

In order to push data into InfluxDB instance, influxdb-java[1] client is used. This is the official library provided by influxDB for java programming language and there were no alternatives available. Information such as hostname, port, username and password of the influxDB instance is provided for this client to connect and push data. To push data into the Kafka Cluster, Maven Kafka plugin[2] is used. Maven build tool has been selected to develop Kafka publisher over other build tools considering the facts such as support for JUnit testing and superior dependency management of Maven.

### 4.2.2 Kafka Cluster

Apache Kafka is an open-source stream processing platform developed by the Apache Software Foundation written in Scala and Java [13]. It is a very efficient stream processing system for processing large chunks of data. Hence Kafka is chosen as the better solution for sensor data processing in Hitech Smart Factory rather than traditional message servers namely RabbitMQ[3] and Apache ActiveMQ[4].

The data pushed by the Kafka publisher is processed in meaningful ways within Kafka Cluster. Kafka Cluster consists of multiple brokers to maintain load balancing. A broker

[1] https://github.com/influxdata/influxdb-java
[2] https://github.com/charithe/kafka-maven-plugin
[3] https://www.rabbitmq.com/
[4] http://activemq.apache.org/

is a Kafka server that runs in a Kafka Cluster. Furthermore, each Kafka broker can handle massive quantities of reads and writes per second without any performance impact. Accordingly, massive quantities of sensor data pushed by different sensors can be processed parallelly and very efficiently within the Kafka Cluster.

### 4.2.3 Long-Term Database

Data generated by Kafka Cluster are stored in a MySQL database. The reason behind selecting MySQL for the Long-Term Database is that after the processing is done, sensor data are organized in a structured manner and MySQL is a stable, reliable and robust solution to store structured data.

Kafka consumer is developed with Java and it pushes sensor data which has been processed within Kafka Cluster into Long-Term Database. In this database, for each sensor, there is a table to store processed information. These data are maintained for several weeks or months, depending on the requirements of the client.

### 4.2.4 Report Generation Server

Long-Term Database of Hitech Smart Factory maintains a massive amount of historical information of sensor data for report generation purpose. Reportico[1] open source PHP report designer tool has been selected in order to develop Report Generation Server of Hitech Smart Factory out of other report designing tools such as JasperReports[2] and OracleReports[3]. Reportico is an inclusion of custom PHP code to allow sophisticated manipulation of data prior to reporting and it runs against MySQL [18]. Those factors were taken into account before choosing Reportico for the Report Generation Server in Hitech Smart Factory. Furthermore, Reportico is designed to run reports either as a standalone report designer or for embedded in a web application. Reports can be invoked directly via web links and outputs are produced in HTML, PDF, and CSV formats.

[1] http://www.reportico.org/site/index.php
[2] https://community.jaspersoft.com/documentation?version=46991
[3] http://www.oracle.com/technetwork/middleware/reports/documentation/index.html

### 4.2.5 Report View User Interface

Report view user interfaces have also been designed with Reportico report designer tool since interfaces can be efficiently designed with the default interface structure provided by this tool. This default structure can be changed by changing the overall styling of Reportico. Moreover, Reportico allows designing user-friendly and easy to use interfaces.

The particular report output will be displayed in HTML, PDF, or CSV format when a user selects the date, company, branch, section, production line, machine and the sensor details in the user interface. Users also have the capability to select output format of the report through the interface as shown in Figure 4.1**.** More report formats can be added according to the client requirements.

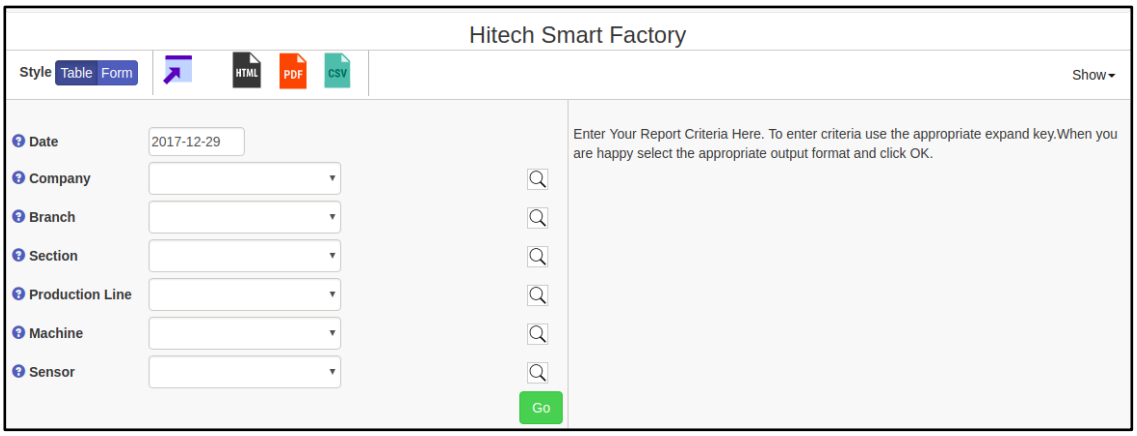


***Figure 4.1**: Report View User Interface*

### 4.2.6 Time Series Database

InfluxDB[1] is a Time Series Database built from the ground up to handle high write and query loads [14]. It is a custom high-performance data store explicitly written for time-stamped data including DevOps monitoring, application metrics, IoT sensor data, & real-time analytics. InfluxDB can be easily connected with Grafana and it uses an SQL style query language which is easy to use [19]. Moreover, InfluxDB is not just a database; it is a reliable and efficient platform which facilitates storage, monitoring, visualization and alerting of time series data. Therefore InfluxDB is selected as the time-

[1] https://www.influxdata.com/time-series-platform/influxdb/

series database solution to store sensor data produced by sensors in Hitech Smart Factory out of other time series databases such as OpenTSDB[1], KairosDB[2] and Heroic[3].

### 4.2.7 Grafana User Interface Server

Grafana[4] has been selected as the time series monitoring solution in Hitech Smart Factory out of other modern time series monitoring solutions such as Graphite[5] and Prometheus[6] because Grafana is the best of all solutions in terms of dashboard creation, visualization and customization. It is flexible, feature-rich and effortless to use visualization tool in the industry. Furthermore, core functionalities of Grafana can be extended by plugins due to its support for a vast set of plugins rather than Graphite, and Prometheus [1]. Accordingly, it is most commonly used for visualizing time series data for infrastructure and application analytics.

Grafana dashboards are composed of panels. Different graphical components such as graphs and gauges are configured on these panels and they are interacting with influxDB data source in order to represent the real-time statuses of sensors in Hitech Smart Factory. Real-time graphical representations of sensors can be requested by the web application or the mobile application using the particular tag of the sensor.

### 4.2.8 Client Database

As discussed in Subsection 3.5.8, Client Database consists of the client details of Hitech Smart Factory. An analysis was conducted on the existing database types in order to figure out the optimal database solution for Client Database and it derived that RDBMSs and NoSQL databases contain the capabilities that are suitable for the requirements. The above two types were compared and contrasted and concluded that RDBMS are more suitable for the requirements more than NoSQL databases considering following facts [8].

[1] http://opentsdb.net/
[2] https://kairosdb.github.io/
[3] https://spotify.github.io/heroic/#!/index
[4] https://grafana.com/
[5] https://graphiteapp.org/
[6] https://github.com/prometheus

- **Data structure**
  Since all the entities and attributes of the data set that are needed to be stored are already known, the schema structures can be easily defined.

- **Privileges**
  In Hitech Smart Factory, different user roles have different access privileges based on the factory, branch, section and production line each user belongs to. Since those privileges can be managed in an RDBMS database itself, this is a massive benefit in security concerns.

    - Ex: If a user has access to a particular branch of a factory, he/she should be able to access all following sections and the production lines of that branch as well. This can be easily arranged through relations of an RDBMS.

- **Data types**
  Each column of each table in RDBMS has a predefined data type. This ensures that the data are well formed. Also, data validations can be done using "CHECK" constraints.

- **Volume of the data set**
  As mentioned earlier the client details are stored in this database. This dataset is not very large hence can be easily handled by an RDBMS.

- **Support**
  Excellent support is available for RDBMSs than NoSQL databases. Also, there are a lot of individual professional consultants who possess a greater knowledge of RDBMSs. Since RDBMSs are in use for many years in the industry, the reliability of community support is higher than NoSQL.

### 4.2.9 System Administrator Backend

This component is responsible for the communication between admin functions and Client Database. It is developed using Java and factors such as security, support for object orientation and ease of scaling are considered beforehand. Java scores more compared to other solutions such as PHP and ASP.Net. System administrator backend comprises of five sets of classes that separately operate on factories, branches, sections, production lines and users. Accordingly, there are five servlet classes and the

two UI components connected to this backend, Admin Dashboard and User manager, send POST requests to these servlets. These requests are processed and necessary data operations are done by relevant Data Access Object (DAO) classes using the relevant model classes.

### 4.2.10 Admin Dashboard

Admin Dashboard is the first entry point to the system an admin of Hitech Smart Factory can have. As described in Subsection 3.5.10, two types of admins are allowed to use Admin Dashboard. The difference is that a factory admin can only view the items regards to his/her factory while system admin can view all the factories. As shown in Figure 4.2**,** factories are listed along with a 'plus' button to add new factories in the beginning.

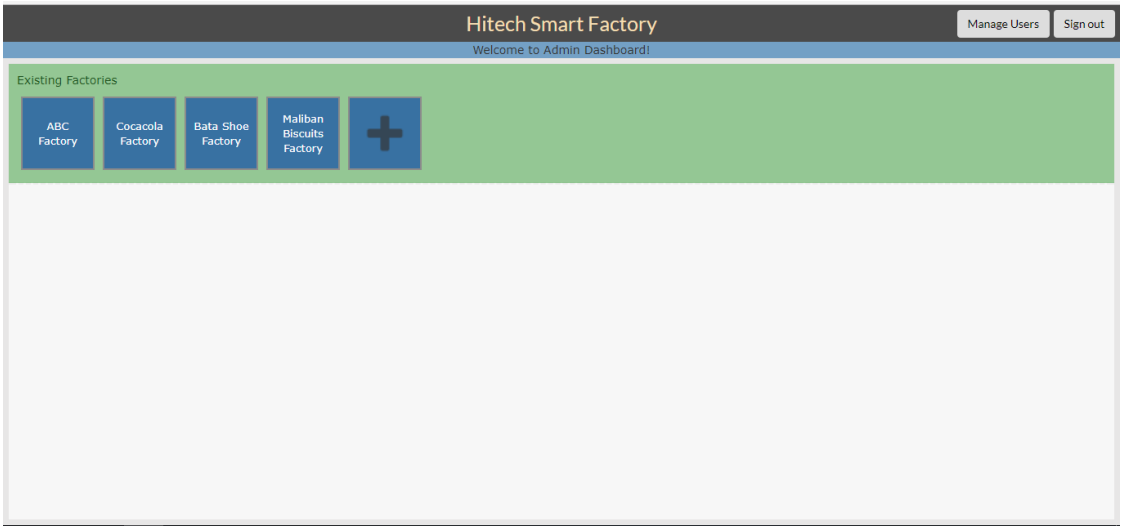


***Figure 4.2**: List of Factories in Admin Dashboard*

Once the admin clicks on a factory, all branches are listed in the same manner. Accordingly, sections and production lines are listed along the way. A completely expanded instance of the interface is shown in Figure 4.3**.**

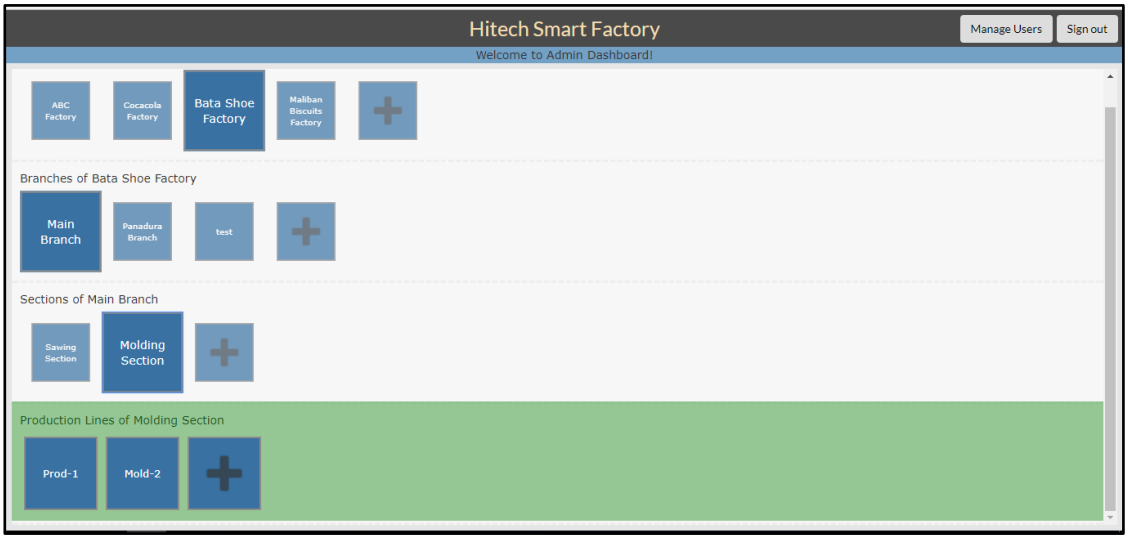


***Figure 4.3**: Expanded view of Admin Dashboard*

Each row has a plus button to add new components to the system and in each component, an edit button and a delete button appears when hovering each box with the mouse pointer. This is shown in Figure 4.4**.** By clicking these small icons, admin can edit or delete any component. When deleting a component, all child elements are deleted as well. For example, if the admin is to delete a branch, all sections under that branch and all production lines under those sections have to be deleted. Therefore a confirmation message is prompted when deleting an item.

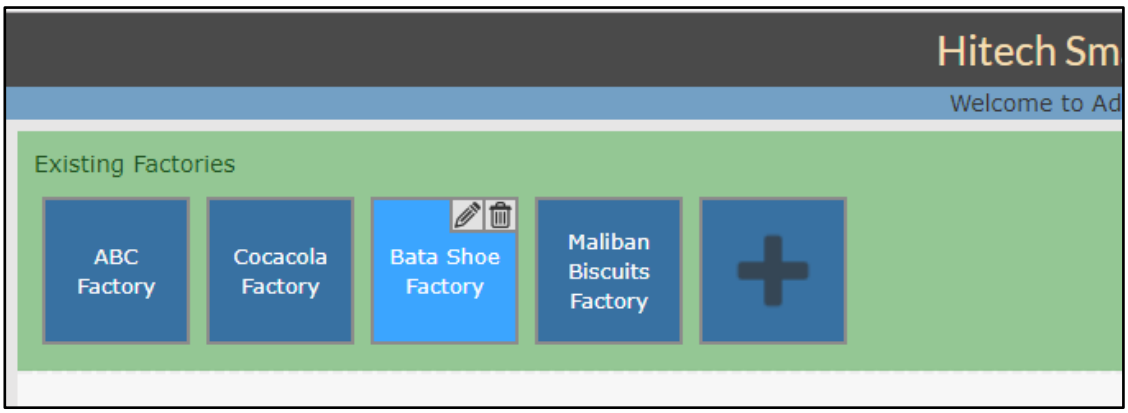


***Figure 4.4**: Edit and Delete Buttons of an item in Admin Dashboard*

In the upper right corner of the UI, two buttons are provided respectively to switch to the User Manager user interface or to sign out. Browser sessions are used to communicate the identity of the logged in user and these session objects are deleted

upon sign out. Admin user can also move into Production Line Editor (will be discussed in Subsection 4.2.12) by clicking on any production line box appear in the interface.

### 4.2.11 User Manager User Interface

System admins of Hitech Smart Factory are able to manage user accounts via this interface. There are two separate views of user manager interfaces available for the two types of administratives. As discussed in Subsection 3.5.11, the system admin can use this particular interface to manage factory administrative accounts assigned to each factory. A screenshot of the UI accessible by a system admin is shown in Figure 4.5. All factory admin accounts are listed in a table along with edit and delete actions. A button is available on the page with 'Add New User' label which can be used to add new factory admin by providing few details and selecting the relevant factory.

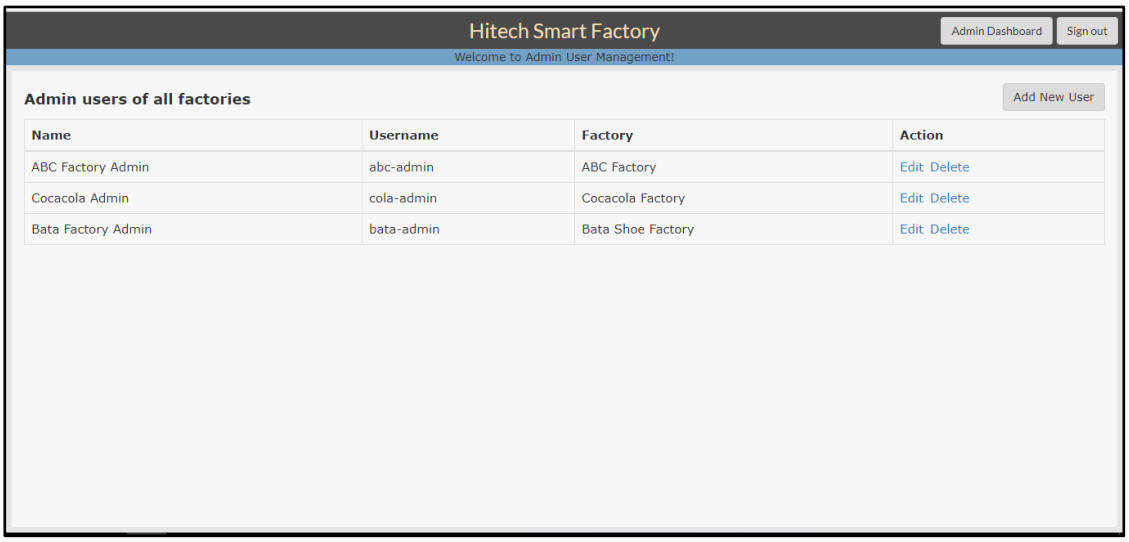


***Figure 4.5**: User Manager User Interface of System Admin*

User manager interface provided for factory admins are a bit different than the above. This interface is shown in Figure 4.6. Apart from a list of all users belonged to the factory of the admin, a special structure is provided on the right side of the interface to grant access privileges to the users.

This structure list down all items in a factory and once the admin selects any user from the table, access permissions are shown by checking relevant checkboxes. An example of this is shown in Figure 4.7. The selected user belongs to the category of section level user as described in Subsection 3.3.4.


***Figure 4.6**: User Manager User Interface of Factory Admin*

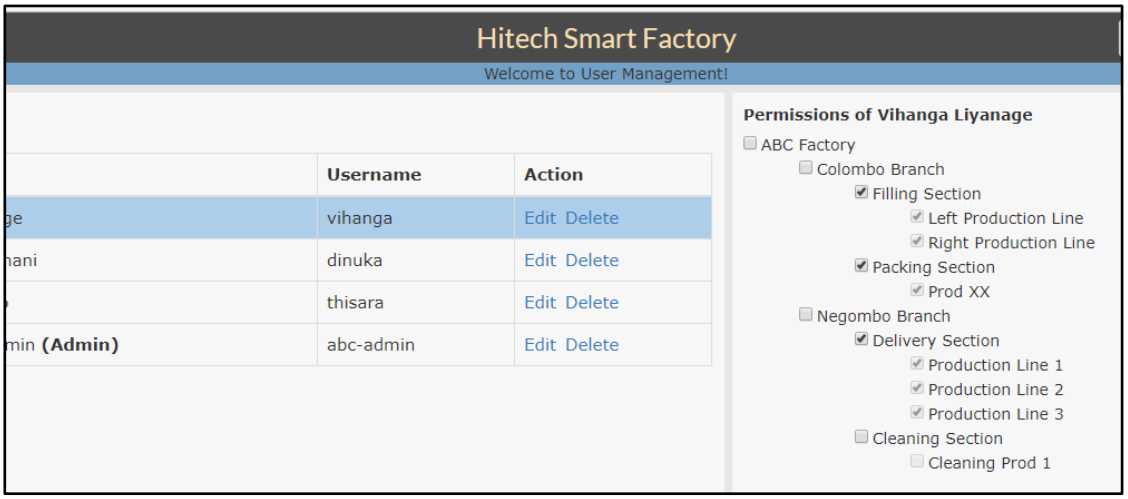


***Figure 4.7**: Example Access Permissions of a Section Level User*

Accordingly, notice that all production line checkboxes are disabled. This is to help the factory admin to assign access permissions correctly. Section level users cannot have separate access to a production line without having permissions to the section it belongs to. Following the same logic, branch level users cannot have separate access to sections or production lines without having access to the branch. To enforce this logic, JavaScript event is used and the code snippet is available in Appendix A.2.

### 4.2.12 Production Line Editor

As mentioned in the Subsection 4.2.10 above, the admin user can model production lines using the Production Line Editor by clicking on any production line box appear in

the Admin Dashboard. The initial requirement of this component is to provide an interface in which the user is able to draw a representation of an actual production line. Accordingly, it is identified that a drag and drop environment where the user can connect predefined components and sensors, is suitable. To implement this kind of a system, various Javascript libraries were considered such as mxGraph[1], vis.js[2], D3.js[3] and sigmajs[4]. Among these considered libraries, mxGraph library is chosen since it is the closest package that offered the exact requirements mentioned above. mxGraph is the library used in draw.io tool[5] provided by Google, which is also a similar drag and drop graph drawing tool. Therefore it has the highest industry reputation and support compared to other libraries as well.

The editor interface of a newly created production line is shown in Figure 4.8. Toolbox is in the left side of the window and the large white colored area is the canvas. To construct this toolbox, mxGraph library uses a special XML configuration file named '*diagrameditor.xml*'. This file defines all toolbox icon image paths, canvas area icon image paths, icon styles, connection styles, etc.


***Figure 4.8**: User interface of Production Line Editor*

[1] https://github.com/jgraph/mxgraph
[2] http://visjs.org/
[3] https://d3js.org/
[4] http://sigmajs.org/
[5] https://www.draw.io/

A button is given in the left upper corner of the toolbox to add new toolbox items and admin users can add new sensors or components to the system using this functionality. Since the required components and sensors are tend to differ from one production factory to another, Production Line Editor is equipped to handle new toolbox items as well. A screenshot of using this feature to add a new item is shown in Figure 4.9. Internally what this feature does is collecting required information from the user and updating the above mentioned configuration file.

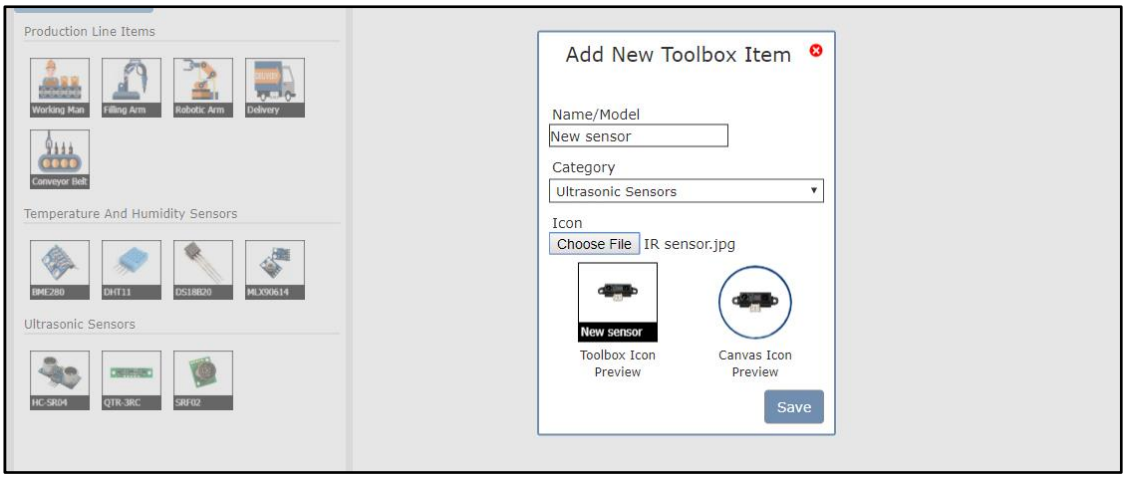


***Figure 4.9**: User Interface of Add New Toolbox Item*

Admin users can drag and drop machinery components and sensors from the toolbox to the canvas area and then they can be connected using connectors. A drawn production line is presented in the Figure 4.10. Once the admin user finishes modeling a production line, the button under the canvas area can be used to save it. The editor generates a XML configuration file for each production line modeled in the editor. This can be viewed by checking the view source checkbox under the canvas area. When the user request to save a production line, editor sends the relevant XML configuration to the file server (will be discussed in Subsection 4.2.13) where it is saved in a uniquely named XML file.

When a component or a sensor is dropped into the canvas area, a unique tag value is generated. As discussed in the Subsection 4.2.1, this tag value is used in the Data Publisher Module to push data into the Time Series Database as well as the Kafka Cluster. Admin user can view this tag value by clicking on the gear icon which appears in each component as shown in the Figure 4.11**.** Moreover, this particular tag value is

also used to request graphical components from Grafana UI server by Real-Time Status View.

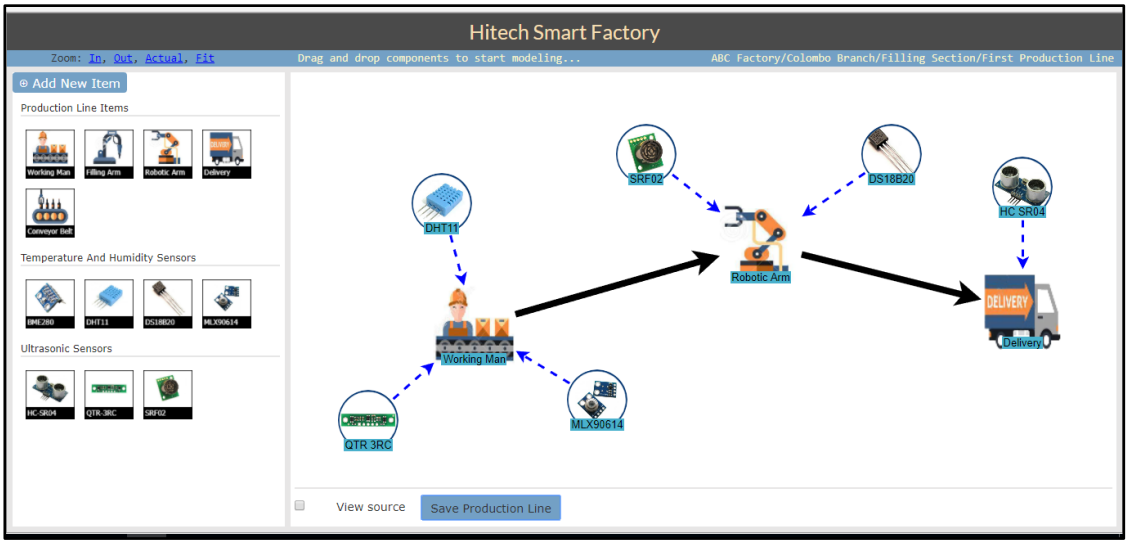


***Figure 4.10**: A Modeled Production Line in Production Line Editor*

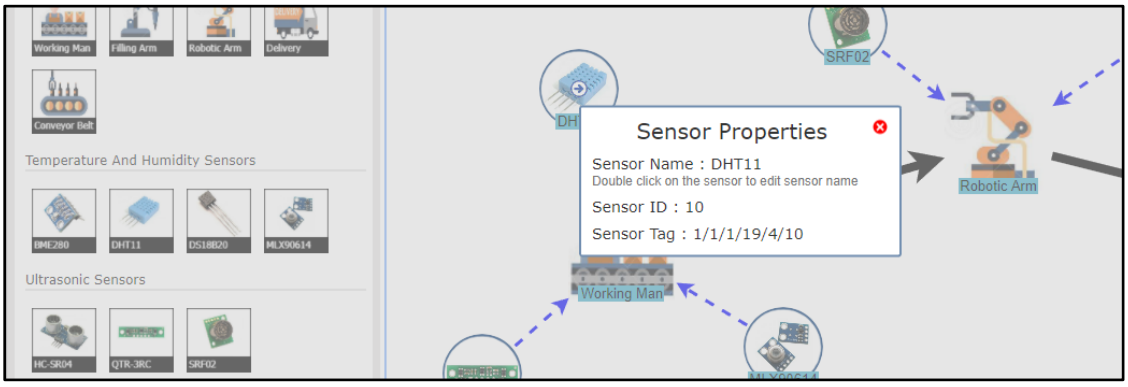


***Figure 4.11**: Properties of the Selected Sensor*

### 4.2.13 File Server

XML configurations generated by the Production Line Editor are saved as XML files, as described above. These files are needed to be accessed by web or mobile Production Line View components as well as by the editor itself in case of updating a previously modeled production line. To easily communicate them across applications, a simple file server is used. Nginx[1], apache tomcat[2] and apache httpd[3] were possible options and

[1] https://www.nginx.com/
[2] http://tomcat.apache.org/
[3] https://httpd.apache.org/

Apache httpd server was selected considering the small learning curve and the simplicity of the workload. The main problem faced during the implementation of file server was the cross-origin resource sharing (CORS) problem[1]. By default browsers do not allow scripts in a domain to access resources in a different domain. To disable this security measure, CORS allow header has to be set in all responses served by the file server.

### 4.2.14 Production Line Server

The production line server is developed using PHP[2]. Apart from PHP, some other server-side technologies such as J2EE[3], NodeJS[4], ASP.NET[5] and python[6] were taken into consideration. All of those technologies require much more workload than PHP. Considering the limited development time period and the moderate learning curve of PHP, it is selected as the technology to develop production line server.

When a user requests to view a production line, a POST request is sent to the Production Line Server including factory id, branch id, section id and the production line id. Then the production line server communicates with the file server and gets the relevant XML configuration file of the production line using the above-mentioned ids along with several other files. These files are used to extract some other details of the components and sensors of the production lines including the URLs of the image of the components and sensors. As the last step Production Line Server generates a response including all those information and sends it to the Production Line View, described in Subsection 4.2.19. This task can be done inside the mobile application, but the intensive processing of XML files can affect the performance.

### 4.2.15 Mobile Application

The mobile application of Hitech Smart Factory is designed for a selected set of users such as engineers and machine operators. They can use the mobile application to view the real-time status of sensors. The primary implementation consideration of the

[1] https://en.wikipedia.org/wiki/Cross-origin_resource_sharing
[2] https://www.asp.net/
[3] http://www.oracle.com/technetwork/java/javaee/overview/index.html
[4] https://nodejs.org/en/
[5] https://www.asp.net/
[6] https://www.python.org/

mobile application is that it needs to be compatible with both Android and iOS platforms. One solution would be developing two separate applications for each of the two platforms. This approach requires writing the same code twice which is very time-consuming. Hence using a cross-platform development technology is taken into consideration. React Native[1], Xamarin[2] and Ionic[3] were compared in order to select the most suitable tool for the development. A comparison between these three technologies is shown in Table 4.1: Comparison of features of selected technologies.

***Table 4.1**: Comparison of features of selected technologies*

| Feature | Xamarin | React Native | Ionic |
|---|---|---|---|
| Learning curve | High<br>(since developer did not have any experience in C#) | Low<br>(Since developer is familiar with Javascript) | Low<br>(Since developer is familiar with Javascript) |
| Deployment | Easy and free | Easy and free | Need to upgrade to a paid subscription |
| Access to native widgets | Yes | Yes | No |
| Hot reloading<br>(Ability to view the changes in the code without recompiling) | No | Yes | No |

Considering the facts of the small learning period, free and easy deployment and hot reloading feature, React Native was selected as the most suitable technology. React Navigation[4], an extensible yet easy-to-use navigation solution is used in order to handle the screen navigations in the application. React native is based on Flux architecture[5],

[1] https://facebook.github.io/react-native/
[2] https://www.xamarin.com/
[3] https://ionicframework.com/
[4] https://reactnavigation.org/
[5] https://facebook.github.io/flux/

an architecture which supports unidirectional data flow. Redux library[1] is used in the application as the implementation of Flux architecture. NPM[2] is used as the package manager and Fetch API[3] is used to manage the API requests.

### 4.2.16 User Authentication

In Hitech Smart Factory user authentication plays a significant role. Therefore it is decided to use a well reputed, free and open source third-party identity server in order to minimize the workload, adapt the development process to suit the limited time period and to avoid possible errors if developed from scratch. After a comprehensive study on existing identity servers, WSO2 IS[4] was selected as the suitable tool based on the facts that it reduces identity provisioning time, guarantees secure online interactions, decreases the identity management and entitlement management administration burden by including the role-based access control (RBAC) convention. Whenever a user provides username and password to Hitech Smart Factory using the web application or the mobile application, the user is authenticated through the WSO2 IS instance. If the login is successful, then only the user can proceed further.

### 4.2.17 User Dashboard

The mobile application and the web application both consist of separate user dashboard components. As mentioned in Subsection **3.5.16**, this is generated dynamically with respect to the user who is logged in to the system. Hence it is necessary to acquire the details of the factory, branches, sections and production lines that are accessible to the logged in user. As mentioned in Subsection **4.2.8**, Client Database possesses these details and they need to be obtained from the database.

The mobile application requires an API service which can fulfill the above-mentioned task of data retrieval from the Client Database. Considering that requirement along with the facts of minimizing the workload, adapting the development process to match the limited time period and to obviate the possible errors of coding from scratch, it is decided to use a third party data service server. Accordingly, WSO2 Data Services

[1] https://redux.js.org/
[2] https://www.npmjs.com/
[3] https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API
[4] https://wso2.com/identity-and-access-management

server (WSO2 DSS)[1] is chosen as it is a free and open source and well-reputed data service server in the industry. Further details on this will be further elaborated in Subsection **4.2.18**.

The mobile and web-based user dashboard components request dashboard details of the logged in user from the data services in WSO2 DSS. Then the dynamic dashboards are generated by processing the responses received.

The dashboard of the mobile application shows branches in the initial screen. This screen is illustrated in Figure 4.12. When a user selects a particular branch from the available set of branches, sections of that particular branch appear. Similarly, by selecting a section, the user can view available production lines as well. Then the user can select a production line among them which triggers a POST request to the Production Line Server to get the details of that production line. A detailed description of Production Line Server is provided in Subsection 4.2.13. After that, the user is redirected to the Production Line View. Further details of the Production Line View component is discussed in Subsection 4.2.18.

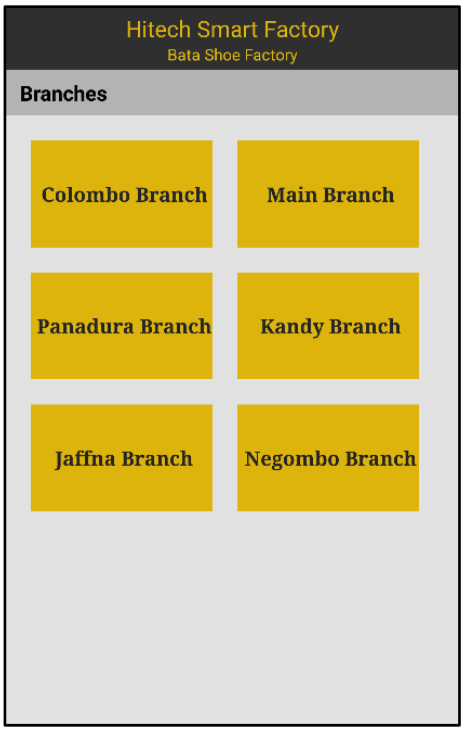


***Figure 4.12**: The Initial Screen of the User Dashboard of Mobile Application*

[1] https://wso2.com/products/data-services-server/

User dashboard of the web application operates similarly to the mobile application internally, yet differs in user interfaces. To implement the Web User Dashboard, Node.js[1] is used as the backend considering the ease of development and maintenance as well as the ability to handle extensive amounts of requests compared to other solutions. The initial screen of the web user dashboard is displayed in Figure 4.13. Left side menu list down the branches of the production lines that are accessible to the user. In this example, user *Vihanga Liyanage* is a branch level user who has access to only the Main Branch of Bata Shoe Factory.

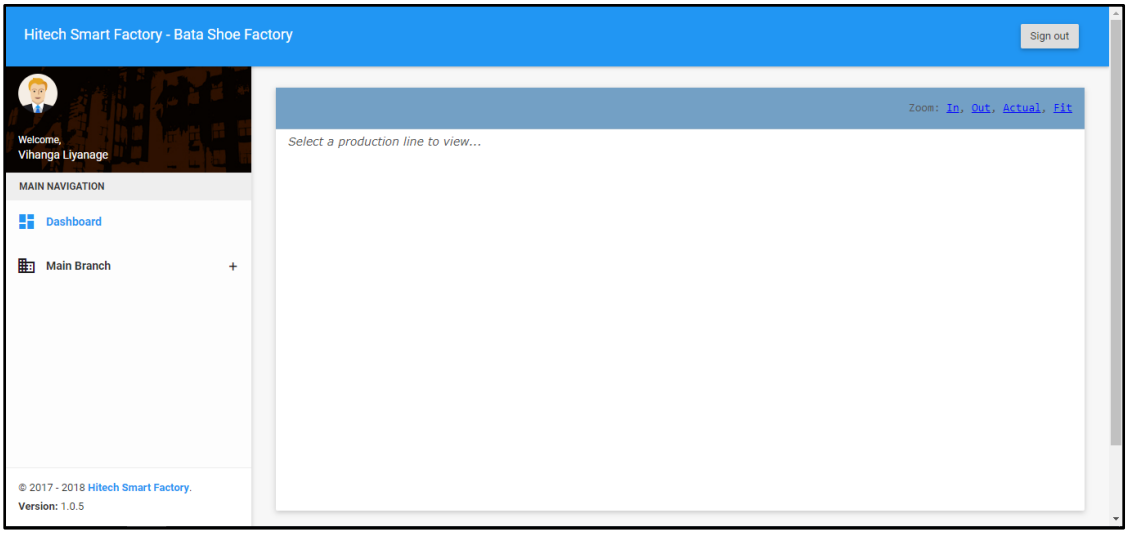


***Figure 4.13**: The initial screen of User Dashboard of Web Application*

By clicking on a branch in the list, the user can discover a list of sections comes under that branch, which is accessible to the user. Accordingly, by clicking on each section, production lines are revealed. This feature is shown in Figure 4.14. Since the user Vihanga Liyanage is a branch level user, he is entitled to access all sections and production lines of Main Branch.

If another user, *Poornima Karunarathna*, is considered, who is a production line level user with access to only the *Sawing Line 1* of Sawing Section of Main Branch, The dashboard she gets is presented in Figure 4.15.

[1] https://nodejs.org/en/


***Figure 4.14**: Navigation Menu of a Branch Level User of Web Application*


***Figure 4.15**: Navigation Menu of a Production Line Level User of Web Application*

Clicking on any production line in the menu triggers a request to production line server which responds with the XML configuration of that production line. This configuration is used to render the production line.

### 4.2.18 WSO2 DSS

WSO2 DSS provides a variety of flexible features which can be adapted to accommodate the user requirements. In WSO2 DSS, users are able to define data services which facilitate automatic generation of CRUD operations, flexible mapping of query responses to custom XML data formats and expose those data services as RESTful web services. The WSO2 DSS instance used in Hitech Smart Factory is hosted on an Ubuntu virtual machine created in Google Cloud Platform. Since the data services need

to be available whenever a user requests them, it is running as an Ubuntu service on the virtual machine.

There are 5 data services defined in WSO2 DSS. Whenever a user is authenticated, a POST request is sent to a data service containing the username of the user as a parameter. This data service then returns a response including user id, user type and the factory which the user is assigned to. According to the user type, another POST request is sent to one of the four data services which returns details of the branches, sections and production lines that particular user is given access to. All these data services are exposed as RESTful web services and both mobile and web applications use above-mentioned data services to generate the User Dashboard.

### 4.2.19 Production Line View

The Production Line View of the mobile application lists down the machinery components of the production line selected by the user from the User Dashboard. When a user selects a particular component from this view, a list of sensors which are connected to that component is shown below. This screen is shown in Figure 4.16.

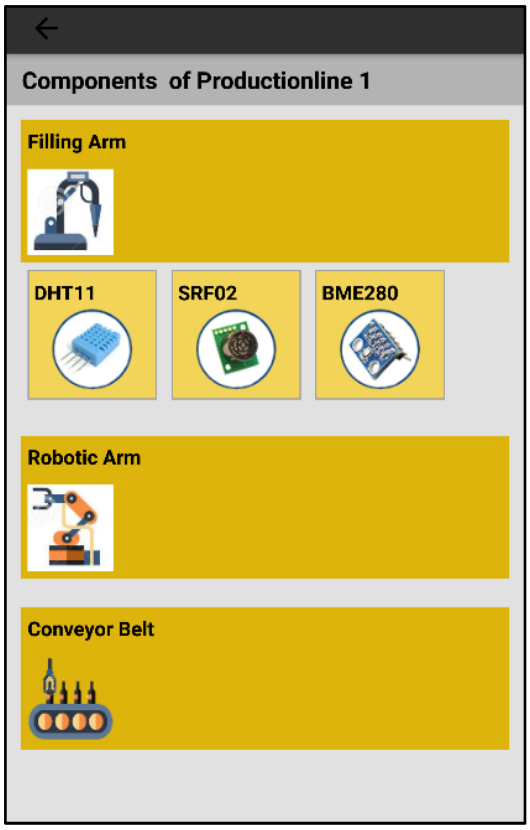


***Figure 4.16**: Production Line View of Mobile Application*

The user can select a sensor to view the real-time status from this screen. Then a POST request is sent to the Grafana UI Server requesting the graphical component which shows the real-time status of the selected sensor. The user has redirected to the Real-Time Status View afterward.

Web-based Production Line View is integrated into the web User Dashboard. As mentioned in the Subsection 4.2.17, when the user clicks on a production line from the left side menu, modeled production line is rendered using the XML configuration sent by production line server. A rendered production line is illustrated in Figure 4.17. mxGraph, which was the same library used to implement the Production Line Editor is used to render the production lines in Web-based Production Line View. The user is able to select any sensor from this view to move onto the Real-Time Status View.

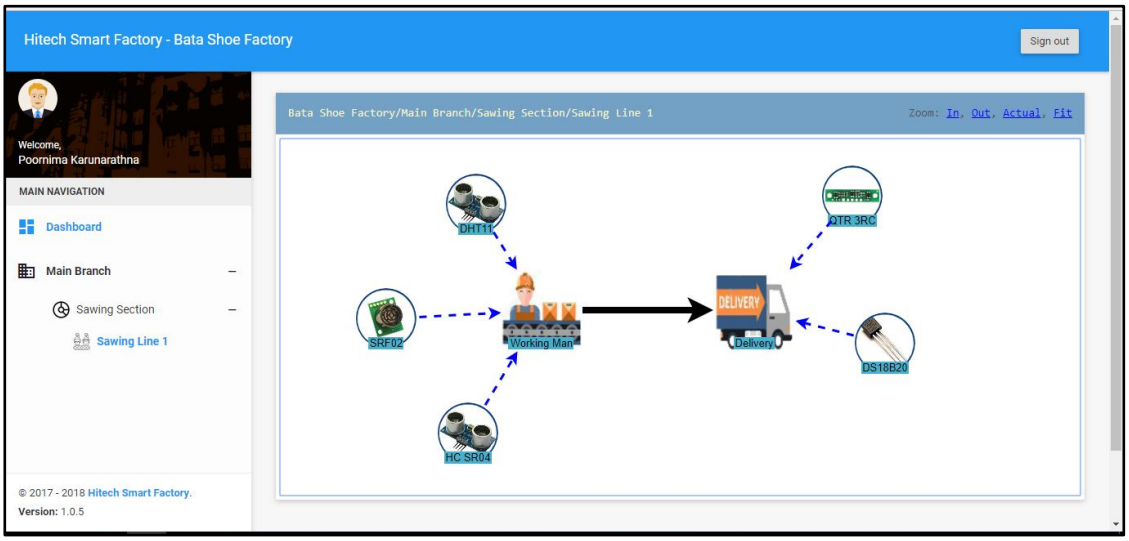


***Figure 4.17**: A rendered Production Line*

## 4.2.20 Real-Time Status View

The Real-Time Status View of mobile application consists of graphical components which are sent by the Grafana UI Server. These graphical components can be graphs or a gauge where the user can see the changes in the sensor value in real time. Here the web view[1], a unique feature provided in React Native, is used to achieve this feature in the mobile application. This screen is presented in Figure 4.18.

[1] https://facebook.github.io/react-native/docs/webview.html

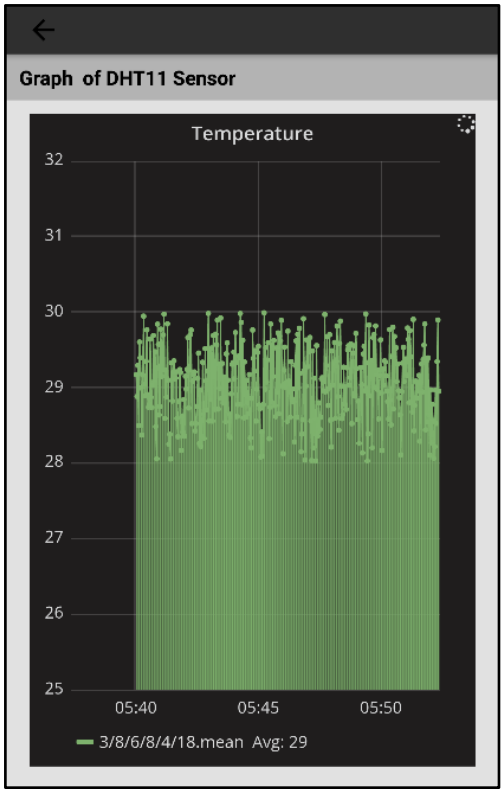


***Figure 4.18**: A Real-Time Status View Graph of Mobile Application*

In the web-based user interface, the user can just click on any sensor presented in the production line model and the graphical component that is attached with that sensor is opened in a new tab with real-time data. Web application takes the tag value of the clicked sensor as an input and uses that to request the graphical component from the Grafana UI Server. An example graph with data is displayed in Figure 4.19.

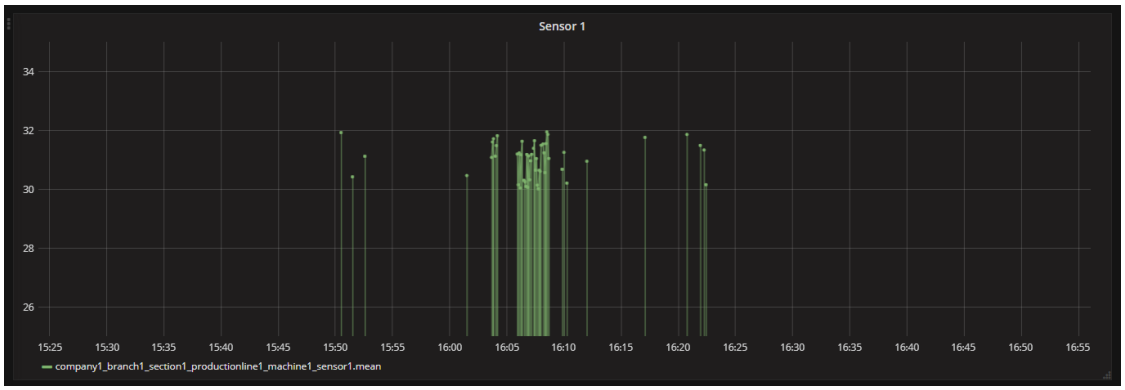


***Figure 4.19**: A Real-Time Status View Graph of Web Application*

## 4.3 Application of the Proposed Solution

The overall process of Hitech Smart Factory is somewhat complicated. Therefore, overall setting up process will be described here, starting from the point which a new production company requests the services of Hitech Smart Factory.

Once the request comes through, the system admin logs into the system and creates a new factory instance. Then, a new factory admin account is created and the newly created factory is assigned to that account. After that, either the system admin or the factory admin can add the branches, sections and the production lines of the factory to the system and model the production line. Since it is not beneficial to monitor and view the real-time statuses of all the sensors in a production line, the admin may discuss with the engineering and managerial staff to select the most preferred set of sensors to add to the model.

After the modeling is done, data publisher modules and graphical components have to be configured to push and view data in the system respectively. Tag values generated for each sensor by the Production Line Editor are used to feed the PLC register mapping as discussed in Subsection 4.2.1. Once all the configurations are done, factory admin creates the user accounts and assign the branches, sections and production lines to the users according to the business requirements of the factory. At this point, setting up of the system concludes. Users can log in to the system and view the real-time status of the sensors in the production lines which they are assigned to, using either web application or the mobile application. Besides that, they are capable of generating and viewing reports according to the user privileges that have been given.

## 4.4 Summary

In this Chapter, implementations of all the individual architectural components in the Hitech Smart Factory are described with the technologies used, along with the reasons behind selecting each of those technologies. In addition to that, the connections between those individual components and the way they communicate with each other. A scenario where Hitech Smart Factory can be applied is described in order to provide a better understanding of the overall functioning of the system. In the next chapter, the evaluation process of Hitech Smart Factory is discussed.

# Chapter 5

# Evaluation

## 5.1 Introduction

This chapter is focused on presenting the conducted evaluation process on the proposed solution. Section 5.2, Evaluation Model describes how the system has been evaluated and Section 5.3 elaborates on the results obtained. In Section 5.4, some observations are discussed regarding the performance of the system.

## 5.2 Evaluation Model

Hitech Smart Factory is a product based project which is focussed on the immense need for cost-effective and easily integrate production line monitoring system in the automation industry. In order to visualize the exact outcome of the project, it had to be deployed into a real-world production factory and monitored carefully for a considerable period of time. This was not possible at the time of composing this dissertation given the limited time availability and the lack of resources. Hence a demo setup was created and a user evaluation was conducted in order to determine the progress made and usability of the system.

The demo setup used is as follows.

- PLC unit - XINJE XC3-60RT-E, connected via a serial cable and a serial to USB adapter RS-232[1]
- Data Publisher Module - Run on a laptop with Intel Core i7 processor and 8GB RAM, connected to the internet using the ethernet connection. (100 Mbps)
- Time Series Database, Long Term Database, Report Generation Server and Grafana UI Server - Installed on a virtual machine with 1 vCPU processor and 3.75 GB RAM, in Google Cloud Platform.

[1] http://www.serialgear.com/1-Port-Serial-USB-CHEAP-SERIAL.html

- File Server, Production Line Server and Web User Dashboard - Installed on a t2.micro[1] virtual machine instance, in Amazon Elastic Cloud.
- Admin Dashboard, User Manager, Client Database, WSO2 DSS - Installed on a virtual machine with 1 vCPU processor and 4GB RAM, in Google Cloud Platform.

A questionnaire was created for the user evaluation of all the functionalities provided by Hitech Smart Factory as well as the overall product. The system was handover to selected users to explore and they were asked to answer the questionnaire describing their experience. The group of selected users consisted of Mr. Buddhika Marasinghe, Director of Hitech Solutions (Pvt) Ltd, Mr. Nuwan Pallewela, Senior Software Engineer at Sysco Labs and Mr. Samith Chathuranga Perera, Assistant Electronic Engineer at Hitech Solutions (Pvt) Ltd.

## 5.3 Results

The responses given by the users for the user evaluation questionnaire are shown in SubSections 5.3.1, SubSections 5.3.2. And SubSections 5.3.3.

[1] https://aws.amazon.com/ec2/instance-types/

### 5.3.1 Responses given by Mr. Buddhika Marasinghe

Name *

Buddhika Marasingha

Designation *

Director / Engineer

Company *

Hitech Solutions

Facory Management *

| | Poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| Ability to add/remove/update/view factory | ○ | ○ | ○ | ○ | ◉ |
| Ability to add/remove/update/view branch | ○ | ○ | ○ | ○ | ◉ |
| Ability to add/remove/update/view section | ○ | ○ | ○ | ◉ | ○ |
| Ability to add/remove/update/view production line | ○ | ○ | ○ | ○ | ◉ |

## User Management *

| | Poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| Ability to add/update/remove users | ○ | ○ | ○ | ◉ | ○ |
| Ability to assign access permissions to users | ○ | ○ | ○ | ◉ | ○ |

## Production Line Editor *

| | Poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| Ability to model production lines by dragging and dropping components and sensors | ○ | ○ | ○ | ◉ | ○ |
| Ability to update modeled production lines | ○ | ○ | ○ | ○ | ◉ |
| Ability to add custom components or sensors to the tool box | ○ | ○ | ○ | ◉ | ○ |

## View real time status *

| | Poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| Ability to select required component or sensor to view system status | ○ | ○ | ◉ | ○ | ○ |
| Ability to view the real time status of a selected component or sensor | ○ | ○ | ○ | ◉ | ○ |

## Report generation tool *

| | Poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| Ability to design reports | ○ | ○ | ◉ | ○ | ○ |
| Ability to generate reports using a design and the data provided | ○ | ○ | ◉ | ○ | ○ |

Overall Product *

| | poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| User interfaces are easy to work with | ○ | ○ | ○ | ◉ | ○ |
| Understandability of Overall process | ○ | ○ | ○ | ○ | ◉ |
| Usability of the system in a real life scenario | ○ | ○ | ○ | ◉ | ○ |

Overall impression about the system. *

Good.

Future developments or points to improve.

need to make it nicely GUI

### 5.3.2 Response given by Mr. Nuwan Pallewela

Name *

Nuwan Pallewela

Designation *

Senior Software Engineer

Company *

Sysco Labs (Pvt) Ltd.

## Facory Management *

| | Poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| Ability to add/remove/update/view factory | ○ | ○ | ○ | ◉ | ○ |
| Ability to add/remove/update/view branch | ○ | ○ | ○ | ◉ | ○ |
| Ability to add/remove/update/view section | ○ | ○ | ○ | ◉ | ○ |
| Ability to add/remove/update/view production line | ○ | ○ | ○ | ◉ | ○ |

## User Management *

| | Poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| Ability to add/update/remove users | ○ | ○ | ○ | ◉ | ○ |
| Ability to assign access permissions to users | ○ | ○ | ◉ | ○ | ○ |

## Production Line Editor *

| | Poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| Ability to model production lines by dragging and dropping components and sensors | ○ | ○ | ○ | ◉ | ○ |
| Ability to update modeled production lines | ○ | ○ | ○ | ◉ | ○ |
| Ability to add custom components or sensors to the tool box | ○ | ○ | ○ | ◉ | ○ |

## View real time status *

| | Poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| Ability to select required component or sensor to view system status | ○ | ○ | ○ | ◉ | ○ |
| Ability to view the real time status of a selected component or sensor | ○ | ○ | ○ | ◉ | ○ |

Report generation tool *

| | Poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| Ability to design reports | ○ | ○ | ○ | ◉ | ○ |
| Ability to generate reports using a design and the data provided | ○ | ○ | ○ | ◉ | ○ |

Overall Product *

| | poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| User interfaces are easy to work with | ○ | ○ | ○ | ◉ | ○ |
| Understandability of Overall process | ○ | ○ | ○ | ◉ | ○ |
| Usability of the system in a real life scenario | ○ | ○ | ◉ | ○ | ○ |

Overall impression about the system. *

Overall system is looking good and need to put into real production so that we could get more idea about the user requirements.

Future developments or points to improve.

User interfaces can be improved and report generation can be extended to be more aligned with business requirements as well as user expectations.

## 5.3.3 Response given by Mr. Samith Chathuranga Perera

Name *

Samith Chathuranga Perera

Designation *

Assistant Electronic Engineer

Company *

Hitech Solutions Pvt LTD

## Facory Management *

| | Poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| Ability to add/remove/update/view factory | ○ | ○ | ○ | ◉ | ○ |
| Ability to add/remove/update/view branch | ○ | ○ | ○ | ◉ | ○ |
| Ability to add/remove/update/view section | ○ | ○ | ○ | ◉ | ○ |
| Ability to add/remove/update/view production line | ○ | ○ | ○ | ◉ | ○ |

## User Management *

| | Poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| Ability to add/update/remove users | ○ | ○ | ○ | ◉ | ○ |
| Ability to assign access permissions to users | ○ | ○ | ○ | ◉ | ○ |

## Production Line Editor *

| | Poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| Ability to model production lines by dragging and dropping components and sensors | ○ | ○ | ○ | ○ | ◉ |
| Ability to update modeled production lines | ○ | ○ | ○ | ◉ | ○ |
| Ability to add custom components or sensors to the tool box | ○ | ○ | ○ | ○ | ◉ |

## View real time status *

| | Poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| Ability to select required component or sensor to view system status | ○ | ○ | ◉ | ○ | ○ |
| Ability to view the real time status of a selected component or sensor | ○ | ○ | ◉ | ○ | ○ |

Report generation tool *

| | Poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| Ability to design reports | ○ | ○ | ◉ | ○ | ○ |
| Ability to generate reports using a design and the data provided | ○ | ○ | ◉ | ○ | ○ |

Overall Product *

| | poor | Fair | Satisfactory | Very good | Excellent |
|---|---|---|---|---|---|
| User interfaces are easy to work with | ○ | ○ | ○ | ◉ | ○ |
| Understandability of Overall process | ○ | ○ | ○ | ◉ | ○ |
| Usability of the system in a real life scenario | ○ | ○ | ○ | ◉ | ○ |

Overall impression about the system. *

User friendly, Industrially valuable system.

Future developments or points to improve.

At the same time if the system can communicate with PLC both ways (Two way Communication) that will be great.

## 5.4 Other Observations

The main functionality of Hitech Smart Factory is the Real Time Status View. Even though the intention was to actually see the changes of sensor data values in real time, latency could occur given the distance a single data point has to travel from PLC unit to a graphical component in order to appear in Real Time Status View. Therefore, these time gaps are also observed to obtain a judgment on the matter. Following are the results gathered.

- Time to send data from PLC to Data Publisher Module.
    - About 1 second

    - Tested using XINJE XC3-60RT-E PLC, connected to a laptop (Intel Core i5 processor with 8GB RAM) running the Data Publisher Module, using a USB to Serial Adapter RS-232.
- Time to send data from Data Publisher Module to a graphical component.
    - Less than 1 second
    - Tested using random values without connecting a PLC.
- Time to send data from PLC to a graphical component.
    - About 1 to 2 seconds
    - Tested using XINJE XC3-60RT-E PLC, connected to a laptop (Intel Core i5 processor with 8GB RAM) running the Data Publisher Module, using a USB to Serial Adapter RS-232.

## 5.5 Summary

This chapter described the evaluation model that is used to evaluate the proposed solution and the results obtained. Some observations regarding the system performance are also discussed in the chapter. The next chapter will conclude the project giving a review of the results obtained, limitations and an implication of future work.

# Chapter 6

# Conclusion and Future Work

## 6.1 Introduction

This chapter includes conclusions gathered from the user evaluations, limitations of the proposed solution and implications for future studies.

## 6.2 Conclusions Gathered From User Evaluation

As mentioned in Section 5.2, a questionnaire was presented to the key people of the project and the results obtained are summarized to make conclusions about the proposed solution. The user feedback on the factory management functionality is shown in Figure 6.1**.** All of the users have agreed that add, view, update and remove functionalities of factories, branches, sections and production lines are above the satisfactory level.

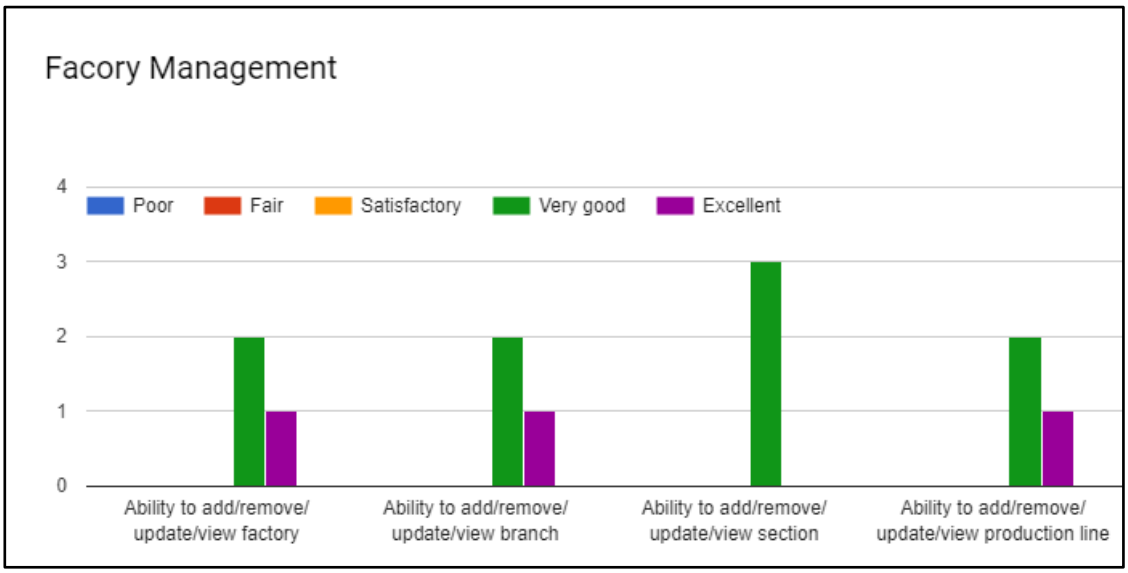


***Figure 6.1**: Summary of User Feedbacks on Factory Management Functionality*

User evaluation of the user management functionality is shown in Figure 6.2. All the users have found that add, remove, view and update features of users are made easy with the provided user interfaces. Assign access permissions of users functionality is also up to the satisfactory level.

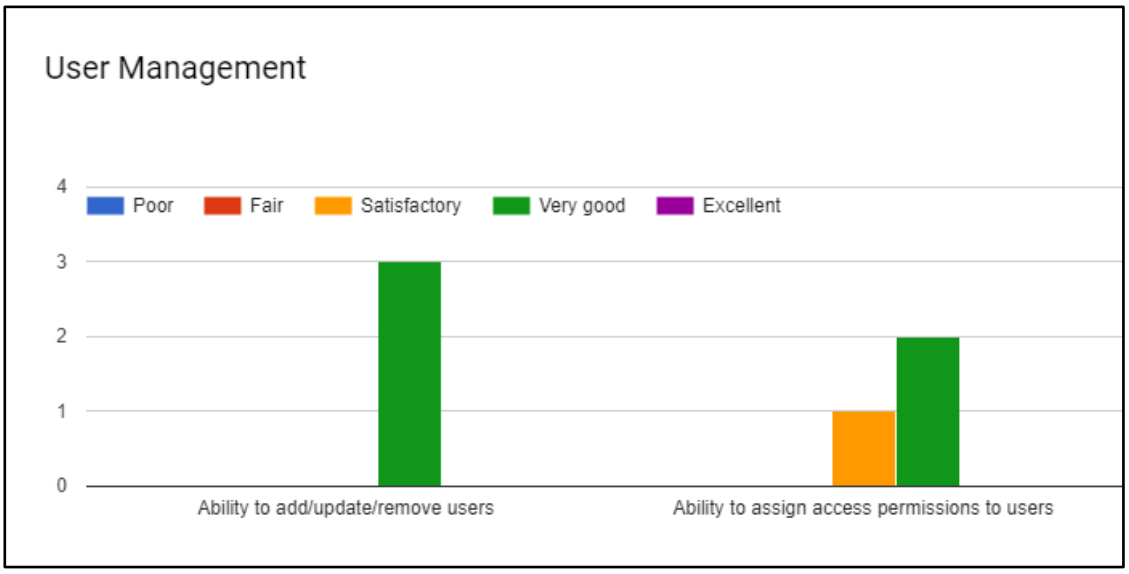


***Figure 6.2**: Summary of User Feedbacks on User Management Functionality*

A summary of the user evaluation on production line modeling functionality is shown in Figure 6.3. All the users have agreed that the Production Line Editor facilitates modeling a new production line with dragging and dropping components, updating a modeled production line and adding custom components, beyond the satisfactory level. This is an important point which shows that the users have understood and accepted the novel approach of production line modeling suggested with Hitech Smart Factory.

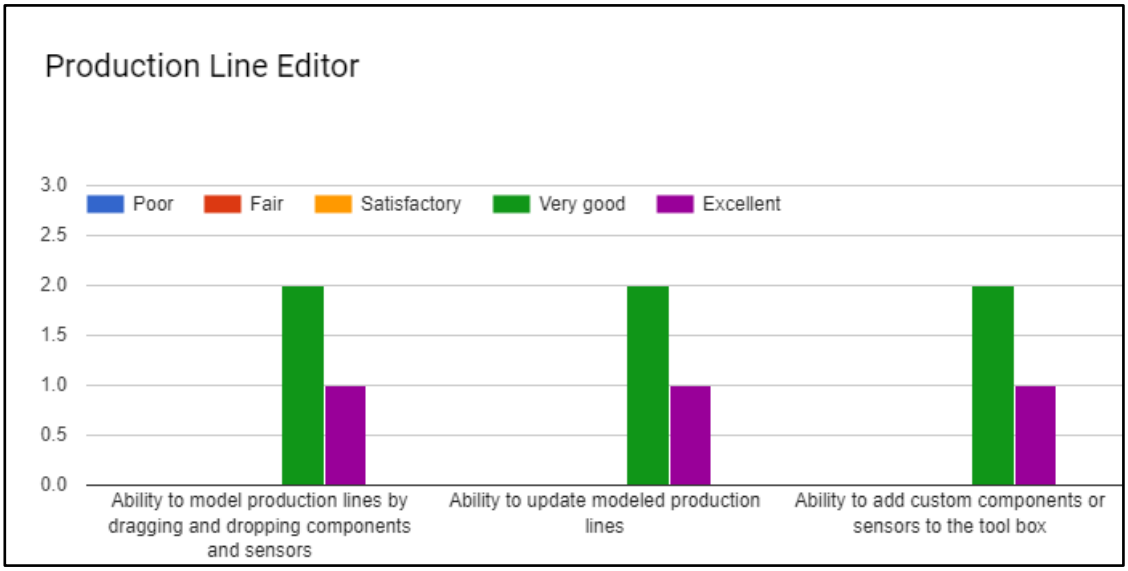


***Figure 6.3**: Summary of User Feedbacks on Production Line Editor*

The summary of the user feedbacks on viewing real-time status functionality is shown in Figure 6.4. It reflects that users have accepted that this functionality is up to the satisfactory level. However, this component could use further refinements to be evaluated as above the satisfactory level. More usability tests are needed to determine exact points to be changed.

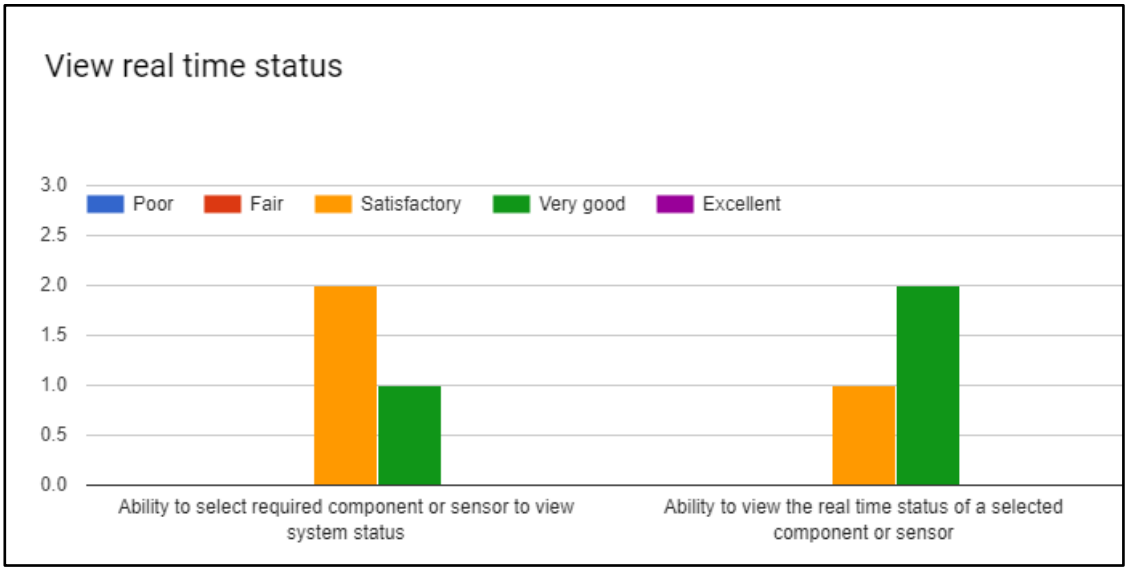


***Figure 6.4**: Summary of User Feedbacks on Real Time Status View Functionality*

The summary of user evaluation on the report generation tool is displayed in Figure 6.5. The functionality is marked as it is up to the satisfactory level. A user has also commented that the reports can be further extended in order to match the business requirements of the client. This component is also identified as a modifiable item in order to receive more user satisfaction.

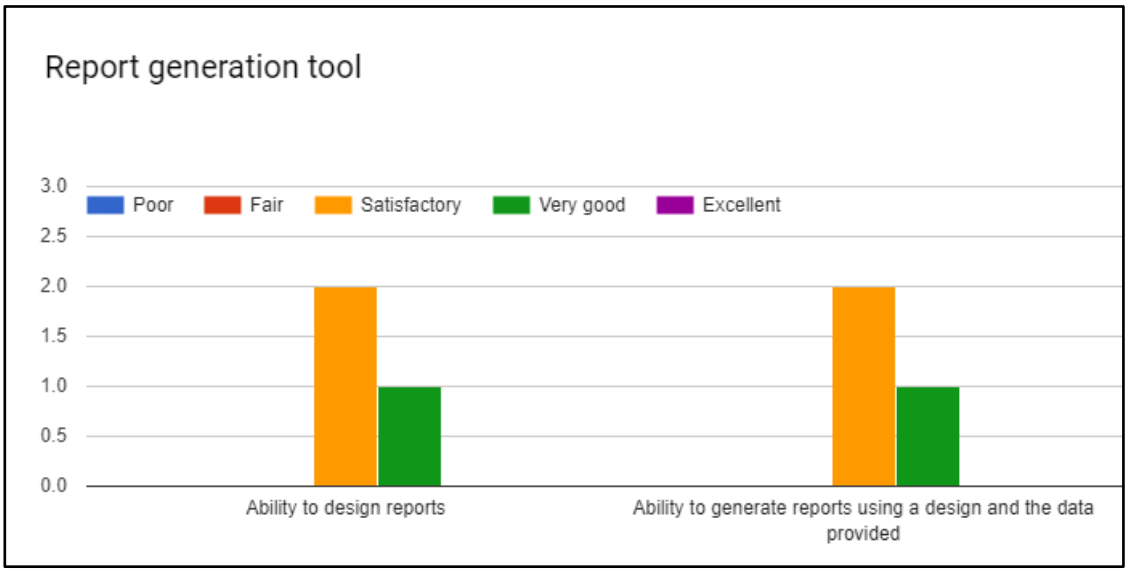


***Figure 6.5**: Summary of User Feedbacks on Report Generation Tool*

Figure 6.6 shows the summarized version of user feedback on the overall product. Users have agreed that user-friendliness of user interfaces and the understandability of the overall process is beyond the satisfactory level. Also, they have found that the usability of the system in a real-life scenario is also up to the satisfactory level. Furthermore, some of them have commented that user interfaces should be improved in order to provide a better look and feel to the users.


***Figure 6.6**: Summary of User Feedbacks on the Overall Product*

## 6.3 Limitations

The proposed solution collects data from PLC units using the Data Publisher Modules, which needs to be run in a processing environment that is physically connected to those PLCs. In the initial setup, a computer is used for this purpose. This can be seen as a limitation to benefit from the full capacity of Hitech Smart Factory.

After the modeling of production lines is completed using the Production Line Editor, graphical components have to be configured manually providing the tag values produced, to the Grafana UI Server. This can also be identified as a limitation.

## 6.4 Implications for Future Work

As a remedy to the limitation mentioned above in Section 6.3 about using a physical device to run Data Publisher Module, a small, sophisticated device that is optimized to run java programs can be used instead of a computer. Furthermore, there are PLC units that are capable of directly connecting to the internet using WiFi network connections and using those PLC units can be seen as another solution. However, further studies need to be carried out in these areas to find an optimal solution.

As for the limitation of configuring Grafana UI Components manually, Grafana comprises of a JavaScript Object Notation (JSON) based configuration mechanism which is capable of representing all graphical components. However, further studies are needed to clarify the feasibility of using these configurations to the requirement of Hitech Smart Factory.

# References


[1] Lichtigstein, "Prometheus vs. Grafana vs. Graphite - A Feature Comparison," *Loom Systems*, 2017. [Online]. Available: https://www.loomsystems.com/blog/single-post/2017/06/07/prometheus-vs-grafana-vs-graphite-a-feature-comparison. [Accessed: 31- Jul- 2017].

[2] Amazon Web Services, Inc., “Lambda Architecture for Batch and Real- Time Processing on AWS with Spark Streaming and Spark SQL,” *Amazon Web Services, Inc.*, May 2015, [online] Available: https://d0.awsstatic.com/whitepapers/lambda-architecure-on-for-batch-aws.pdf. [Accessed: 31- Jul- 2017].

[3] "Apache Kafka - A distributed Streaming Platform," *Kafka Apache*, 2017. [Online]. Available: https://kafka.apache.org/. [Accessed: 31- Jul- 2017].

[4] A. Saldhi, D. Yadav, D. Saksena, A. Goel, A. Saldhi and S. Indu, "Big data analysis using Hadoop cluster," 2014 IEEE International Conference on Computational Intelligence and Computing Research, Coimbatore, 2014, pp. 1-6.

[5] B-Scada, Inc.,"Status Enterprise," *B-Scada, Inc.*, 2017. [Online]. Available: http://scada.com/Content/Brochures/Brochure_SE.pdf. [Accessed: 31- Jul- 2017].

[6] Iman,"The Advantages of WinTr System," *program-plc*, August 24, 2016. [Online]. Available: https://program-plc.blogspot.com/2016/08/the-advantages-of-wintr-system.html. [Accessed: 30- Jul- 2017].

[7] A. Norman, “The ‘problem’ of automation: Inappropriate feedback and interaction, not ‘over-automation’,” Philosophical *Transactions of the Royal Society of London. Series B, Biological Sciences*, vol. 327, no. 1241, pp. 585-593, April 1990.

[8] Soltesz, "The Advantages of a Relational Database Management System," *Techwalla*, 2017. [Online]. Available: https://www.techwalla.com/articles/the-advantages-of-a-relational-database-management-system. [Accessed: 31- Jul- 2017].

[9] E.yu,"Korea's UlalaLAB Equipped with IoT Technology Enters a MOU Agreement with China's Government-Affiliated Organization," *Aving.net*, May 2016. [Online]. Available: http://us.aving.net/news/view.php?articleId=1369434. [Accessed: 25- Sep- 2017].

[10] Fultek Kontrol Sistemleri,"SCADA SOFTWARE – SCADA SYSTEMS," *Fultek Kontrol Sistemleri*, 2017. [Online]. Available: https://www.fultek.com.tr/en/scada/. [Accessed: 30- Jul- 2017].

[11] "InfluxDB," *Influx Data*, 2017. [Online]. Available: https://www.influxdata.com/time-series-platform/influxdb/. [Accessed: 31- Jul- 2017].

[12] “Introduction to Real time Big Data processing architectures - Lambda, Kappa and Zeta,” *CA Technologies*, Mar. 03, 2016. [Online]. Available: https://communities.ca.com/community/ca-apm/blog/2016/03/03/real-time-big-data-processing-architectures-lambda-kappa-and-zeta. [Accessed 22 Dec. 2017].

[13] J. Kreps, N. Narkhede and J. Rao, "Kafka: A Distributed Messaging System for Log Processing,” in *ACM SIGMOD/PODS Conference 2011*, Athens, Greece, June 12-16, 2011.

[14] M. Kiran, P. Murphy, I. Monga, J. Dugan and S. S. Baveja, "Lambda architecture for cost-effective batch and speed big data processing," 2015 IEEE International Conference on Big Data (Big Data), Santa Clara, CA, 2015, pp. 2785-2792.

[15] "POST (HTTP)," *Wikipedia*, 2017. [Online]. Available: https://en.wikipedia.org/wiki/POST_ (HTTP). [Accessed: 31- Jul- 2017].

[16] "Programmable Logic Controller," *Wikipedia*, 2017. [Online]. Available: https://en.wikipedia.org/wiki/POST_ (HTTP). [Accessed: 31- Jul- 2017].

[17] R.T.Fielding and R.N. Taylor, “Architectural styles and the design of network-based software architectures,” Ph.D. dissertation, University of California, Irvine, 2000.

[18] "Reportico User Manual Dev Version," Repotico, 2016. [Online]. Available: http://www.reportico.org/documentation/4.5/doku.php?id=reporticointro. [Accessed: 31- Jul- 2017].

[19] S. Acreman, "Top 10 Time Series Databases," *Outlyer*, 2016. [Online]. Available: https://blog.outlyer.com/top10-open-source-time-series-databases. [Accessed: 31- Jul- 2017].

[20] "Scada explained," *Inductive Automation*, 2017. [Online]. Available: https://inductiveautomation.com/what-is-scada. [Accessed: 31- Jul- 2017].

[21] T. Goldschmidt, A. Jansen, H. Koziolek, J. Doppelhamer and H. P. Breivold, "Scalability and Robustness of Time-Series Databases for Cloud-Native Monitoring of Industrial Processes," 2014 IEEE 7th International Conference on Cloud Computing, Anchorage, AK, 2014, pp. 602-609.

[22] UlalaLAB,Inc.,"WimFactory," *Wimfactory.com*, 2016. [Online]. Available: http://www.wimfactory.com/index_v2_en.html. [Accessed: 30- Jul- 2017].

[23] U.Shafaque and P. D. Thakare,"Algorithm and Approaches to Handle Big Data," *IJCA Proceedings on National Level Technical Conference X-PLORE 2014,* vol. XPLORE2014, pp. 18-22, *May* 2014.

[24] W.Wingerath, F.Gessert and S. Friedrich,"Real-time stream processing for Big Data," *it - Information Technology*,vol 54, no 4, pp 186–194, August 2016.

[25] Xinje Electronic Co. Ltd. , "XC Series Programmable Controller-User's Manual," *Imenista Andish Ltd*, 2009, [Online]. Available: http: //www.imenista.com/pdf/XCPLCV2.51.pdf. [Accessed: 31- Jul- 2017].

[26] Xylem Inc.,"AQUAFORCE – MODBUS COMMUNICATIONS PRIMER," *Xylem Inc*, 2012. [Online]. Available: http://documentlibrary.xylemappliedwater.com/wp-content/blogs.dir/22/files/2012/07/aqforwhite.pdf. [Accessed: 31- Jul- 2017].

# Appendix

## A.1 Use Case Diagrams of the System

Functional requirements of a manager are shown in Figure A.1.1**.** A manager can generate, view and print reports using from the system.

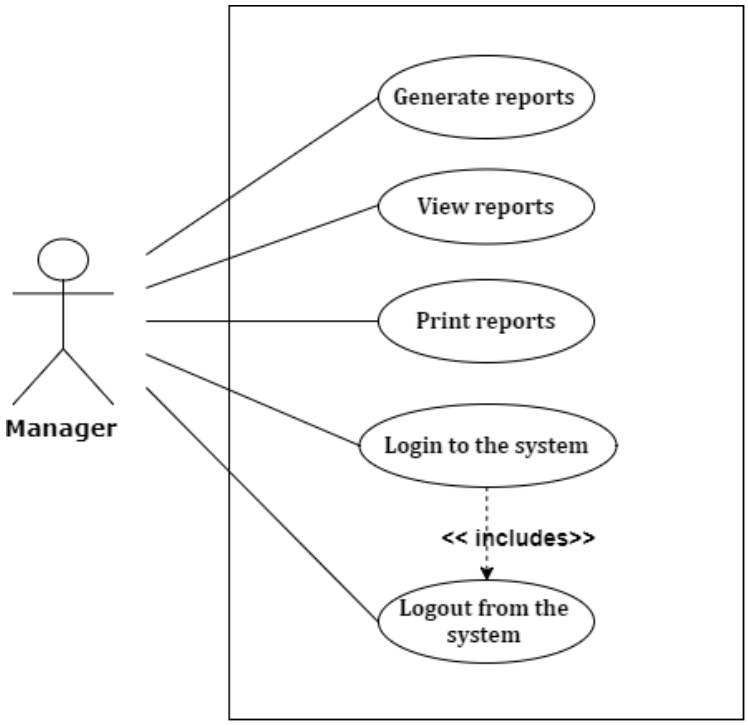


*Figure A.1.1 : Use case Diagram of Manager*

The functional requirements of engineer and the machine operator are shown in Figure A.1.2. They both can view the real time status using the web application and the mobile application.

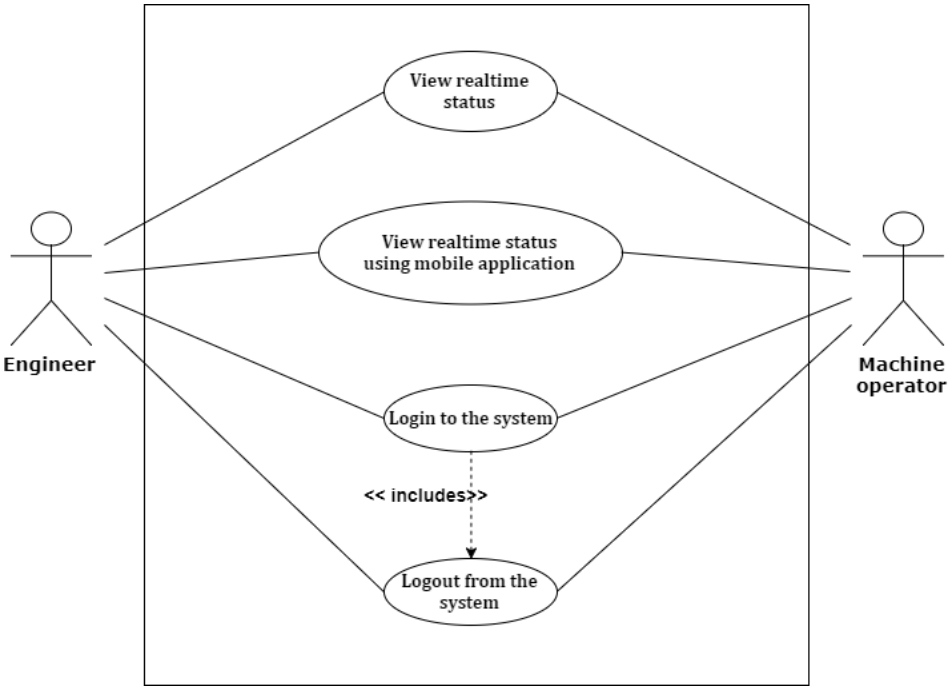


*Figure A.1.2 : Use case Diagram of Engineer and Machine Operator*

The functional requirements of *System Admin* and *Factory Admin* are shown in Figure A.1.3. Both of them have same functionalities except for creating and deleting factories. Only the System Admin is capable of performing those functionalities. Factory admin can create, view, update and delete branches, sections, production lines of the factory he is assigned to. User management can be done by both of them.

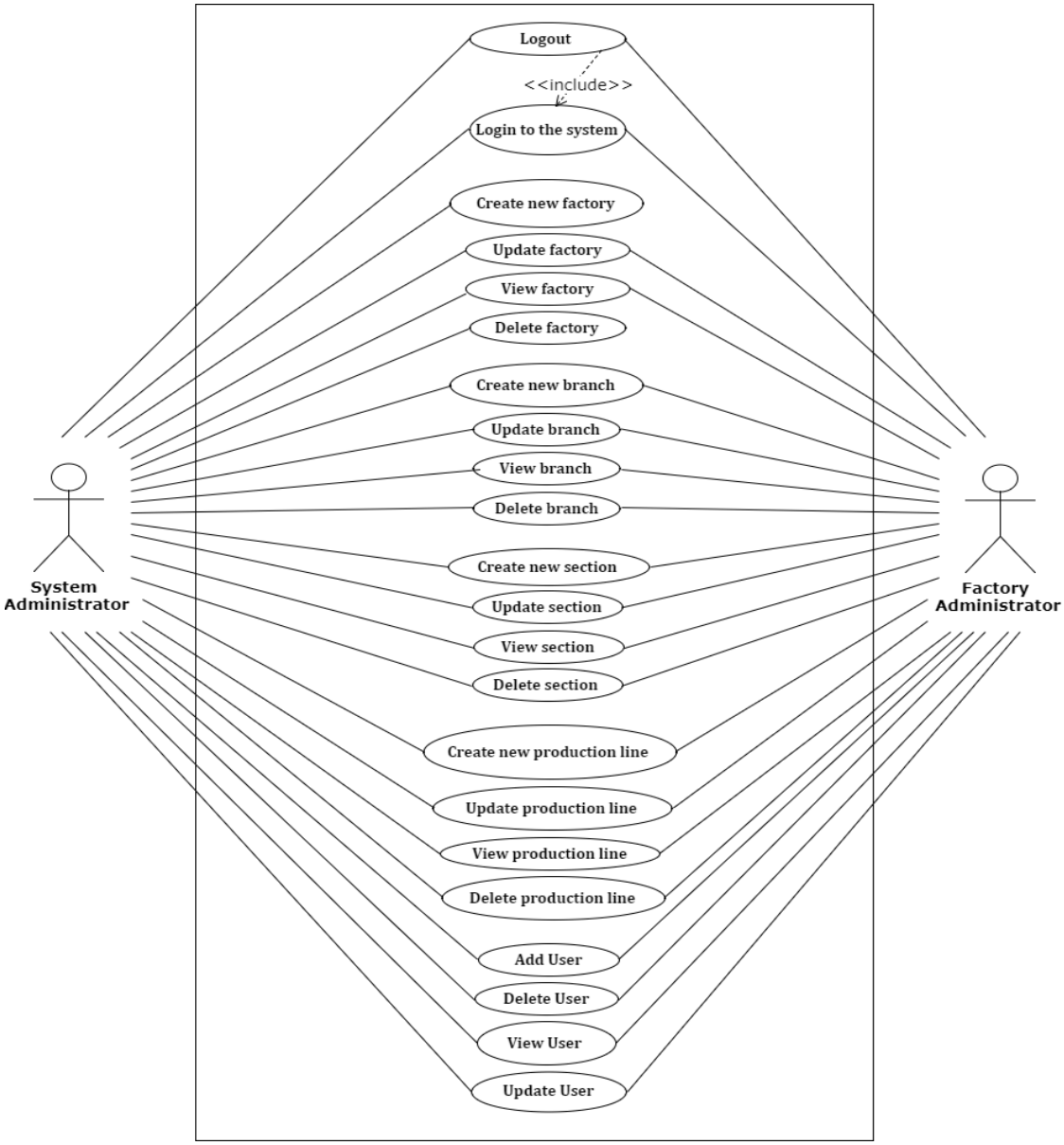


*Figure A.1.3: Use case Diagram of System Admin and Factory Admin*

## A.2 Code Snippet used to enforce user permissions

```
$("input:checkbox").on('click', function () {
   var checkID = $(this)["0"].id;
   var checkDivID = checkID.substring(0, checkID.length - 5) + 'div';
   var type = checkID.split('-')[0];

   // update variables, disable checkboxes as necessary
   if ($(this).prop("checked")) {
       if (type == 'factory') {
           $('#' + checkDivID + ' :input').prop("disabled", true);
           $('#' + checkDivID).find('input[type=checkbox]').prop('checked', true);
           CHECKED_BRANCHES = [];
           CHECKED_SECTIONS = [];
           CHECKED_PRODLINES = [];
       } else if (type == 'branch') {
           $('.sections :input').prop("disabled", true);
           CHECKED_BRANCHES.push(checkDivID);
           CHECKED_SECTIONS = [];
           CHECKED_PRODLINES = [];
           checkChildren(CHECKED_BRANCHES);
       } else if (type == 'section') {
           $('.prodlines :input').prop("disabled", true);
           CHECKED_SECTIONS.push(checkDivID);
           CHECKED_PRODLINES = [];
           checkChildren(CHECKED_SECTIONS);
       } else if (type == 'prodline') {
           CHECKED_PRODLINES.push(checkDivID);
       }
   } else {
       if (type == 'factory') {
           $('#' + checkDivID + ' :input').prop("disabled", false);
       } else if (type == 'branch') {
           CHECKED_BRANCHES.splice(CHECKED_BRANCHES.indexOf(checkDivID), 1);
           if (CHECKED_BRANCHES.length == 0)
               $('.sections :input').prop("disabled", false);
       } else if (type == 'section') {
```

```
            CHECKED_SECTIONS.splice(CHECKED_SECTIONS.indexOf(checkDivID), 1);
            if (CHECKED_SECTIONS.length == 0)
                $('.prodlines :input').prop("disabled", false);
        } else if (type == 'prodline') {
            CHECKED_PRODLINES.splice(CHECKED_PRODLINES.indexOf(checkID), 1);
        }
        $('#' + checkDivID).find('input[type=checkbox]').prop('checked', false);
    }
});
```

## A.3 Individual Contribution

In accordance to the methodology discussed in Section 1.4 in this dissertation, all the three team members were equally contributed in the initial steps of studying existing similar systems and related research areas in order to get an overall idea about the domain. Developing the design of proposed solution also obtained by the equal contributions of all group members. Then the implementation of the proposed design is conducted by the members as described below in the topics, from Contribution 1 to Contribution 3. The work carried out by each member is illustrated with respect to the components in the System Architecture, discussed in Section 3.5.

### A.3.1 Contributions by V. D. Liyanage

Following are the implementations carried out by V D Liyanage.

- Client Database discussed in Subsection 4.2.8
- System Administrator Backend discussed in Subsection 4.2.9
- Admin Dashboard discussed in Subsection 4.2.10
- User Manager User Interface discussed in Subsection 4.2.11
- Production Line Editor discussed in Subsection 4.2.12
- File Server discussed in Subsection 4.2.13

- Web application component of the User Authentication discussed in Subsection 4.2.16
- Web User Dashboard discussed in Subsection 4.2.17
- Web application component of Production Line View described in Subsection 4.2.19
- Web application component of Real-Time Status View discussed in Subsection 4.2.20

### A.3.2 Contributions by H. M. P. M. Karunarathne

Following are the implementations carried out by H. M. P. M. Karunarathne.

- Production Line Server discussed in Subsection 4.2.14
- Mobile application components of User Authentication with configuration and development of WSO2 IS discussed in Subsection 4.2.16
- Mobile application components of User Dashboard discussed in Subsection 4.2.17
- Configuration and development of WSO2 DSS described in Subsection 4.2.18
- Mobile application components of Production Line View discussed in Subsection 4.2.19
- Mobile application components of Real-Time Status View discussed in Subsection 4.2.20

### A.3.3 Contributions by D D Mathangaweera

Following are the implementations carried out by D D Mathangaweera.

- Data Publisher described in Subsection 4.2.1
- Kafka Cluster described in Subsection 4.2.2
- Long-Term Database described in Subsection 4.2.3
- Report Generation Server described in Subsection 4.2.4
- Report View User Interface described in Subsection 4.2.5
- Time Series Database described in Subsection 4.2.6
- Grafana User Interface Server described in Subsection 4.2.7