Capgemini

IOT DATA LOGGER

High Level Design & Low Level Design

The purpose of this document is to provide with a template for documenting both HLD & LLD.

Document Control :

Project Revision History

Date	Version	Author	Brief Description of Changes	Approver Signature
07-Dec-2022	0.1	Ezaz Sarkar	Start	

1. Introduction

The IoT data logger is used to measure and store environmental data such as temperature, humidity, illuminance, CO content, PM2.5, and PM10 contents. The climate IoT data logger also calculates the dew point, absolute humidity and the water vapor content in the air.
Data loggers are electronic devices which automatically monitor and record environmental parameters over time, allowing conditions to be measured, documented, analyzed and validated. Data Loggers are used in geotechnical monitoring and instrumentation on a large scale because they automatically collect real-time data from the sensors and display it over the devices. A data logger is an electronic device that records data (temperature, pressure, impact, humidity, etc.) at specific intervals. Discover the right data logger for you by exploring the different types and their intended applications. Data loggers vary from general-purpose types for a range of measurement applications to very specific devices for measuring in one environment or application type only. It is common for general purpose types to be programmable. One of the primary benefits of using data loggers is the ability to automatically collect data on a 24-hour basis. Upon activation, data loggers are typically deployed and left unattended to measure and record information for the duration of the monitoring period. This allows for a comprehensive, accurate picture of the environmental conditions being monitored, such as air temperature and relative humidity. Here we use the CSV file for storing the information and can also be updates.
Much has been said about the benefit of using data loggers to collect data over long periods of time for quality control, maintenance, transportation, etc., and the benefits of using such IOT data loggers are beyond doubt. Some of them can be integrated into systems, others can be installed in the area to be monitored or in the vehicle and after the monitoring time has expired, the data on the device can be called up and of course viewed on the computer - IOT data loggers, on the other hand, can be queried remotely at any time and installation is usually very easy, as it is wireless and fast.
The examples of IoT data loggers can be found in every area of life, not necessarily just in business, industry or agriculture. The term "smart", which has always been applied to people, is increasingly being applied to things, especially the sensors that measure each of the desired parameters, monitor the state and can later "report" on them to the user, regardless of whether they are information about temperature and humidity, soil quality, moisture content, number of people, noise level and many, many other parameters.
Here we use CSV files for collecting different data of a particular city which consist of information about temperature, humidity, PM2.5, PM10, NO2. This information is for each day over a year and also it shows the averages of all the

parameters of every month for a particular city. The overall strike of the project is it will also show the graphical representation report of a parameter change in a city over a year. It will also show the pollution level for a particular month and date. Here we process the data through the CSV file and the data logger which work as a sensor for receiving the information through the csv file.

1.1. Intended Audience

*[This section of the **HLD LLD Document** should provide the Names and Roles of the resources the **HLD LLD Document** is intended for and what is to be expected out of this document.]*

1.2. Acronyms/Abbreviations

IOT	Internet of Things
PM	Particulate Matter
NO2	Nitrogen Dioxide
CSV	Comma-Separated Values

1.3. Project Purpose

- **The purpose of this software or project is to show the data files from different cities based on date.**
- **Data logging means to record events, observations, or measurements systematically. It consists of the following data like temperature, rainfall, humidity, PM2.5, PM10, N02 recordings date wise for a period of a year.**
- **This project will also show graphical report of a parameter changes in a city over a year.**
- **The user can also find the pollution level or temperature for a particular month. The Data Logger module allows you to save data in form of CSV file.**
- **You can create and configure file according to your needs and monitor them while they are being logged.**

1.3.1. In Scope

Scope: This project aims to monitor environmental behavior using data loggers. Data loggers can record and transmit precise temperature, humidity, and pressure data so that an organization can keep a close eye on the environment of a given area. Data logger are implicitly stand-alone devices, this stand-alone aspect of data loggers implies onboard memory that is used to store acquired data. Sometimes this memory is very large to accommodate many days, or even months, of unattended

recording. Some equipment log data 24 hours a day and 365 days of year while some logs data only during its predefined period. Once it is setup, it performs data logging automatically and does not require presence of human beings. It is very accurate as likelihood of human error is not there. In future, other climatic parameters such as wind, light intensity etc. can be included for analysis and prediction purpose and this system can therefore be used for analysis and prediction of such parameters in remote areas.

1.3.2. Out of scope

LIMITATIONS:

- Disparate Data and Data Storage Limitations
- If data logging equipment malfunctions, some data could be lost or will not be recorded.
- Certain data logging equipment can take readings only during interval configured at the start.
- Basic training is needed to use the equipment

1.4. Functional Overview

*[This section of the **HLD LLD Document** Provides a general description of the software system including its functionality and matters related to the overall system function and its design Feel free to split this discussion up into subsections (and sub sections, etc ...).]*

1.5. Assumptions, Dependencies & Constraints

*[This section of the **HLD LLD Document** is for describing any assumptions, dependencies or constrains that are taken into consideration while preparing this document. These may concern such issues as:*

Related software or hardware

Operating systems

End-user characteristics

Possible and/or probable changes in functionality]

1.6. Risks

*[This section of the **HLD LLD Document** is for describing the risks that are taken into consideration while preparing this document. All the risks related to Software, Hardware, Operating System, Users, etc have to be documented here.]*

2. Design Overview

In this section, a general description of the software system including its functionality and matters related to the overall system and its design has to be documented.]

2.1. Design Objectives

Instant getting information about the environment details of a particular city through the csv file. It is used to measure and store environment data such as temperature, humidity etc.

There are primary and secondary objectives behind this project.

Primary:

The user will get the averages of different parameters regarding environment of a particular city including the date and the city name. This average information consists of the averages of every month in a year. It is used to measure and store environment data.

The parameters include the following: -

- CITY
- DATE
- TEMPERATURE
- HUMIDITY
- PM2.5
- PM10
- NO2

Secondary:

The user will search any month and get the average of all parameter which they want and also can search about any particular date in a year and will get the information regarding all parameters for that particular date.

2.1.1. Recommended Architecture

[In this section, a document the Recommended System Architecture]

2.2. Architectural Strategies

[Describe any design decisions and/or strategies that affect the overall organization of the system and its higher-level structures. These strategies should provide insight into the key abstractions and mechanisms used in the system architecture. Describe the reasoning employed for each decision and/or strategy (possibly referring to previously stated design goals and principles) and how any design goals or priorities were balanced or traded-off.]

2.2.1. Design Alternative

[All the available alternatives have to be documented here along with the reasons for selection or rejection of the particular alternative.]

2.2.2. Reuse of Existing Common Services/Utilities

[Document the details of all the available common services or Utilities that will be used by this system here.]

2.2.3. Creation of New Common Services/Utilities

[Document the details of all the new services or Utilities that have to be created as part of this system here.]

2.2.4. User Interface Paradigms

[Document the User Interface Paradigms here.]

2.2.5. System Interface Paradigms

[Document the System Interface Paradigms here.]

2.2.6. Error Detection / Exceptional Handling

[A good system design ensures Error Detection and Exception handling procedures. Document all the details on how the Error detection has to be done in the system and how the Exceptions are thrown and handled in the system in this section.]

2.2.7. Memory Management

[Memory Management is a critical aspect of any system. A system designed keeping Memory Management in view uses very less Memory and frees up unused memory at frequent intervals. Document all the Memory Management policies, Critical issues related to Memory Management like Relocation, Protection, Sharing, Logical and Physical Organization etc. to be implemented in the system here. Focus on Design Decisions to Manage Memory.]

2.2.8. Performance

The system will work on the client's terminal . The performance will depend on the hardware component of the user's system.

2.2.9. Security

[Security has emerged as the most important aspect of any system. A system designed with good security principles ensures Integrity of the system and prevents from attacks and data leakage. Document all the Security Requirements and features implemented in the system including the use and management of integrity and access controls that apply to the system and its components. Also include any tools that will support security and privacy requirements.]

2.2.10. Concurrency and Synchronization

[If the system needs to be in synch with another system, the details of the same have to be documented here.]

2.2.11. Maintenance

Very little maintenance is required for this setup. Only the logging takes space. It is reliable and low cost and time efficient monitoring solution for

any measuring opportunity. It is of high accuracy, easy to use and greater versatility in every application.

3. System Architecture

[This section should provide a high-level overview of how the functionality and responsibilities of the system were partitioned and then assigned to subsystems or components. The main purpose here is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together to provide the desired functionality.

At the top-most level, describe the major responsibilities that the software must undertake and the various roles that the system (or portions of the system) must play. Describe how the system was broken down into its components/subsystems (identifying each top-level component/subsystem and the roles/responsibilities assigned to it). Describe how the higher-level components collaborate with each other in order to achieve the required results.]

3.1. System Architecture Diagram. (Not Necessary)

[If there are any diagrams, models, flowcharts, documented scenarios or use-cases of the system behavior and/or structure, they may be included here.]

3.2. System Use-Cases

[If there are any documented scenarios or use-cases of the system behavior and/or structure, they may be included here]

3.3. Subsystem Architecture

[If a particular component is one which merits a more detailed discussion than what was presented in the System Architecture section, provide that more detailed discussion in a subsection of the System Architecture section. If necessary, describe how the component was further divided into subcomponents, and the relationships and interactions between the subcomponents.

If any subcomponents are also deemed to merit further discussion, then describe them in a separate subsection of this section. Proceed to go into as many levels/subsections of discussion as needed in order for the reader to gain a high-level understanding of the entire system or subsystem (but remember to leave the gory details for the Detailed System Design section).]

3.4. System Interfaces

[A good design ensures that all the System's Interfaces are well documented. List out the details of all the System Interfaces, interface Design along with diagrammatic representation if possible with details of flow, frequency etc.]

3.4.1. Internal Interfaces

[Document all the details of Internal Interfaces the system interacts with along with the details of data flow and frequency.]

3.4.2. External Interfaces

[Document all the details of External Interfaces the system interacts with along with the details of data flow and frequency.]

4. Detailed System Design

[Most components described in the System Architecture section will require a more detailed discussion. Other lower-level components and subcomponents may need to be described as well. Each subsection of this section will refer to or contain a detailed description of a system software component. The discussion provided should cover the following software component attributes in complete detail.]

4.1. Key Entities

[Provide a Comprehensive list of the Key Entities associated with the System in this section.]

4.2. Detailed-Level Database Design

[The detailed database design information can be included here. Describe in detail the design of the database; all database related files associated with the system, and any non-DBMS files pertinent to the database design. Include discussions about or references to the following:

• Logical Data Model (LDM) and LDM Entity Relationship Diagram (ERD).

• Physical Data Model (PDM) and PDM ERD.

• A comprehensive Data Dictionary showing data stores, data element name, type, length, source, constraints, validation rules, maintenance (create, read, update, delete (CRUD) capability), audit and data masking requirements, expected data volumes, life expectancy of the data, information life-cycle management strategy or at least an archiving strategy, outputs, aliases, and description.

• Indexes that will be required for the data objects.

• Planned implementation factors (e.g., distribution and synchronization) that impact the design.]

4.2.1. Data Mapping Information

[The detailed data mapping information has to be documented here. Describe in detail the requirements of data mapping, Data Models to be mapped, Integration details etc. including

Data transformation or data mediation between a data source and a destination

Identification of data relationships as part of data lineage analysis

Discovery of hidden and sensitive data, such as data masking.

Consolidation of multiple databases into a single database and identifying redundant columns of data for consolidation or elimination.]

4.2.2. Data Conversion

[The detailed data conversion information has to be documented here. Describe in detail the requirements of data conversion, formats of conversion, resource requirements, files associated etc.]

4.3. Archival and retention requirements

[Describe in detail the Archival and retention requirements of the system including the schedule and frequency of archival and retention and the strategies involved.]

4.4. Disaster and Failure Recovery

[Describe in detail the disaster and recovery procedures of the system in case of untoward incidents including the scope of disaster recovery procedures, requirement of resources, data restoration paths etc.]

4.5. Business Process workflow

[Document the Business Process Workflow in this section here.]

4.6. Business Process Modeling and Management (as applicable)

[Document the Business Process Modeling and management details in this section]

4.7. Business Logic

[Document the complete Business Logic this section including the code.]

4.8. Variables

[Document the details of Variables, naming conventions, usage etc in this section.]

4.9. Activity / Class Diagrams (as applicable)

[Document the details related to Activity / Class Diagrams in this section.]

4.10. Data Migration

[The Data Migration section should provide details of Data Migration involved in the section below. Further sections or subsections can be added depending up on the requirements of the project.]

4.10.1. Architectural Representation

*[This section describes what software architecture is for the current system, and how it is represented. Of the **Use-Case**, **Logical**, **Process**, **Deployment**, and **Implementation Views**, it enumerates the views that are necessary, and for each view, explains what types of model elements it contains.]*

4.10.2. Architectural Goals and Constraints

[This section describes the software requirements and objectives that have some significant impact on the architecture: use of an off-the-shelf product, portability, distribution, and reuse. It also captures the special constraints that may apply: design and implementation strategy, development tools, team structure, schedule, legacy code, and so on.]

4.10.3. Logical View

[This section describes the architecturally significant parts of the design model, such as its decomposition into subsystems and packages and for each significant package, its decomposition into classes and class utilities. You should introduce architecturally significant classes and describe their responsibilities, as well as a few very important

relationships, operations, and attributes.]

4.10.4. Architecturally Significant Design Packages

[For each significant package, include a subsection with its name, its brief description, and a diagram with all significant classes and packages contained within the package.

For each significant class in the package, include its name, brief description, and, optionally a description of some of its major responsibilities, operations and attributes.]

4.10.5. Data model

- **Legacy system data model**
- **Proposed system data model**
- **Interface data model**

4.10.6. Deployment View

*[This section describes one or more physical network (hardware) configurations on which the software is deployed and run. At a minimum for each configuration it should indicate the physical nodes (computers, CPUs) that execute the software, and their interconnections (bus, LAN, point-to-point, and so on.) Also include a mapping of the processes of the **Process View** onto the physical nodes.]*

5. Environment Description

[The complete details of the System Environment has to be documented in this section including the details of all requirements, time zones etc.]

5.1. Time Zone Support

- IST Kolkata (GMT + 5:30)

5.2. Language Support

- English

5.3. User Desktop Requirements

- 64-bit processor, 1 GHz or faster
- At least 2 GB free hard drive space
- At least 1 GB RAM

5.3.1. Application Server Disk Space

- No such disk space is required as the program is fully functional on online IDE(s) as well.
- The Local Operating System is required and one text file to store the records of processes.

5.3.2. Database Server Disk Space

- No such disk space is required as the program is fully functional on online IDE(s) as well.

- The Local Operating System is required and one text file to store the records of processes.

5.3.3. Integration Requirements

- Language: C
- Tools: Ctags, Valgrind, Make
- Complier: gcc
- Operating System: Linux Environment Ubuntu 22.04

5.3.4. Jobs

[Details with respect to addition, modification, deletion of Jobs for this system have to be documented here.]

5.3.5. Network

[Network requirement details have to be documented here]

5.3.6. Others

[Any details which are specific to this system and are not covered in the sections above have to be documented here.]

5.4. Configuration

[Complete information with respect to the Configuration requirements has to be detailed out here in this section and sub sections.]

5.4.1. Operating System

- Linux Environment Ubuntu 22.04

5.4.2. Database

[Describe the Database configuration requirements here.]

5.4.3. Network

[Describe the Network configuration requirements here. Details of all the Network Components etc.]

5.4.4. Desktop

[Describe the desktop configuration requirements here. Details of Application software required and other configurations.]

6. References

*[This section should provide a complete list of all documents referenced elsewhere in the **HLD LLD Document**. Each document should be identified by title, report number (if applicable), date, and publishing organization. Specify the sources from which the references can be obtained.] This information may be provided by reference to an appendix or to another document.]*

7. Appendix

[This section should provide a complete list of all documents or links on the Internet where related material can be found.]

Change Log

QMS Template Version Control (Maintained by QA)			
Date	**Version**	**Author**	**Description**
28-May-2015	1.0	QA Team	Initial Version