
Asian Institute of Technology
School of Engineering and Technology
AT70.19: Software Development and Quality Improvement



***“AIR QUALITY (PM_{2.5}) MONITORING SYSTEM”
(IOT-BASED PLATFORM)***

***Software Requirements Specification
(SRS) Document***

Submitted by:

Smrity Baral (st121662)

Shubhangini Gontia (st121473)

Suyogya Ratna Tamrakar (st121334)

Younten Tshering (st121775)

Submitted to:

Dr. Apichon Witayangkurn

Submitted Date:

3rd March 2021

Table of Contents

1. Introduction.....	3
1.1 Introduction.....	3
1.2 Problem Statement	3
1.2 Scope.....	4
1.3 Overview.....	4
2. Functional Requirement.....	5
2.1 Visualization Module.....	5
2.2 System Admin Module	5
2.3 Data Collection Module.....	5
3. Non-functional Requirements	5
3.1 Performance	5
3.2 Reliability.....	6
3.3 Interoperability.....	6
3.4 Portability.....	6
3.5 Scalability	6
3.6 Reusability	6
3.7 Serviceability	6
4. Preliminary System models	7
4.1 Use Case Model	7
4.2 Dynamic models	8
4.2.1 Sequence Diagram	8
4.2.2 State diagram	9
4.2.3. Activity diagram	10
4.3 Object and class model	11
5. User interface - navigational paths and screen mock-ups.....	11
5.1 Interface for Visitors	12
5.2 Interface for admin.....	12
5.3 Admin Dashboard	13
5.4 Sensors Management	14
5.5 Reports	14
5.6 Logout.....	15

5.7 Update Profile	15
5.8 Parameter Settings	16
6. Risk and mitigation plan	17
7. Planned Schedule	20
8. Details on software requirement - Apache Kylin	21
9. Glossary & references	21
9.1 Glossary	21
9.2 References	22

1. Introduction

1.1 Introduction

The airborne PM_{2.5} has returned to become a serious issue since the start of 2020. The thickening smog, exceeding the standard safe level, prompted The Royal Thai government to approve measures to prevent and address the on January 21, 2020.

PM 2.5 comes primarily from combustion - Fireplaces, car engines, and coal- or natural gas-fired power plants are all major PM 2.5 sources. PM stands for particulate matter (also called particle pollution): the term for a mixture of solid particles and liquid droplets found in the air. Some particles, such as dust, dirt, soot, or smoke, are large or dark enough to be seen with the naked eye.

Fine particulate matter (PM_{2.5}) is an air pollutant that is a concern for people's health when levels in air are high. PM_{2.5} are tiny particles in the air that reduce visibility and cause the air to appear hazy when levels are elevated.

To measure this level in air, sensor PM_{2.5} (PM_{2.5} refers to particles that are 2.5 microns or smaller in diameter) can be placed at different places where the chances of such issues. This sensor uses laser scattering to radiate suspending particles in the air, then collects scattering light to obtain the curve of scattering light change with time.

1.2 Problem Statement

In this contemporary environment air pollution or air quality have become a big concern as the quality is degrading exponentially. One of which is the PM_{2.5} crisis where the size of particles is directly linked to their potential for causing health problems. Fine particles (PM_{2.5}) pose the greatest health risk. These fine particles can get deep into lungs and some may even get into the bloodstream. Exposure to these particles can affect a person's lungs and heart.

On a very clear and non-hazy day, the PM_{2.5} concentration can be as low as 5 µg/m³ or below. But the PM_{2.5} is considered unhealthy when it rises above 35.4 µg/m³.

Our motive for this project is to analyze the air quality data from the sensor and give a clear vision in a way of dashboard. Therefore, we are going to develop the web app which will show all the required details related to air quality and accordingly people can take preventive measures.

For this we will receive sensor data from nodes or stations with different types of data such as text value and number and process using Big Data Tool.

1.2 Scope

The users for the system are the students and employees (staff and faculties) of AIT, including the Administrator (management team), and the developers of the system. In case of AIT, this application can be used to monitor the air quality on campus. Once the application can give accurate and real time information on air quality from the data imported from sensors then other investors or different organizations can use it.

1.3 Overview

Air Quality is a big question in these times and to reduce the air pollution, we are looking forward to making an air quality monitoring system. This system will be taking the data from sensors such as PM2.5. This web application will be using Hadoop and Apache Kylin as a backend to hold a huge amount of data from sensors and they will be processed with BI tools and different important insights are visualized using an interactive dashboard. As of now, we will be starting with Java Spring Boot framework for developing the system.

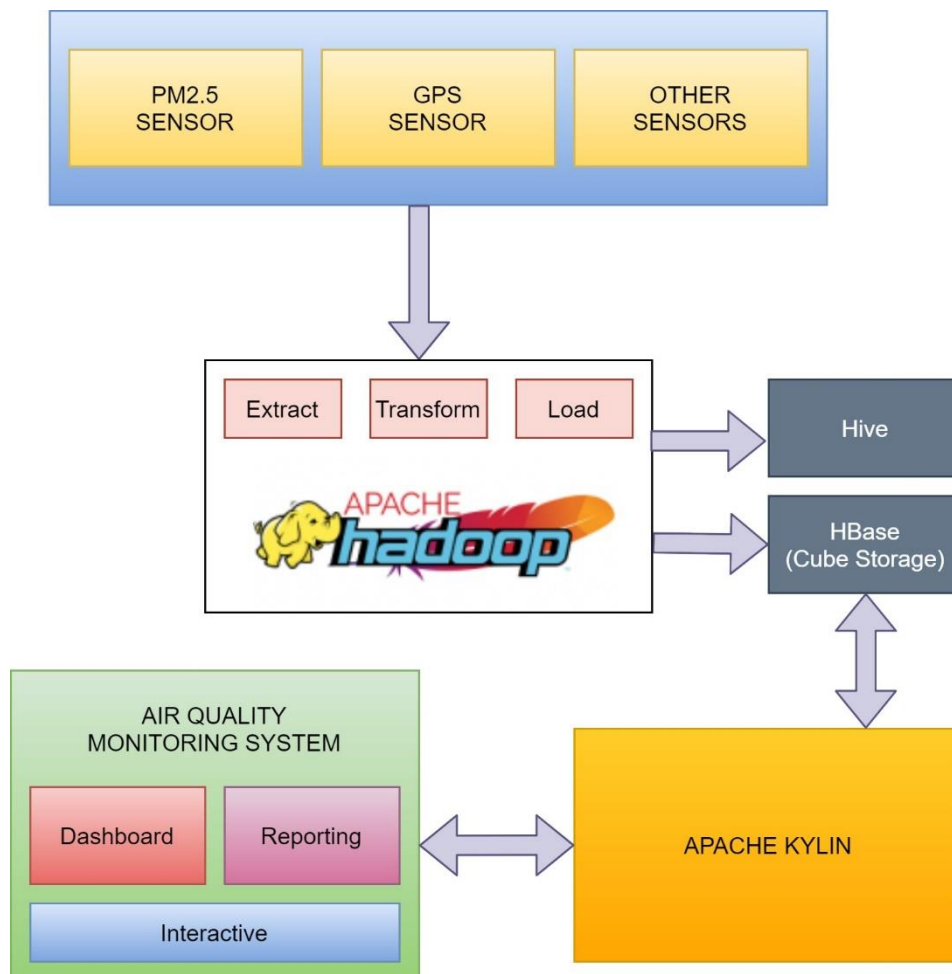


Figure 1. System Architecture

2. Functional Requirement

2.1 Visualization Module

- The end users should be able to see an interactive dashboard of air quality monitoring with different forecasts and insights.
- The system should be able to stream real-time data from different nodes and stations.

2.2 System Admin Module

- The admin should be able to login, logout to the system and modify the system parameters and toggle dashboard controls.
- The admin should be able to register new sensors and manage the sensors in system.
- The admin should be able to generate reports of specific time periods and export those in various formats.

2.3 Data Collection Module

- Data will be extracted from sensors and stored in Hadoop which is acting as data warehouse using Hive.
- Kylin does aggregation functions on cube (HBase) and provide the required parameters to the system.

Note:

- ❖ Air quality and location details provided to the system from the sensor shall be stored in the database (Hadoop).
- ❖ The information shall be accessible via distributed system and JPA (Java Persistence API).
- ❖ The data stored should be able to be manipulated through interface.

3. Non-functional Requirements

3.1 Performance

The system shall be designed with high performance level to handle concurrent or multiple information from sensors which are placed at different locations in real time. Large numbers of data collected from the sensor should be displayed on the responsive web application and we need to focus on concurrency, response time and block time.

3.2 Reliability

Reliability is one of the key attributes of the system. Back-ups will be made regularly so that restoration with minimal data loss is possible in the event of unforeseen events. The system will also be thoroughly tested by all team members to ensure reliability.

3.3 Interoperability

For the application, we just import and integrate various information with different values into the system from the sensors. Therefore, we need to know to segregate the data and proceed forward.

3.4 Portability

The system shall be designed in a way that allows it to be run on multiple computers with different browsers. As it is a web application, mobile phone web browsers can also access the application.

3.5 Scalability

With data, the storage size will increase but can be managed with time. This app can be made horizontally scalable when there are issues of memory storage.

3.6 Reusability

The system should be designed in a way that allows the database to be re-used regularly for the various similar sensors that the organization shall hold.

3.7 Serviceability

Once the system is deployed, the maintenance of the system including tasks such as monitoring the system, repairing problems that arise, updating or upgrading software components, should be easy to be sufficiently performed by any person with a basic understanding of the dashboard system.

4. Preliminary System models

This section presents a list of the fundamental sequence diagrams and use cases that satisfy the system's requirements. The purpose is to provide an alternative, "structural" view of the requirements stated above and how they might be satisfied in the system.

4.1 Use Case Model

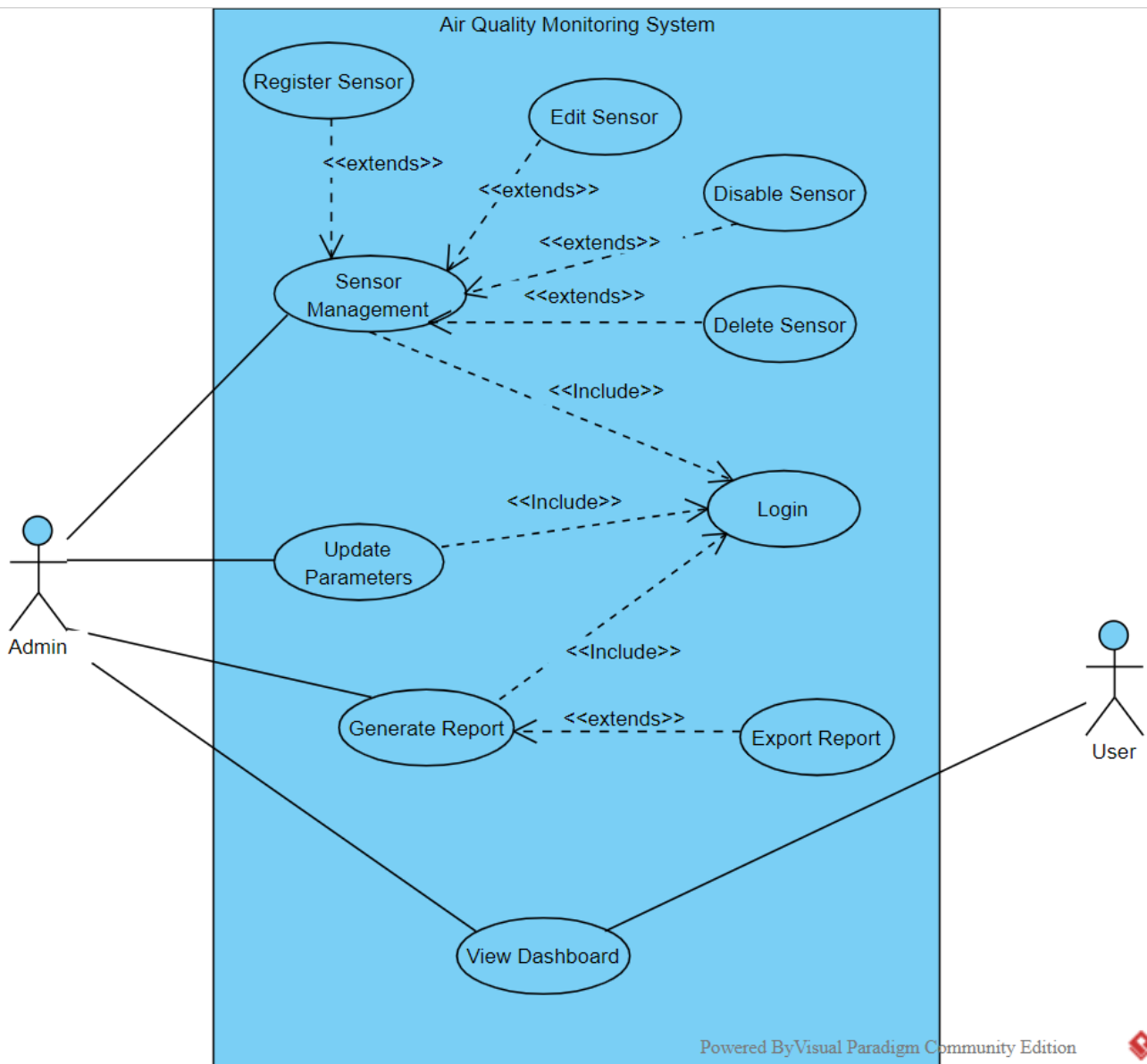


Figure 2. Use case diagram.

4.2 Dynamic models

4.2.1 Sequence Diagram

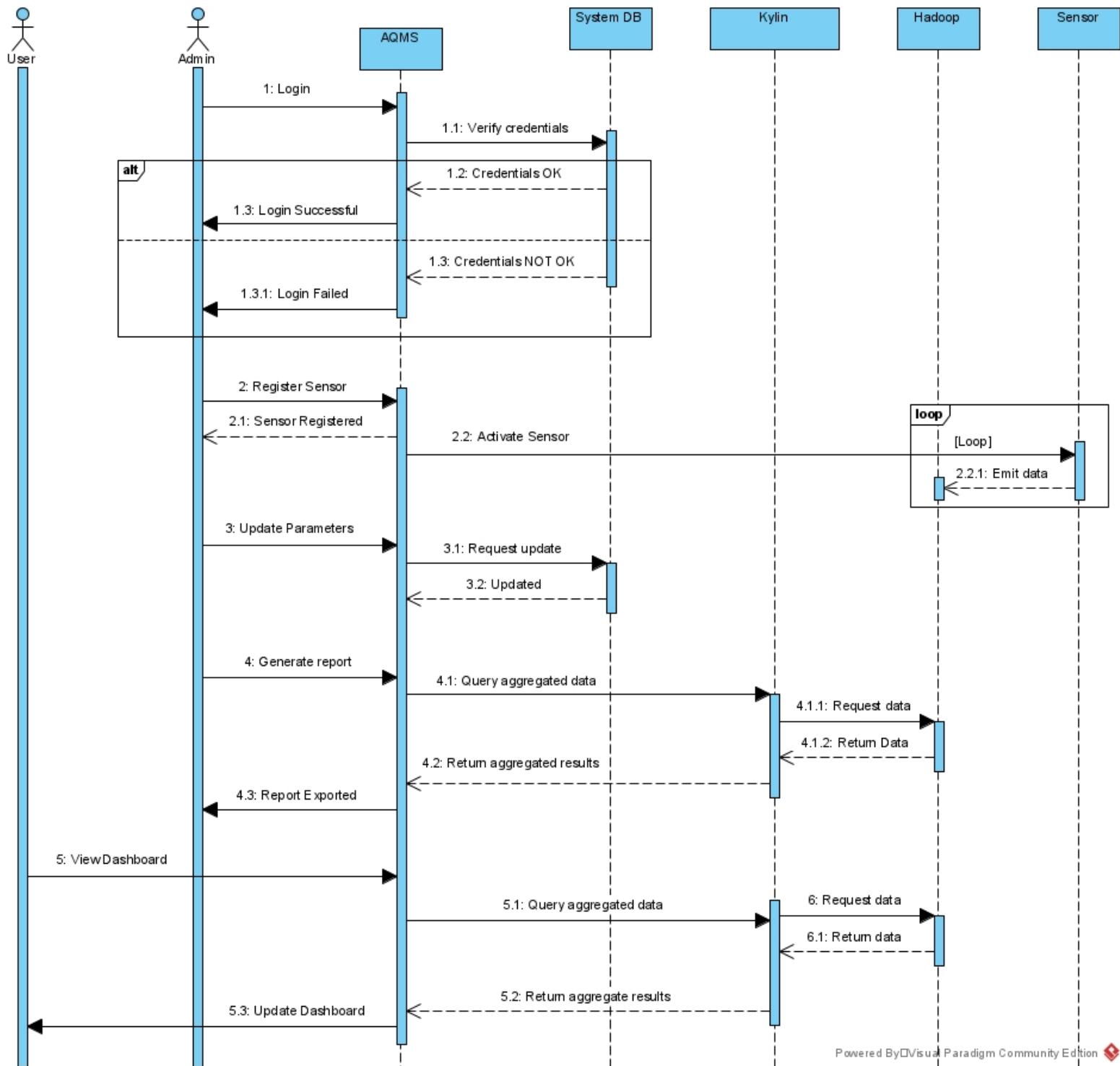


Figure 3. Sequence diagram

4.2.2 State diagram

a) State diagram for user with internal process.

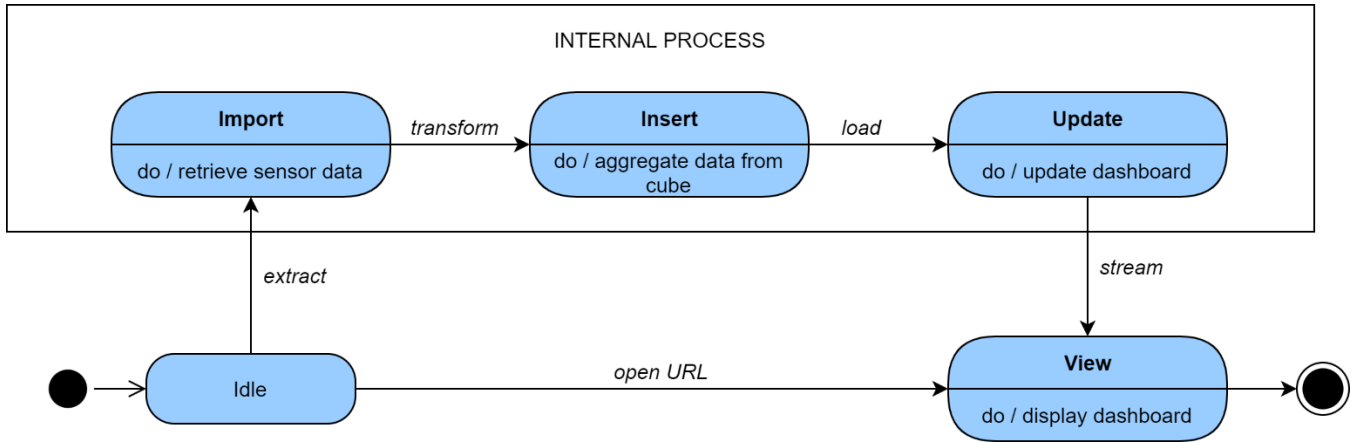


Figure 4. User State diagram

b) State diagram for admin with different state.

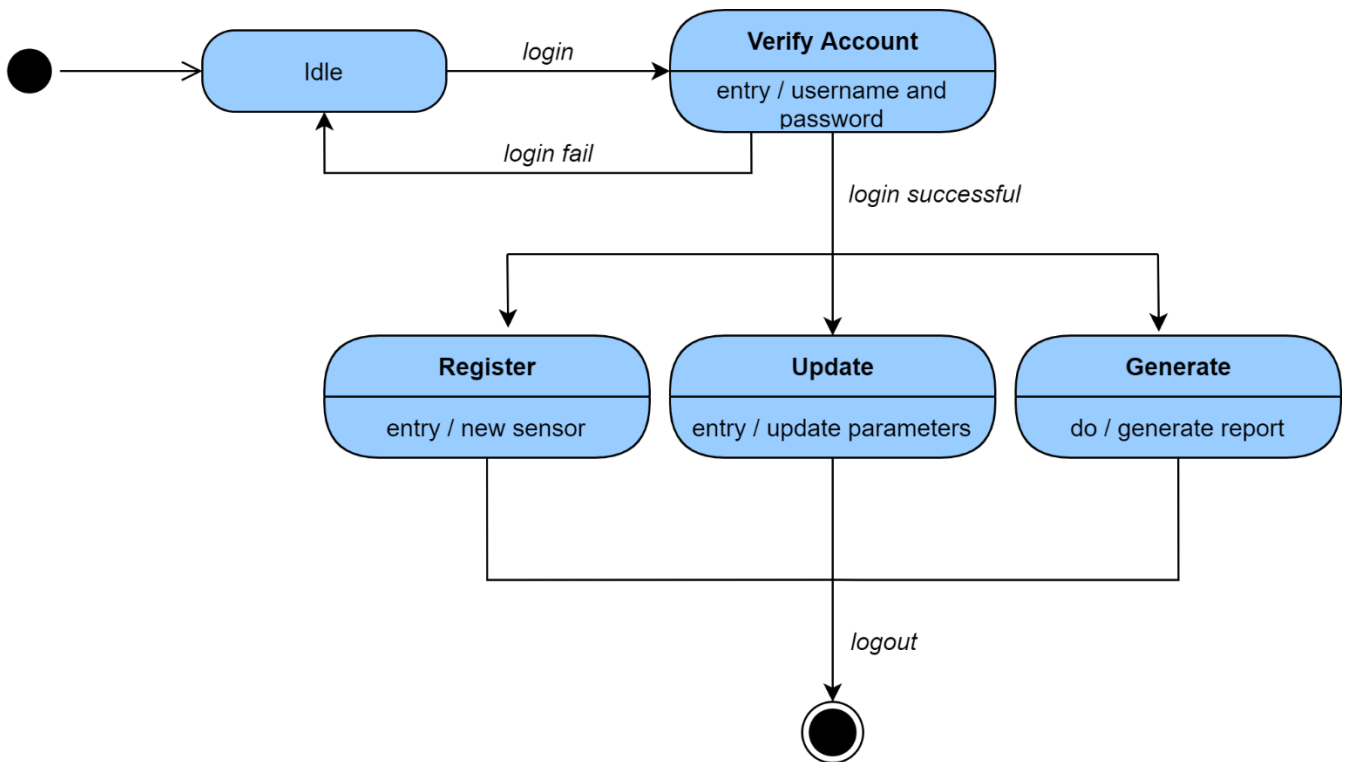


Figure 5. Admin State diagram

4.2.3. Activity diagram

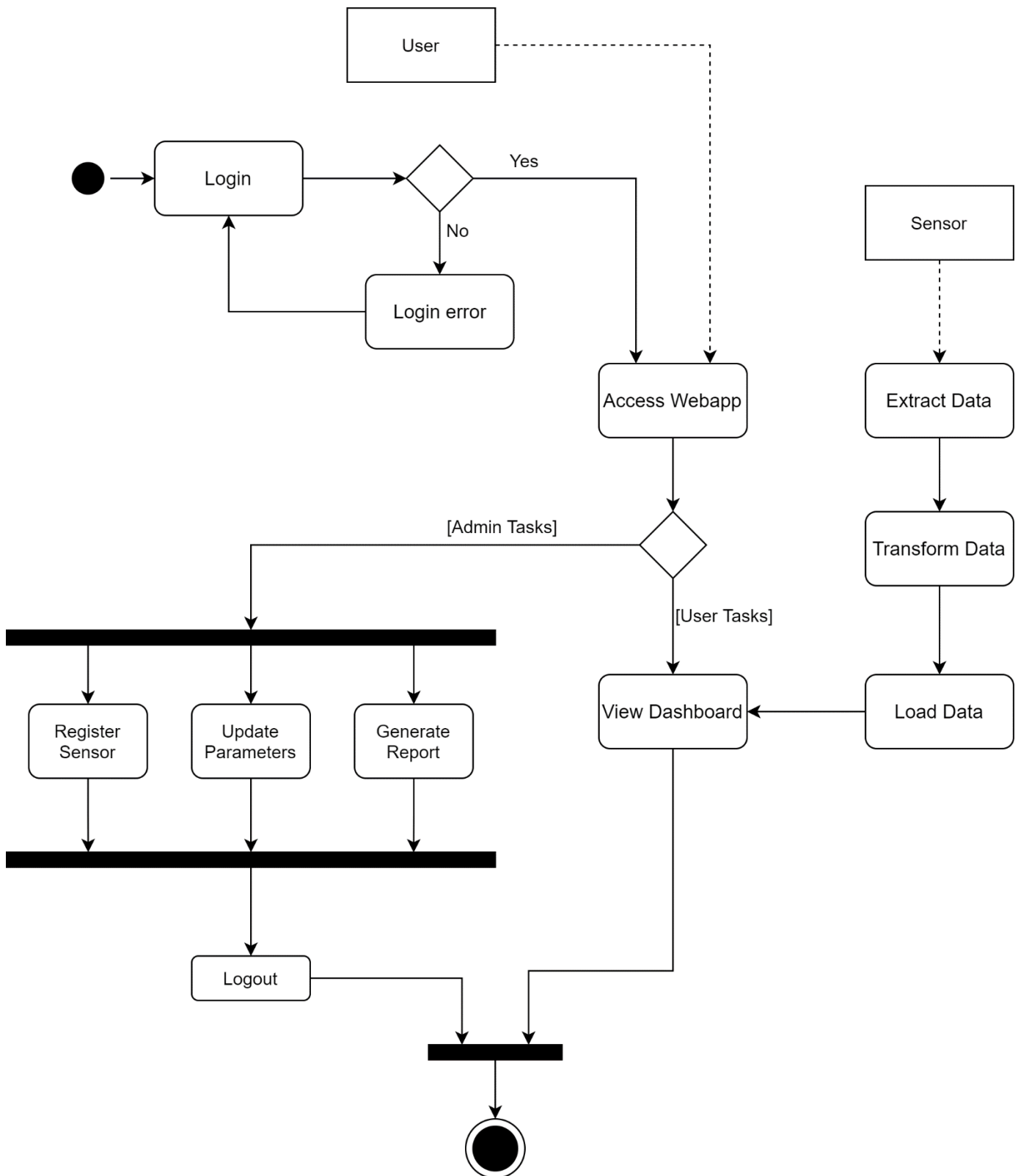


Figure 6. Activity diagram

4.3 Object and class model

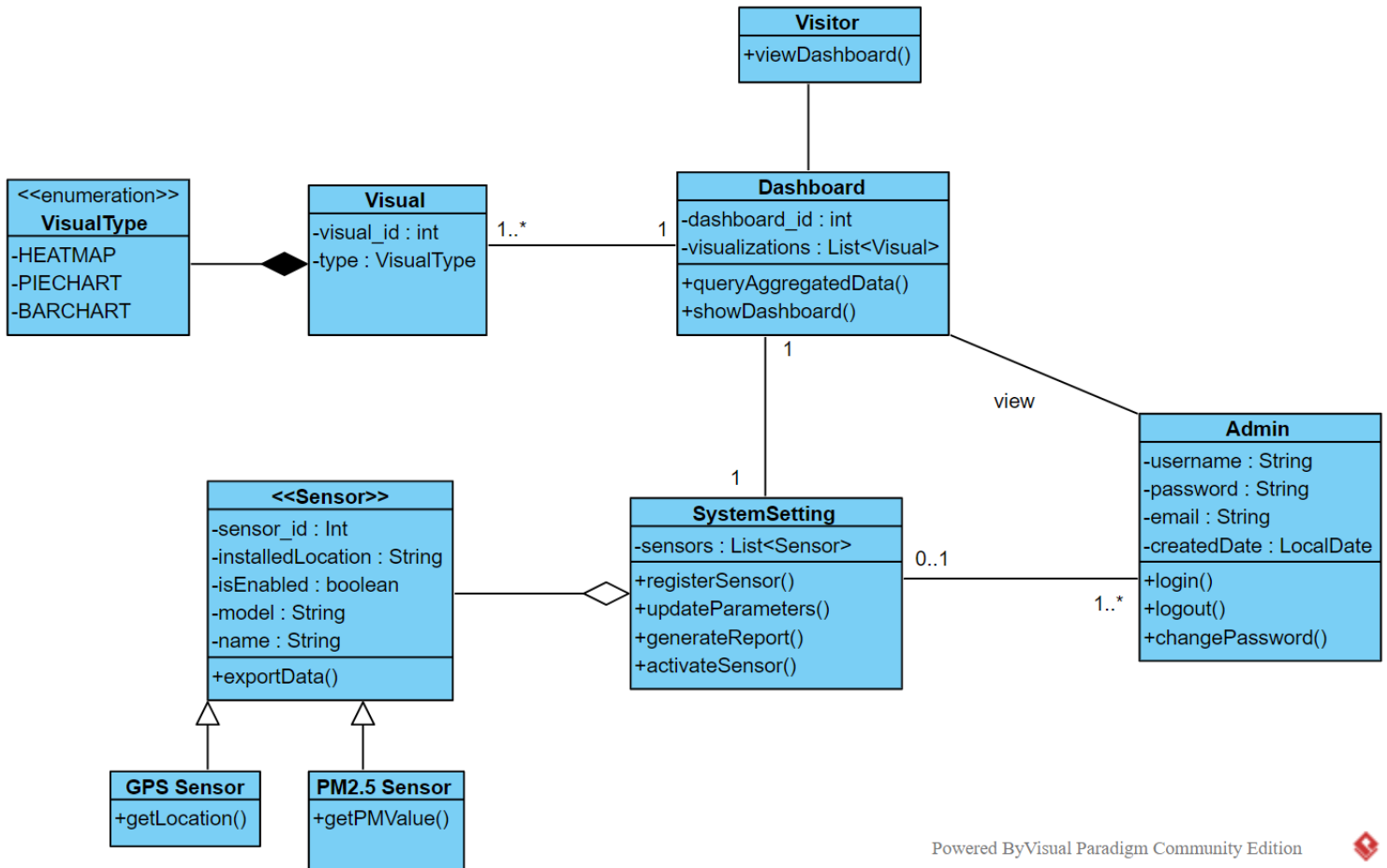


Figure 7. Initial class diagram

5. User interface - navigational paths and screen mock-ups

Our system will be a web-based application system that visualizes the important details about the PM 2.5 to give clear vision on air quality of the selected location.

Air quality monitoring web-app is expected to be user-friendly with easy access of the features, as we will be using a navigation containing each component allowing the user to interact with the application in a natural and intuitive way.

For our system there are two users: Admin who will have privilege to manage the website, control sensors, make changes to the website as well as generate reports and next is Visitor who will have access to dashboard only.

5.1 Interface for Visitors

For guest users, once the URL is entered, they will be able to see the air quality monitoring system webpage. This webpage will include a dashboard that contains descriptive analysis of the air quality of the place based on PM2.5. This will give a quick view of the detail of the webpage. The option to choose location will be provided. Visualization can be about the current situation of air quality or for the given time, within the selected location such as pollution level, level of PM2.5, level of quality of air, etc. The mockup screen for visitor is given below:

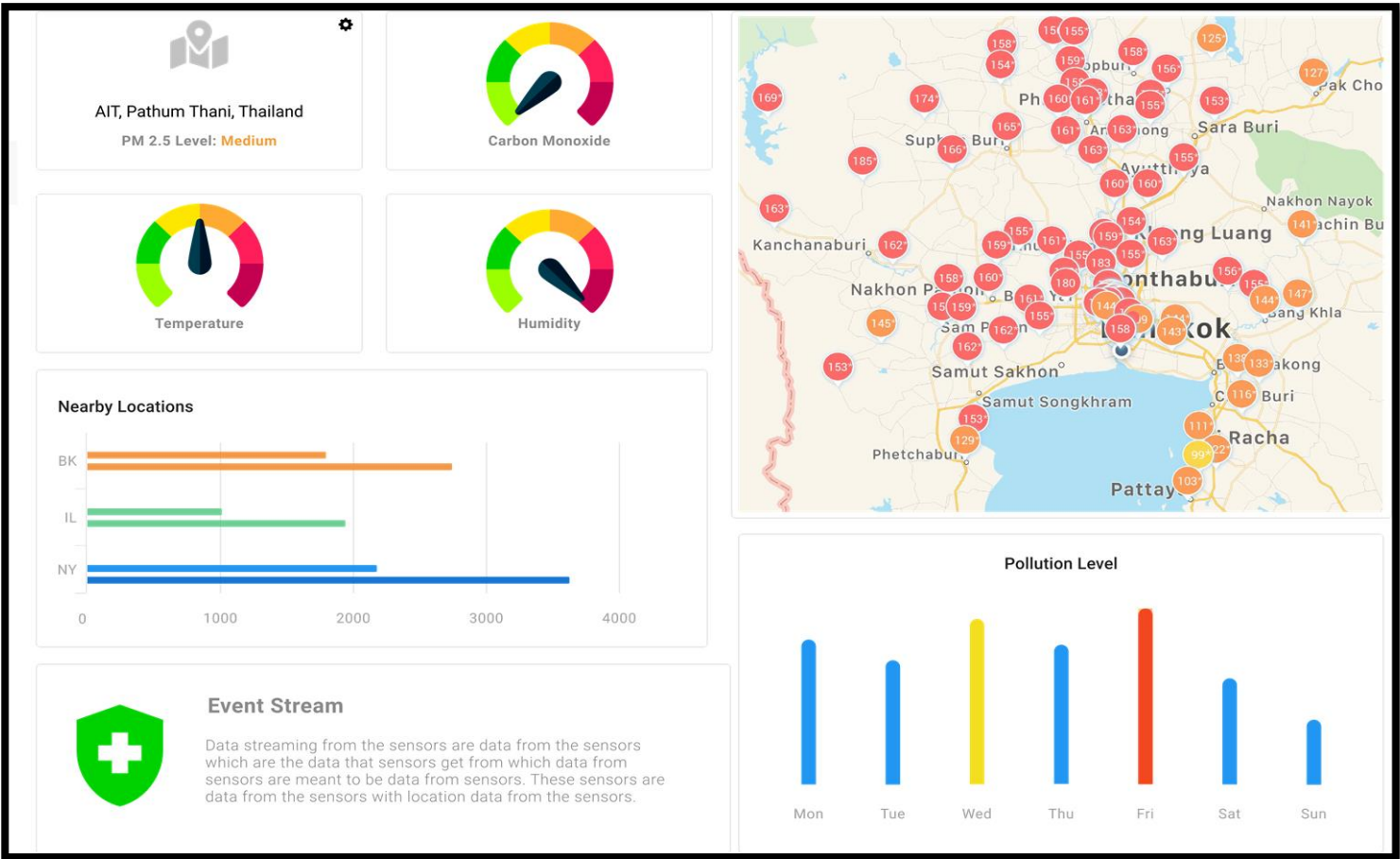



Figure 8. Dashboard for visitor

5.2 Interface for admin

Admin Login page: Once the admin enters the URL of the admin webpage, login-form will pop-up. Once the correct credentials (username and password) are given, admin will be directed to the admin page. This page contains various menu options in the sidebar.


 Air Quality
Monitoring System

Username

Enter your username

Password

Enter your password

LOGIN

Figure 9. Login form

5.3 Admin Dashboard

Once logged in, admin will be directed directly to the dashboard. Admin will be able to access the dashboard that contains the visualization same as visitors as explained above.

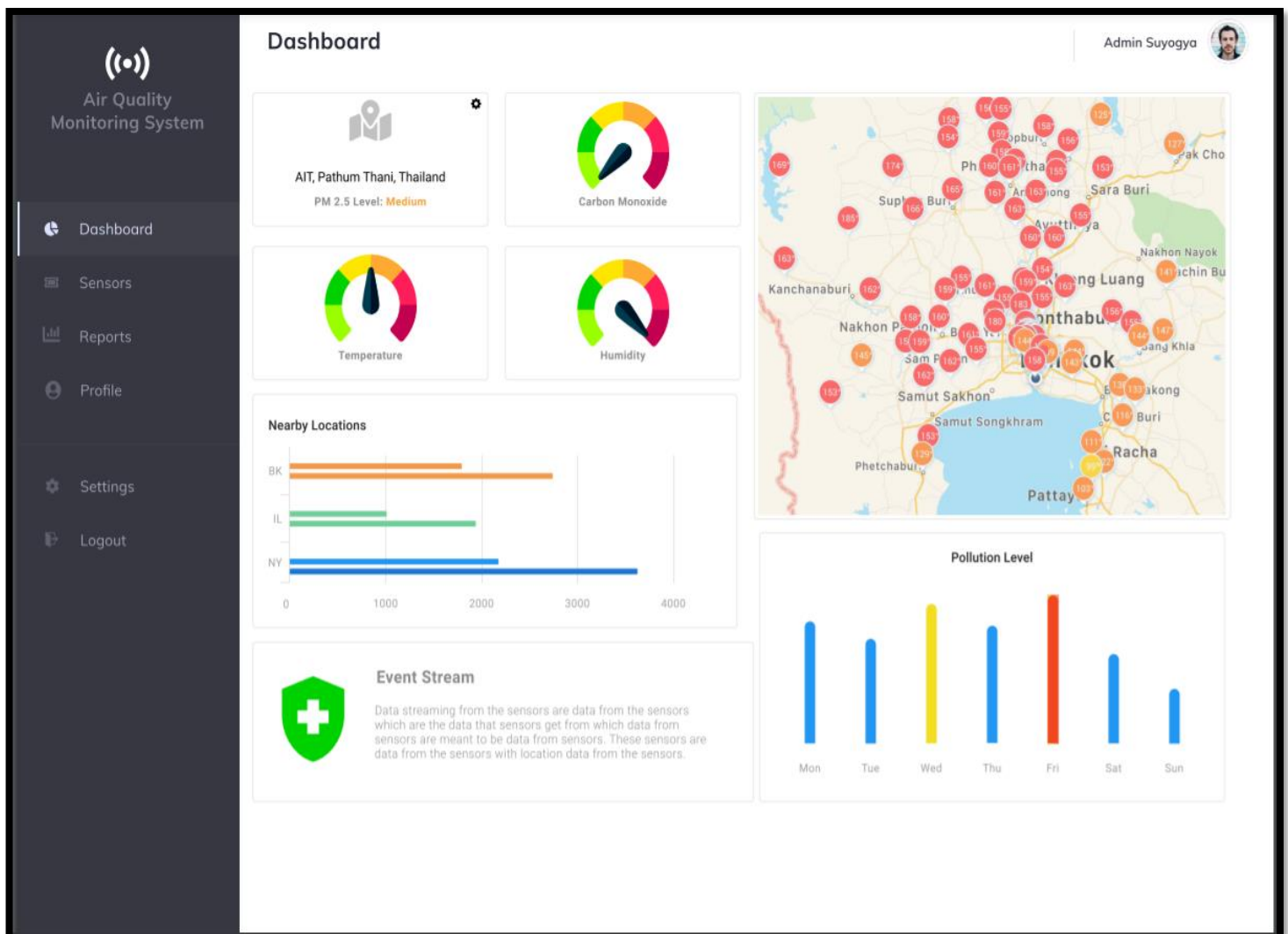


Figure 10. Dashboard for admin

5.4 Sensors Management

This menu contains the registration of new sensors as well as edit, disable and delete options for the registered sensors. It also displays the details about the registered sensors such as: Name, Model, API, type, and Region.

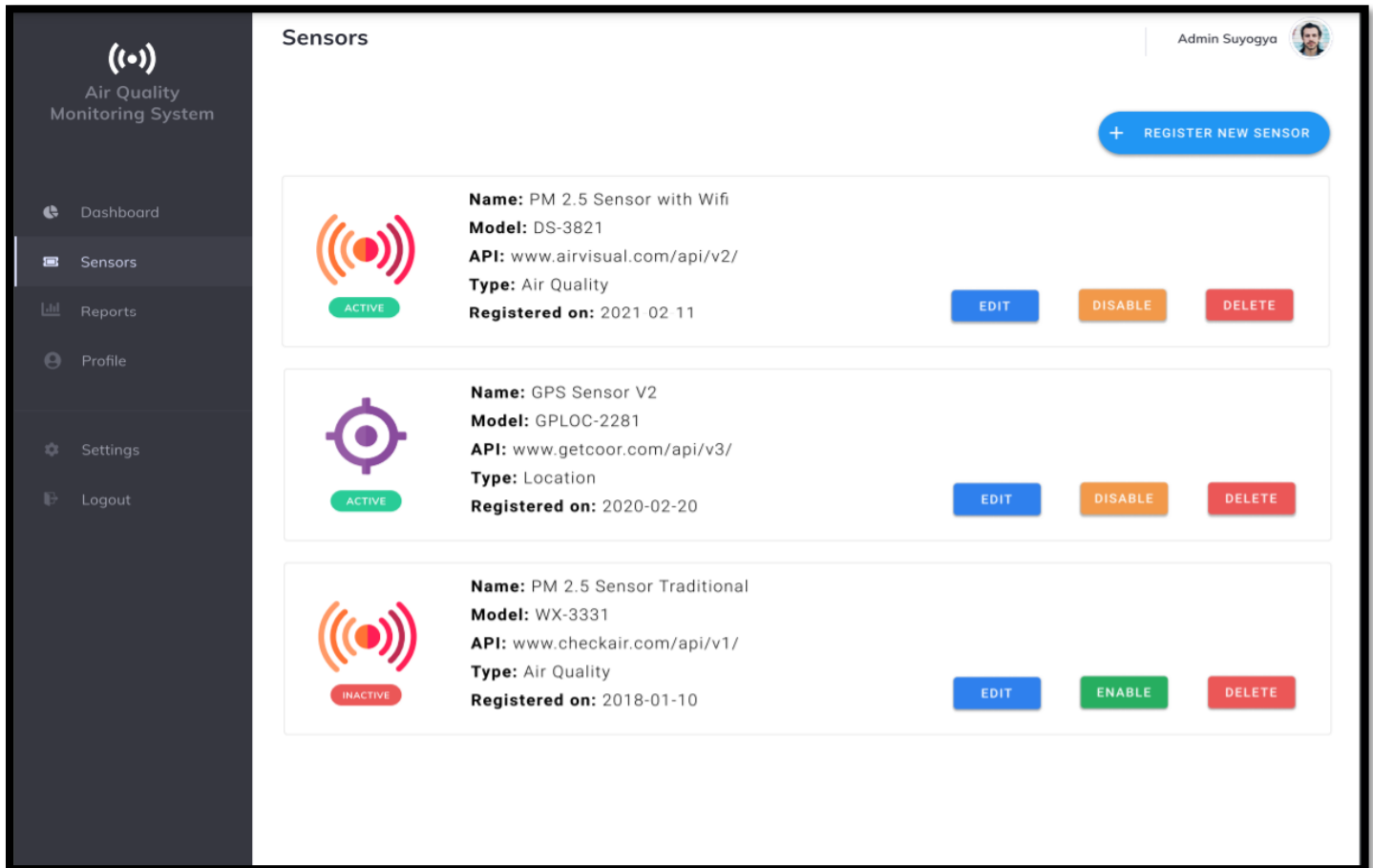


Figure 11. Sensor Management

5.5 Reports

Admin can generate reports in charts, tables, and other visualization forms, as per the requirement, using this Reports menu. It contains forms fields like Title of Report, Descriptive Content, Export File Format, select sensors option, Date, and generate submit button.

Air Quality Monitoring System

Dashboard
Sensors
Reports
Profile
Settings
Logout

Reports

Admin Suyogya

Title of Report
Text

Descriptive Content
Enter your report text here

Export File Format
PDF

Select sensors
☒ ID 1: PM 2.5 Sensor
☒ ID 2: GPS Sensor

From
Select Date

To
Select Date

+ GENERATE REPORT

Figure 12. Generate report.

5.6 Logout

Admin can logout from the admin page with this menu. Once logged out, admin will be redirected to the login page.

5.7 Update Profile

This menu contains the information of the admin. Fields like Name, Email and Change Password options are available. Admin can update these fields, if needed.

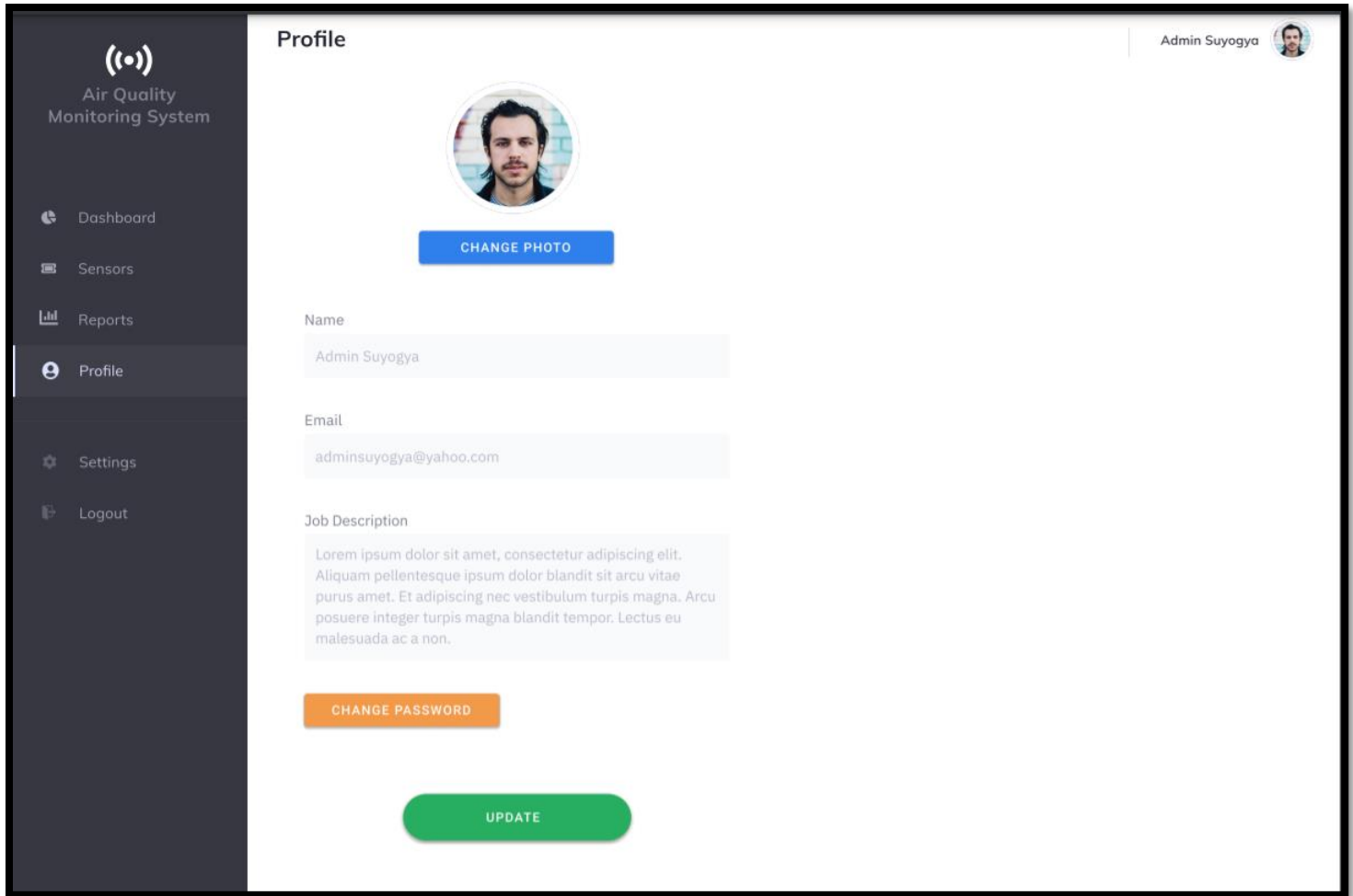


Figure 13. Update admin profile

5.8 Parameter Settings

Admin is privileged to update parameters of the system. As per the requirement admin can modify dashboard tiles, dashboard layout, manage data and data source with available menu like Enable/Disable Dashboard Tiles, Modify Dashboard Layout, Manage Data Sources and Data Backup/Restore.

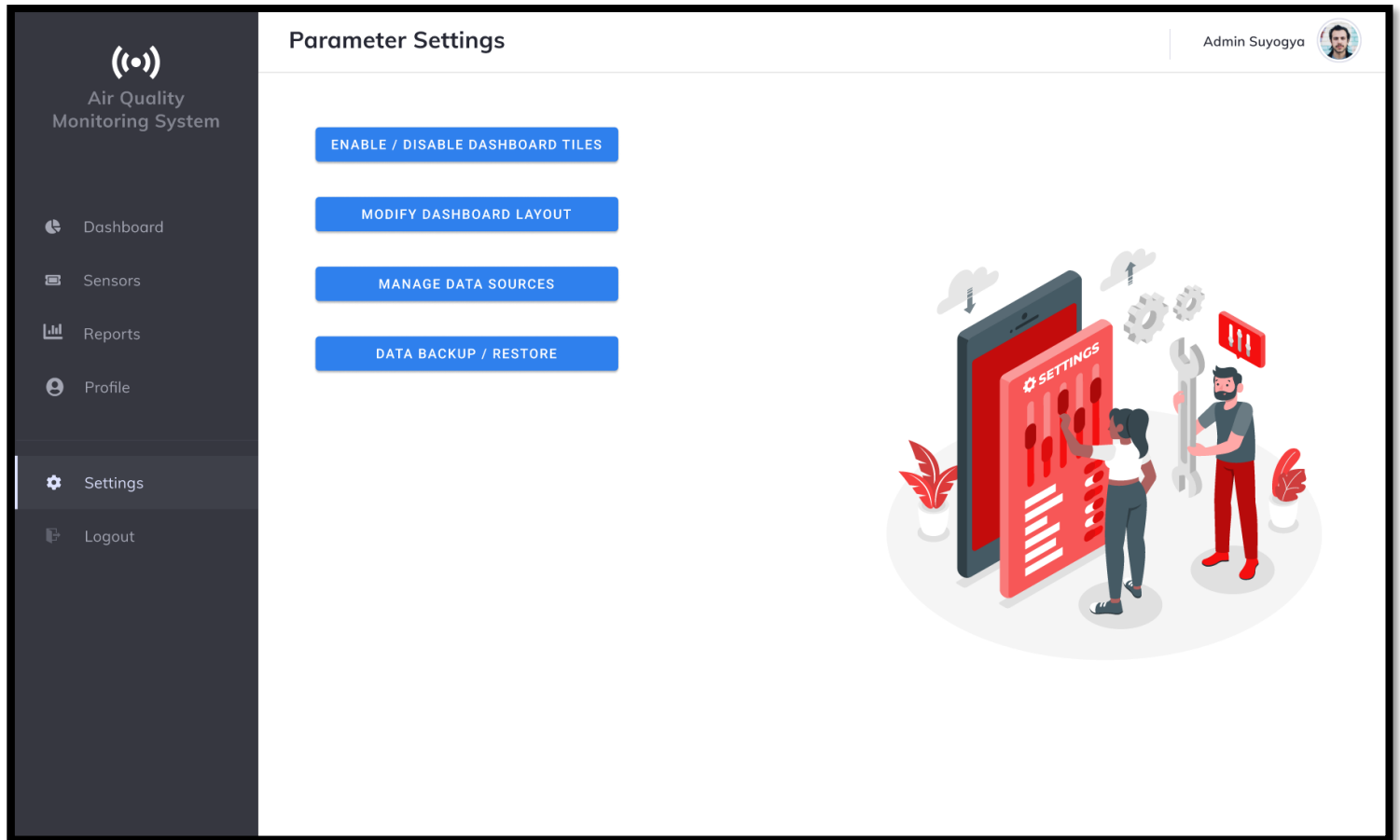


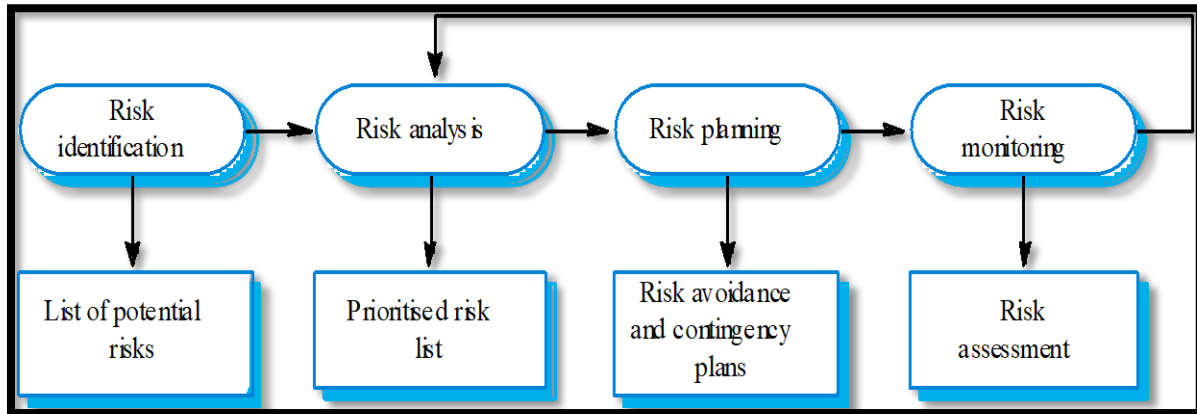
Figure 14. Update parameters

6. Risk and mitigation plan

Risk management is the identification assessment, and prioritization of risks followed by coordinated and economical application of resources to minimize, monitor, and control the probability and/or impact of unfortunate events or to maximize the realization of opportunities.

The main objective of Risk Management is to minimize, monitor and control the probability of infortune events that may occur while doing the project. It is also to assure that uncertainty does not deviate the endeavor from goals thus maximizing the realization of opportunity.

Figure 15. Risk Management process



A. Different risks that can occur during the progress of the project are:

Table 1. Risk Identification

<i>Risk</i>	<i>Description</i>	<i>Example</i>
Technology	The risk that can come from uncertainty in technologies that we used during our project.	System crash
People	During the progress of the project, the potential problem that might occur due to the group members' contribution.	Members are ill during critical times in the project.
Tools	The risks that occur when the CASE tools which support the project do not perform as expected.	Code is inefficient.
Requirement	While doing the project, if any changes occur in our mind to either add or reduce some functionality on our project then it will lead to requirement risk.	Deletion of one of the functionalities.
Estimation	The risk that are likely to occur due to time of completion and size of the project.	Underestimate time and size

B. RISK ANALYSIS

Table 2. Risk Analysis

Risk	Probability	Effect
System crashes	High	Serious
No internet access	Low	Tolerable
Members not able to come on time due to sickness or other reasons	Moderate	Serious
CASE tool which supports the project do not perform as expected	Moderate	Tolerable

Changes to requirement which require major work design are proposed	Moderate	Serious
Time required to develop the software is underestimated.	High	Serious
The size of the software is underestimated.	Moderate	Tolerable

C. Risk Planning

Consider each risk and develop a strategy to manage that risk by:

1. Avoidance strategies

Taking actions that will avoid the risk altogether.

2. Minimisation strategies

Try to either reduce the impact of the risk or reduce the probability of risk arising.

3. Contingency plans

If the risk arises, contingency plans are plans to deal with that risk.

Table 3. Risk Planning

Risk	Strategies
System crashes (avoidance)	Constant systems scan and backup the files.
Members unavailable due to some reasons (minimize)	By mitigating the work in between and making an extra effort to meet the deadlines.
CASE tool which supports the project do not perform as expected (avoidance)	We could look for plugins to work with the current CASE tools and if it does not work then, we can use other CASE tools to complete our project on time.
Changes to requirements which require major work. (contingency)	Maximizing information hiding in the design by reviewing the report.
Time required to develop the software is underestimated (contingency)	Create a short-term plan to complete the spill over tasks and give our extra effort
The size of the software is underestimated. (minimization)	Produce simple and effective products so that we can briefly describe the functionality.

D. Risk Monitoring

Risk Management is an iterative process:

- Assess each identified risk regularly to decide whether or not it is becoming less or more probable.
- Also assess whether the effects of the risk have changed.
- Each key risk should be discussed at group meetings.

7. Planned Schedule

Link to below Jira software:

<https://sqdi2021g1.atlassian.net/jira/software/c/projects/AQMS/boards/4/roadmap>

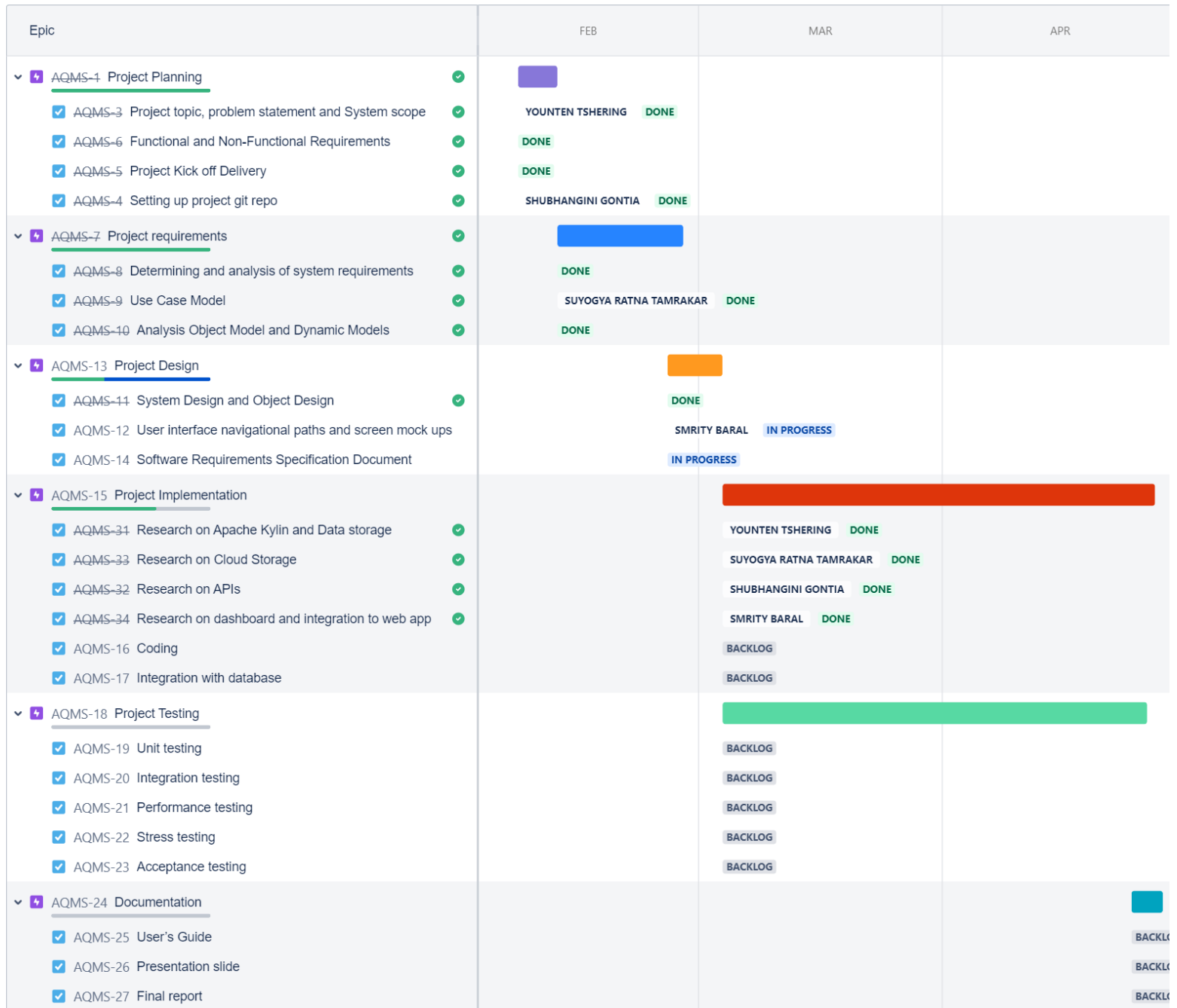


Figure 16. Work Plan

For more details on implementation follow our link:

https://github.com/shubhanginigon/SDQI2021_G1

8. Details on software requirement - Apache Kylin

Apache Kylin is an open-source distributed analytics engine designed to provide a SQL interface and multi-dimensional analysis on Hadoop supporting extremely large datasets. It was originally developed by eBay and is now a project of the Apache.

In addition, it easily integrates with BI tools via ODBC driver, JDBC driver, and REST API. It was created by eBay in 2014, graduated to Top Level Project of Apache Software Foundation just one year later, in 2015 and won the Best Open-Source Big Data Tool in 2015 as well as in 2016.

Currently, it is being used by thousands of companies worldwide as their critical analytics application for Big Data. While other OLAP engines struggle with the data volume, Kylin enables query responses in the milliseconds. It provides sub-second level query latency over datasets scaling to petabytes. It gets its amazing speed by precomputing the various dimensional combinations and the measure aggregates via Hive queries and populating HBase with the results.

Apache Kylin™ lets you query billions of rows at sub-second latency in 3 steps.

1. Identify a Star/Snowflake Schema on Hadoop.
2. Build Cube from the identified tables.
3. Query using ANSI-SQL and get results in sub-second, via ODBC, JDBC or RESTful API.

9. Glossary & references

9.1 Glossary

Here are some domain terms we are using in document:

- **Data Warehouse:** a data warehouse (DW or DWH), also known as an enterprise data warehouse (EDW), is a system used for reporting and data analysis.
- **Business Intelligence:** Business intelligence (BI) is the set of techniques and tools for the transformation of raw data into meaningful and useful information for business analysis purposes.
- **OLAP:** OLAP is an acronym for online analytical processing
- **OLAP Cube:** an OLAP cube is an array of data understood in terms of its 0 or more dimensions.
- **Lookup Table:** a lookup table is an array that replaces runtime computation with a simpler array indexing operation.
- **Dimension:** A dimension is a structure that categorizes facts and measures in order to enable users to answer business questions. Commonly used dimensions are people, products, place and time.

9.2 References

- Apache Kylin. (2015). Bring OLAP back to big data! Retrieved from Apache Kylin | Analytical Data Warehouse for Big Data
- Fann,N.,& Risley,D. (2011,January 5). The public health context for PM2.5 and ozone air quality trends. Air Qual Atmos Health 6, 1–11 (2013). <https://doi.org/10.1007/s11869-010-0125-0>
- Gupta,A.k., & Johari,R. (2019). IOT based electrical device surveillance and control system. International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU), <https://doi.org/10.1109/IoT-SIU.2019.8777342>