Cloud Web App for DC Motor Control

Group No.: B07-SCD-BITS-PS1-2021

A REPORT

ON

"CLOUD WEB APP FOR DC MOTOR CONTROL"

BY

Nirzari Kalpesh Shah **2019A8PS0576G**

At

Shalaka Connected Devices LLP – Embedded Systems/IoT, Pune

A Practice School-I Station of

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

(July, 2021)

A REPORT

ON

"CLOUD WEB APP FOR DC MOTOR CONTROL"

BY

Nirzari Kalpesh Shah **2019A8PS0576G**

Prepared in partial fulfillment of the
Practice School-I Course Nos.
BITS C221/BITS C231/BITS C241

At

Shalaka Connected Devices LLP – Embedded Systems/IoT, Pune

A Practice School-I Station of

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

(July, 2021)

Acknowledgements …

We express our sincere thanks to Prof. Sharda Tripathi and Instructor Hemant Kamat who gave us the opportunity to make this project and for their invaluable contribution in the making of the project. We would like to thank the PS Division of BITS Pilani University for taking such a brilliant and necessary initiative. This will forever contribute to our work ethic and learning experience for our career.

We are immensely obliged to our batch mates for their elevating inspiration, encouraging guidance and kind supervision in the making of our project. They inspired us to do justice to this opportunity awarded to us.

In preparation of our project, we were helped and guided by our peers and mentors, among the others who utilized their time and energy and we would like to expand our gratitude to all those who have guided us in this project.

.

Abstract Sheet ...

BIRLA INSTITUTE OF TECHNOLOGY AND SCENCE PILANI

(RAJASTHAN)

Practice School Division

Station: Shalaka Connected Devices LLP **Centre:** Pune

Duration: 2 months **Date of Start:** 31st May, 2021

Date of Submission: 22nd July, 2021

Title of the Project: Cloud Web App for DC Motor Control

Student ID/ Name/ Discipline:

2019A8PS0576G - Nirzari Kalpesh Shah -Electronics and Instrumentation

Expert's Name/ Designation: Dr. Hemant Kamat - CTO (Chief Technology Officer)

PS Faculty's Name: Prof. Sharda Tripathi - Professor, Department of Electrical and Electronics Engineering at Bits Pilani

Key Words: HTML, CSS, Javascript, Node.js, Angular, MySQL (for Database management), Mosquitto, Python, Edge Computer, MQTT Broker, Local GUI, Web GUI, Cloud Computer

Project Areas: Cloud Web Application development, API Development, Data Visualisation

Abstract: Develop Web App for controlling 3 DC Motors over MQTT and collect health status of the system. Plot graph for the same using selected data. Collaborate with SCD-BITS-PS1-2021-09 (group working on embedded project of DC Motor Control). We are Group No.: B07-SCD-BITS-PS1-2021.

Signature of Student *Signature of PS Faculty*

Date: 22nd June,2021 *Date: 22nd June, 2021*

Table of Contents …

- Acknowledgements……………………………………4
- Abstract…………………………………………………..5

1. **Introduction……………………………………………………………………………………………….8**
    a. **Description…..8**
    b. **Objectives…..8**
    c. **Platforms and Tools…..9**
    d. **Team members…..9**
    e. **What is Cloud Application?…..10**
    f. **Benefits of Cloud Application…..10**
    g. **Project Requirements…..12**
    h. **Project Background…..12**

2. **Progress Report………………………………………………………………………………………13**
    a. **Learning Progress……………………………………………………………………………13**
        i. **Introduction to Shalaka Connected Devices…..13**
        ii. **What is Industry 4.0?…..14**
        iii. **What is IIoT?…..15**
        iv. **Product Development Lifecycle and it's stages…..17**
        v. **Documentation…..22**
            1. **It's Importance and advantages**
            2. **Two types: Featured and Design Documentation**
        vi. **Best UI/UX Practices…..25**
        vii. **Group Discussions…..28**

    b. **Work Progress……………………………………………………………………………………30**
        i. **PHASE 1: DOCUMENTATION…..30**
            1. **Feature Documentation**

2. Design Documentation
3. Concept
4. Architecture
5. Revision History
6. Features
7. Implementation
8. Python Team (Device Simulator)
9. Javascript Team (Layout Designing (UI))
10. Project Timeline (we followed up for the project)

ii. PHASE 2: UI/UX DESIGNING…..42
1. Step 1: Information Architecture – User Experience (UX) Design
2. Step 2: Visual Designing – User Interface (UI) Design
3. Step 3: Prototype of website

iii. PHASE 3: WEBSITE DEVELOPMENT…..45
1. Version 1.0 of website
2. Version 2.0 of website
3. Version 3.0 and 4.0 of website
4. Version 5.0 of website
5. Final Version 6.0 of website
6. Process followed up in setting up the website
7. Final version website screens

3. Conclusion……………………………………………………………………………………………….52
4. Appendices……………………………………………………………………………………………….53
a. Appendix 1: Details of Software Packages used in the project…..53
b. Appendix 2: Information on MQTT…..55
c. Appendix 3: Code breakup of Final Version 6.0 of website explained…..59
d. Appendix 4: About ReMoNet…..77

5. References……………………………………………………………………………………………..78
6. Glossary……………………………………………………………………………………………….79

Introduction ...

DESCRIPTION:

The Project is of developing web application for controlling 3 DC Motors over MQTT and collect health status of the system. Further plot graph for the same using selected data. For this, we need to collaborate with SCD-BITS-PS1-2021- 09, group working on Embedded part project of DC Motor Control. The data we will require to work on, for the project, will be send over by this group through MQTT broker. We have implemented the project mainly in 3 phases:

1. Documentation 2. UI/UX Designing 3. Website Development

OBJECTIVES:

- ✓ Develop Web App to Acquire data from Controller for 3 DC motors using MQTT
    - Voltage
    - Current
    - Temperature
    - Vibration
    - Speed and Direction

- ✓ Save the data in database
- ✓ Display the same in tabular form of Main Page of the Web Application
- ✓ Plot graph for the same using filtered data
- ✓ Set and Monitor Alerts
- ✓ Send Control Commands to the Controller using MQTT
- ✓ Collaborate with SCD-BITS-PS1-2021-09

PLATFORMS AND TOOLS:

- Edge Computer
    - Python

  MQTT Broker
    - Mosquitto

  Web Application
    - Javascript
    - Node.js
    - Angular
    - MySQL

  UI/UX Design Tool
    - Figma

  Code Language
    - HTML
    - CSS

TEAM MEMBERS:

- Nirzari Kalpesh Shah
- Lakshya Pratap Singh Shekhawat
- Priyanshu Gupta
- Anisha Anilkumar
- Tushar Khandelwal

What is Cloud Application?

A cloud application, or cloud app, is a software program where cloud-based and local components work together. This model relies on remote servers for processing logic that is accessed through a web browser with a continual internet connection.

Cloud application servers typically are located in a remote data center operated by a third-party cloud services infrastructure provider. Cloud-based application tasks may encompass email, file storage and sharing, order entry, inventory management, word processing, customer relationship management (CRM), data collection, or financial accounting features.

Benefits of cloud apps:

- **Fast response to business needs.** Cloud applications can be updated, tested and deployed quickly, providing enterprises with fast time to market and agility. This speed can lead to culture shifts in business operations.

- **Simplified operation.** Infrastructure management can be outsourced to third-party cloud providers.

- **Instant scalability.** As demand rises or falls, available capacity can be adjusted.

- **API use.** Third-party data sources and storage services can be accessed with an application programming interface (API). Cloud applications can be kept smaller by using APIs to hand data to applications or API-based back-end services for processing or analytics computations, with the results handed back to the cloud application. Vetted APIs impose passive consistency that can speed development and yield predictable results.

- **Gradual adoption.** Refactoring legacy, on-premises applications to a cloud architecture in steps, allows components to be implemented on a gradual basis.

- **Reduced costs.** The size and scale of data centers run by major cloud infrastructure and service providers, along with competition among providers, has led to lower prices. Cloud-based applications can be less expensive to operate and maintain than equivalents on-premises installation.

- **Improved data sharing and security.** Data stored on cloud services is instantly available to authorized users. Due to their massive scale, cloud providers can hire world-class security experts and implement infrastructure security measures that typically only large enterprises can obtain. Centralized data managed by IT operations personnel is more easily backed up on a regular schedule and restored should disaster recovery become necessary.

<u>Benefits of Cloud Applications</u>

Project Requirements:

Cloud Front End:

- Functional Requirement
    - Access data from MQTT Broker
    - Save data to database
    - Display data in various formats
    - Analyze data
    - Display results of analysis
- Platform
    - Node.js framework
- Programming Language
    - Javascript
- Tools
    - Eclipse IDE
    - Node.js Modules
    - Other Javascript Libraries

Project Background:

The DC Motor Control Website is part of ReMoNet website. Initial two pages of ReMoNet* are:

1. First page, i.e., welcome page: anyone with the access to server can land on the page (No Login required)
2. Second page: contains micro-sites with different team's login page

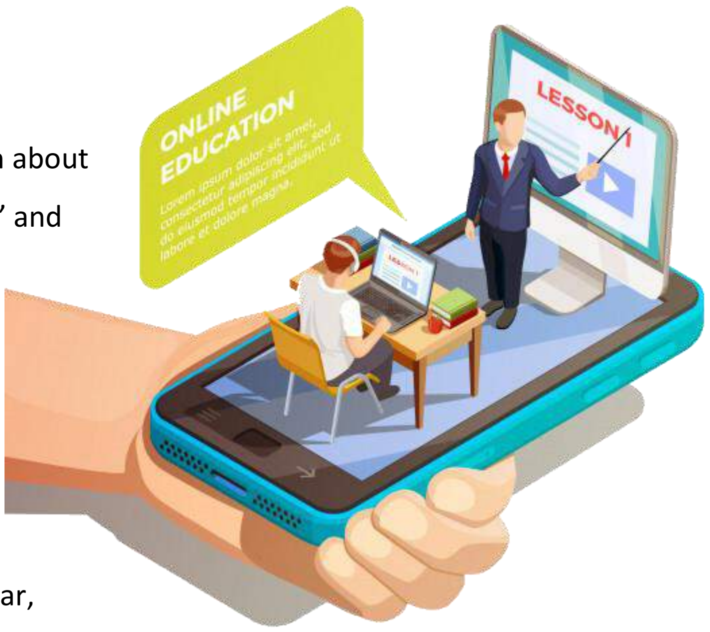
3rd page onwards, we had to build the website for DC Motor Control and monitoring it's Health Status. We have completed this project in 3 main phases:

1. Documentation
2. UI/UX Design
3. Website Development

*More about ReMoNet in Appendix 4

Progress Report …

LEARNING PROGRESS:


ONLINE EDUCATION
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna.
LESSON 1

During initial 6 Week's period, we got to learn about the company 'Shalaka Connected Devices LLP' and about various industry standards and practices: Industry 4.0, IIoT, Product Development Lifecycle, Feature and Design Documents, best UI/UX practices. We were given small introduction to the tools:

HTML, CSS, Javascript, MySQL, Node.js, Angular, Mosquitto, Python

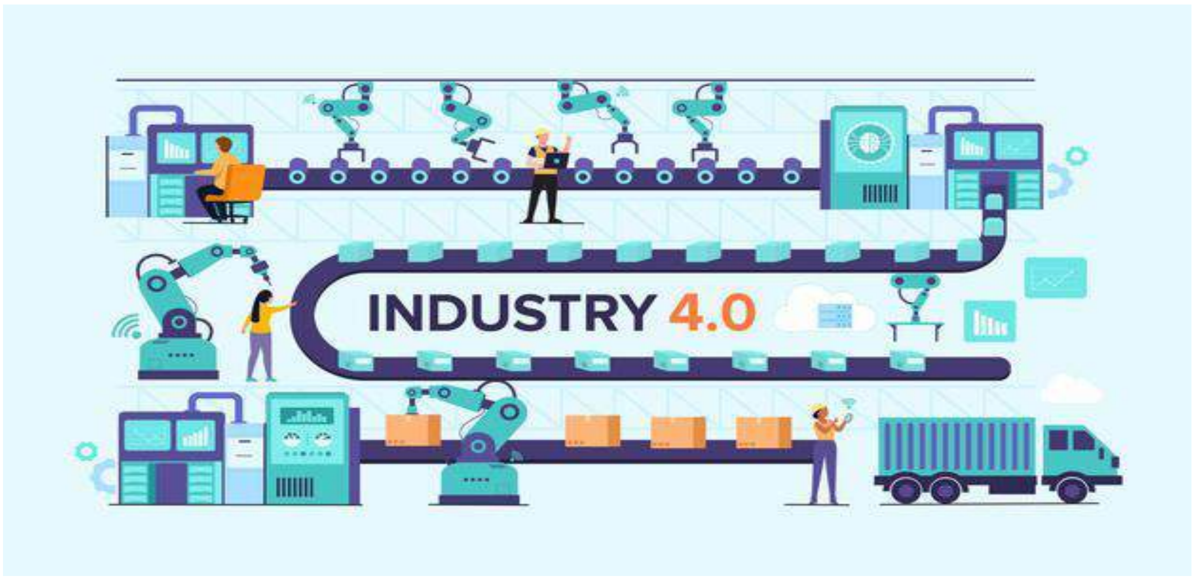
The Induction Session, hosted by expert Hemant Kamat (CTO), started with the introduction to the Shalaka Connected Devices LLP. This introduction included the description of the team, it's experience, it's skills and expertise.

- *Team Introduction:* We were introduced to electronics and communication engineer Mr. Hrishikesh Kamat, who is the Chief Executive Officer with 5 years of experience & Mr. Hemant Kamat, who is the Chief Technology Officer with 34 years of experience.
- *Experience:* in various job like 'Proving client concepts', 'Prototyping client products', 'Creating engineering samples of client products', 'Volume manufacturing of client products', 'Integrating embedded systems with IoT' and 'Designing to various International Compliances'.
- *Skills:* in various fields like 'Shortlist key components', 'Assemble PCB' and:

  # Inhouse: Design circuit schematic, Design PCB, Design & Develop Firmware, Test integrated product, Design Enclosure, Design & Develop Andriod & iOS Applications

  #Outsource: Manufacture PCB, Manufacture/ procure off the shelf enclosures

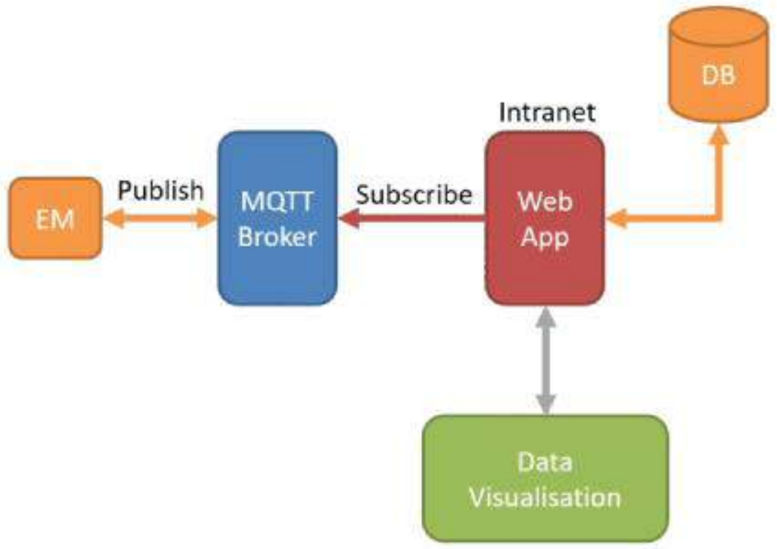
- *Expertise:* in Microcontrollers, System on Chip & Wireless Technologies



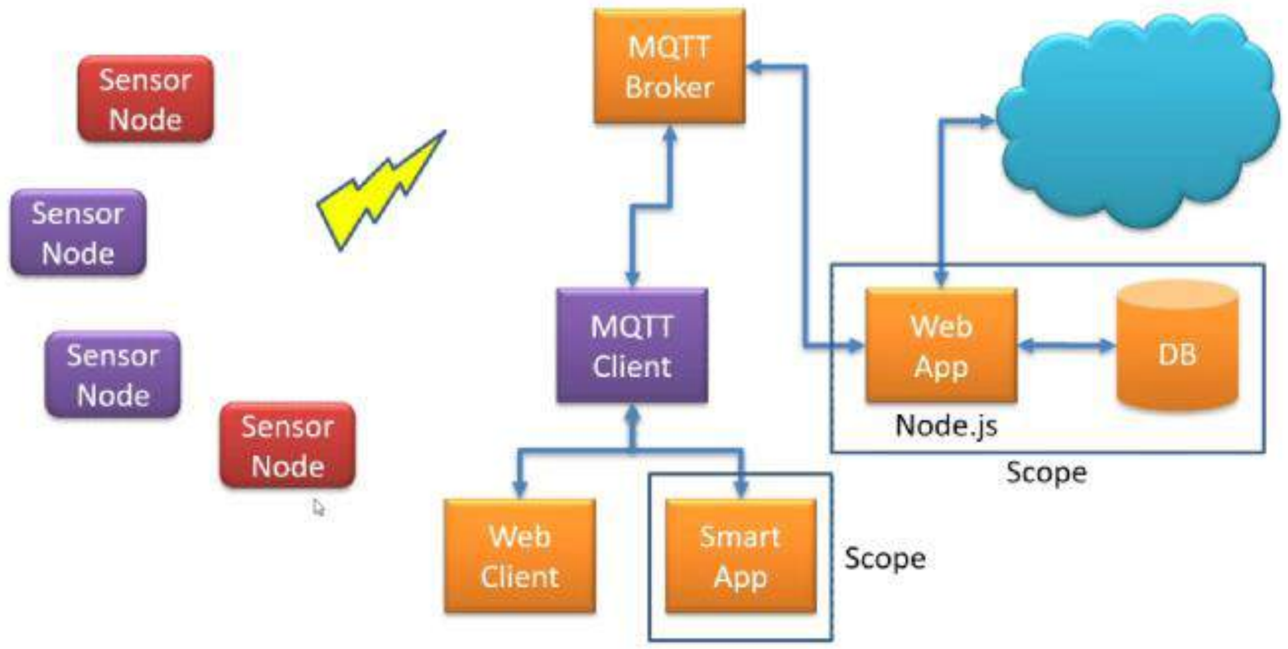
Following this we learnt about ‘Industry Architecture’. “What is Industry 4.0?” is the question that was addressed in the session. Industry 1.0 - Steam Engine; Industry 2.0 – Discovery of Electricity; Industry 3.0 - Computer Application; Industry 4.0 - current trend of automation and data exchange in manufacturing technologies includes cyber-physical systems, the Internet of things and cloud computing. Industry 4.0 creates what has been called a “smart factory”.

Then Data Flow was discussed after it and it’s parts: MQTT Broker, Web Application, Data Visualization, Data Base. Next the IoT protocols were discussed. Two of them MQTT and COAP were introduced.

DATA FLOW



Industrial Internet of Things



IIoT stands for the Industrial Internet of Things or Industrial IoT that initially mainly referred to an industrial framework whereby a large number of devices or machines are connected and synchronized through the use of software tools and third platform technologies in a machine-to-machine and Internet of Things context, later an Industry 4.0 or Industrial Internet context.

Today IIoT is mainly used in the scope of Internet of Things applications outside of the consumer space and enterprise IoT market, as an umbrella term for applications and use cases across several industrial sectors.

The Industrial Internet of Things or IIoT is defined as “machines, computers and people enabling intelligent industrial operations using advanced data analytics for transformational business outcomes”.

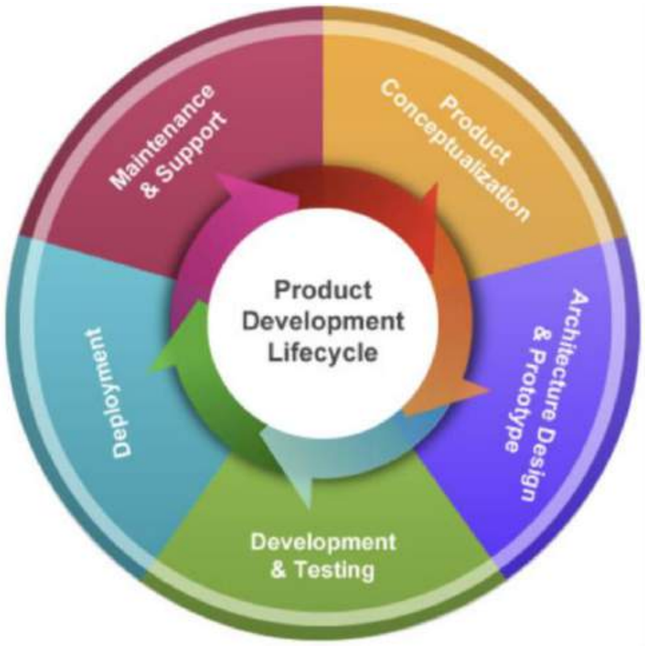
Some of the key benefits of IIoT in an industry context

- ✓ Improved and intelligent connectivity between devices or machines
- ✓ Increased efficiency
- ✓ Cost savings and
- ✓ Time savings
- ✓ Enhanced industrial safety

Architecture:

IIoT systems are usually conceived as a layered modular architecture of digital technology.The device layer refers to the physical components: CPS, sensors or machines. The network layer consists of physical network buses, cloud computing and communication protocols that aggregate and transport the data to the service layer, which consists of applications that manipulate and combine data into information that can be displayed on the driver dashboard. The top-most stratum of the stack is the content layer or the user interface.

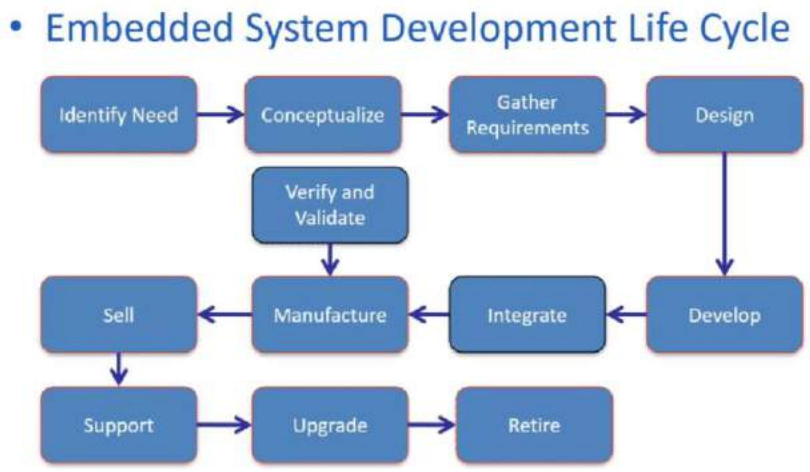
Product Development Lifecycle


Product Development Lifecycle
Product Conceptualization
Architecture Design & Prototype
Development & Testing
Deployment
Maintenance & Support

Further target topic was Product Development Lifecycle – Design & Development Lifecycle. It started by introducing us to the ‘Embedded System Development Life Cycle’ with the help of block diagram as follows:

Stages in Lifecycle of a Product:

1. Need Identification
2. Conceptualization
3. Requirement Gathering
4. Design and Development
5. Implementation
6. Manufacturing
7. Marketing
8. Support
9. Upgrade
10. Retirement



What deliverables would be required at every stage and who all participants would be the part of the system are conveyed through the following points:

1. **Need Identification:**
    - Market Research
    - Identify new requirement
    - Identify lacunae in existing products
    - Create Wish list
    - Requires investment
    - No "profits"
    - Deliverables: Market Survey Report
    - Participants: Consumers/ End Users, Market Survey Professionals, Marketing Team, "Others"

2. **Conceptualization:**
    - Analysis of Market Survey Report
    - Brainstorming
    - First level formalization of Wish list
    - Pragmatic analysis of Wish list
    - Pooling of innovative ideas
    - Requires investment
    - No "profits"
    - Deliverables: Concept Document
    - Participants: Top Management, Research and Development Team, Marketing Team

3. **Requirement Gathering:**
    - Specifications finalization from Concept Document
    - Scope of the product
    - Technology required to implement the product
    - Technology required to manufacture the product
    - Technology required to support the product
    - Technical and Business feasibility of the product
    - Deliverables: Product Requirement Document, Functional Specifications Document, Manufacturing Process Document
    - Participants: Top Management, Research and Development Team, marketing Team, Product Management

4. **Design and Development:**
    - Input/ Output Specification of the Product
        - System Specification
    - Product Architecture
    - Rapid Prototyping
        - Simulation
        - Analysis
            a. Functional
            b. Structural
            c. FMEA
    - Prototyping
    - Usability and Reliability Testing
        - Laboratory Testing
        - Field Testing
    - Deliverables: Architecture, High and Low Level Design, Simulation and Analysis Reports, Prototypes
    - Participants: Top Management, Research and Development Team, Marketing Team, Product Management, Systems Engineering, Manufacturing and Process Engineering, Quality Assurance

5. **Implementation:**
    - Identify Bill of Material
    - Identify Manufacturing Process
    - Identify Quality Assurance Testing
    - Identify Capital Equipment
        - Manufacturing
        - Testing
    - Build Human Resources Team
    - Vendor Development
    - Material Procurement
    - Deliverables: Bill of Material, Manufacturing Process Document, Quality Testing Document, Manufacturing Maintenance Manual, Product User Document, Product Servicing Document
    - Participants: Research and Development Team, Marketing Team, Product Management, Systems Engineering, Manufacturing and Process Engineering, Quality Assurance

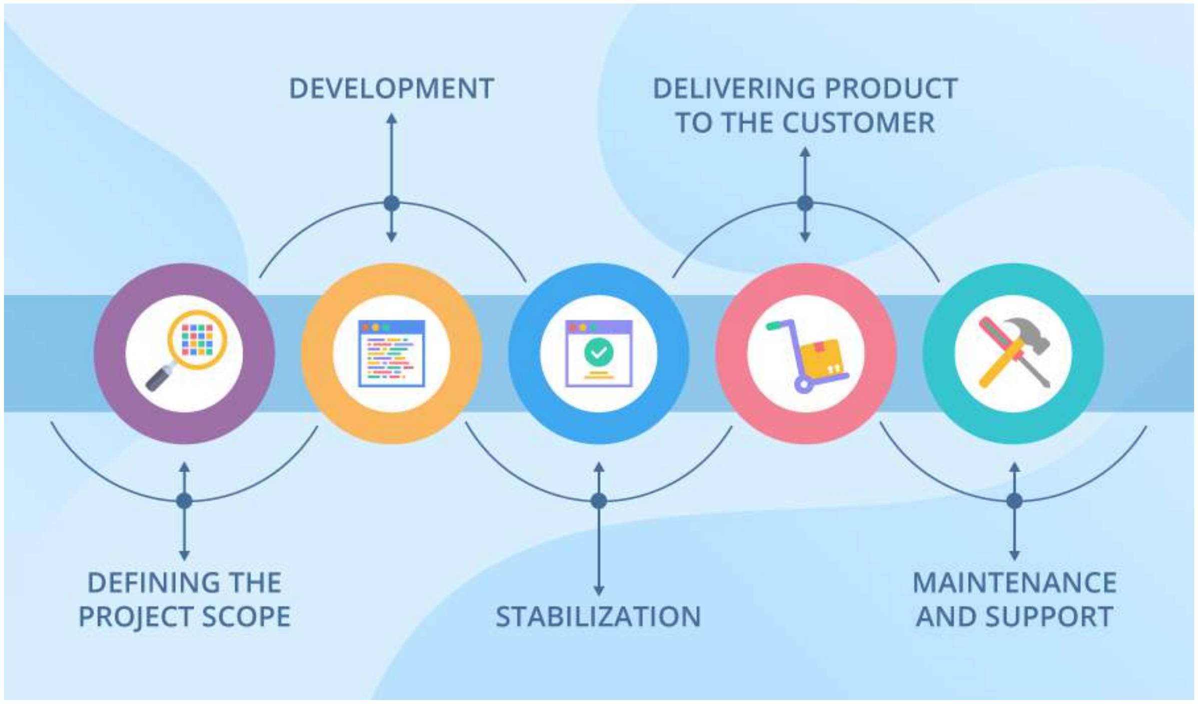
6. **Manufacturing:**
    - Manufacture Product
    - Test Product
        - Functional
        - Reliability
            - Accelerated tests
        - Usability
    - Packaging of Product
    - Dispatch to warehouse for “store and forward”
    - Deliverables: Product, Quality Test Reports, Compliance Reports
    - Participants: Research and Development Team, Marketing Team, Product Management, Systems Engineering, Manufacturing and Process Engineering, Quality Assurance

7. **Marketing:**
    - Run advertisement campaigns
    - Run promotional offers
    - Client visits
    - Technical Marketing
    - Deliverables: Marketing collaterals, Sales and Marketing Reports, End User experience feedback
    - Participants: Research and Development Team, Marketing Team, Product Management, Systems Engineering, Manufacturing and Process Engineering, Quality Assurance

8. **Support and Maintenance:**
    - Address and resolve end user queries
    - Address and resolve end user quality complaints
    - Address and resolve field breakdown of product
        - In Warranty
        - Out of Warranty
        - Maintenance Contracts
    - Deliverables: End User experience feedback, End User quality complaints
    - Participants: Research and Development Team, Marketing Team, Product Management, Systems Engineering, Manufacturing and Process Engineering, Quality Assurance, Support Team

9. **Upgrade (Optional):**
    - Upgrade the product by
        - Retro fitting new features
        - Replacing old features with new ones
    - Postpone Obsolescence
    - Deliverables: New Features
    - Participants: Research and Development Team, Marketing Team, Product Management, Systems Engineering, Manufacturing and Process Engineering, Quality Assurance, Support Team

10. **Retirement:**
    - Declare End of Life Schedule
    - Product replacement campaign
    - After market product support through third party contracts
    - Deliverables: End of Life Schedule
    - Participants: Top Management, Marketing Team, Product Management, Systems Engineering, Manufacturing and Process Engineering, Support Team


DEVELOPMENT
DELIVERING PRODUCT TO THE CUSTOMER
DEFINING THE PROJECT SCOPE
STABILIZATION
MAINTENANCE AND SUPPORT

We learned on how to work on the initial phase of any project in Product Life Cycle, i.e. **"Documentation"**. There are ***two types of Documentation in the Product Life Cycle***:

1. Feature Documentation;
2. Design Documentation

We further learned elaborate details of each type of documentation:

1. **Feature Document:** A feature set can best be summarized as a written document that lists the specifications of a product. It includes the list of features that together makes a product. On top of that, you cover your design vision [UI (User Interface)] that will be used to build the product.

2. **Design Document:** Design documentation is a collection of documents and resources that covers all aspects of your product design. Documentation should include information about users, product features, and project deadlines; all essential implementation details; and design decisions that your team and stakeholders have agreed on.

Importance and various advantages of documentation in the product life cycle.

For a programmer **reliable documentation is always a must**. The presence of documentation helps keep track of all aspects of an application and it improves on the quality of a software product. Its main focuses are **development, maintenance and knowledge transfer (KT)** to other developers. Successful documentation will make information easily accessible, provide a limited number of user entry points, help new users learn quickly, simplify the product and help cut support costs. Documentation is usually focused on the following components that make up an application: **server environments, business rules, databases/files, troubleshooting, application installation and code deployment.**

1. Server Environments:

Detailed documentation about an application and its environments is always a must. This information will help with setting up new environments for your application and it should present the location and function of the systems that run your services. Things that should be specified here are the **application name/version, server name, IP, code directory, URL to the application, operating system, user account information and a point of contact**.

2. Business Rules:

Business rules documentation **help new additions to the team adapt faster to the working habits of the company**. It provides information on how the product works and why. Business rules documentation can easily be supported with requirements documents if available. This will speed up a developer's learning curve significantly. In addition to business rules, a **help document**, **FAQs**, or **user guide** can help highlight the main points of an application for a developer who needs context for the application they are supporting.

3. Database/Files:

Database information is mandatory for **porting, reverting, sharding, migrating** and so on. It is important to know the **type of database, the server information, the version but most importantly to have a data model diagram**. Documentation of the database will make bringing additions to the table, modifications to the structure and types, additions of indexes and keys much more simpler and easier to control/debug. Also, if an application presents a file transfer functionality, it is recommended to document which way the transfer is made, through which protocol and datatypes, if and what SSL certificates are needed.

4. **<u>Troubleshooting</u>:**

The troubleshooting documentation helps when **running into production issues**. Most technical issues should have error codes that should help with troubleshooting. In this document there should also be included an FAQ section to deal with general or usual problems (such as configuration issues). The errors should be documented split by type of error, module where it comes from, and level of error (exception, warning, critical, etc...).

5. **<u>Application Installation</u>:**

Installation and configuration documents are useful for when developers need to **set up new or additional application environments**. If possible, the steps should be detailed and easy to follow and can include screenshots if necessary. Anyone should be able to follow the steps and successfully install an application. Having the steps identified will help the installer prevent problems because of missing parts of an application. Details such as necessary software, libraries, and application server versions, can be included to ensure the environment will be compatible and set up as intended.

6. **<u>Code</u>:**

The code documentation is the backbone of every application. Code documentation can be split in multiple parts. The first one, the most helpful for programmers are the **comment blocks**. These will be found through every file explaining classes, methods, parameters, possible errors. Then comes the **specific file documentations**. These are usually generated through a third party script which will parse a file and, based on the comment blocks, will create an explicit PDF. Afterwards there should be information regarding the **code repository**, where the file updates are found, and where they need to be moved. In addition, there should be step-by-step instructions on how to create an application package or a build to be deployed.

Some of the best practices of UI/UX that were discussed are as follows:

- **KEEP IT SIMPLE**

People spend less than 15 seconds on a website. Keeping them focused on your brand's message requires a design that is clear and simple. This can be achieved by following the "less is more" concept and avoiding a busy, cluttered design.

Users are more likely to find the key messaging on your website if they have less content to scroll through and fewer options to choose from. Minimizing the options on your site will increase the likelihood that readers will take the action your brand seeks, such as leading them through a sales conversion funnel. Imagine the best grocery store experience where everything along the way is exactly where you need it!

A simpler design also loads faster and reduces the bounce rate from your site.

Bonus points: this also boosts your site's search engine optimization (SEO)!

An excellent example of simple design is Virgin's website, which gives you enough information to keep you interested without overloading you. Despite the fact that Virgin is a large conglomerate, the site presents the brand in a simple, clean way. Of particular interest is how the navigation is laser-focused down to four primary menu items in the header.

A perfect example of "less is more."

- **USE CONSISTENCY**

Readers will find it easier to scroll through your site if you use consistency in your design. This applies to your colors, fonts, buttons, layout structure, photograph style, and more. The point is to make these design components familiar to your users, which helps make things more intuitive and predictable. And people do like predictability!

"Predictability in UX can be defined as how much a user can successfully foresee the result of an interaction."

Beyond the internal consistency, website visitors also like to make connections with design conventions used across the web. The location of navigation buttons, the option to "read more" on content, and the use of autofill on a web form are all common design features across the digital landscape.

All of this consistency takes out the guesswork users may have when trying to interact with your website.

Airbnb does a fantastic job achieving consistency with the visual elements on its site.

The brand relies heavily on "cards" to create this uniformity: big cards, little ones, some in-betweens, simple cards, and more complicated ones. These cards can be sorted when you're looking for experiences or places to stay, but ultimately you can expect a few consistent elements—a stunning photo, a title to capture your attention, and a variety of details deemed important at that phase of your journey.

BE INTENTIONAL

Choose the elements of your design with a purpose in mind. The colors, visual elements, and layout you select should all be intentional and not just because they look good.

Looks aren't everything! Design is more about how something works than how it looks. It's about both form and function.

Here are some guidelines for designing with intent:

- ✓ Choose a color because it's part of the brand identity guide or because it evokes the desired emotion.
- ✓ Choose a font because it shows personality and allows the user to more easily read your content.
- ✓ Choose a visual element that helps users accomplish a task on the site.
- ✓ Choose a layout that showcases the most important parts of your design and that improves site navigation.

When in doubt, challenge the intent even when it may be uncomfortable. You may need to question why you want things done a certain way. By doing this, you can uncover the rationale behind design decisions.

One example of intentionality can be found in a visit to Koval's website, a web production company. Upon arrival, the user is greeted with many distractions. While this goes against the 'less is more' philosophy that makes for a clean user experience, the company's site experience fits its motto: "We create fast sites from which no one will escape." True to form, as you scroll down the page and hover, things move, they change and you just might find yourself entranced.

Be honest. How long did the website capture your attention?

SET EXPECTATIONS…

Along the lines of using consistency to create predictability, users also want their expectations for browsing your website to be met. They want to understand what will happen if they take an action on your platform. Tell users what the buttons on your website are intended to do. If the button will sign them up for your newsletter, add appropriate text on the button—"Sign Up Now" or "Join Our Email List"—that tells them exactly what will happen if they submit their email address.

Another feature readers will appreciate is an animated loading graphic. These "loaders" will let them know that the page is not frozen if there is a wait time while something loads.

Meeting your users' expectations requires some design elements that may be obvious. For example, give commonly used names to the headers on the navigation menu. This will reduce any potential frustration readers may have while navigating and becoming familiar with a new platform.

A HOSPITABLE SITE IS ACCESSIBLE AND INCLUSIVE

How will users who are colorblind or have a hearing impairment experience your site? Creating a website that is disability-friendly might not be something you "see" right from the start—unless you happen to be someone who needs to navigate the platform using accommodations.

Try using a screen reader, assistive touch, or a colorblind web page filter and you'll have an entirely different perspective on whether your site is truly accessible. The first step in making your site available to all users is to determine whether your site's design creates obstacles for users with disabilities.

Here are a few tips for providing an accessible site:

- ✓ Choose black text on a white background instead of colored text.
- ✓ Use subtitles on videos posted on the site.
- ✓ Use descriptive alt tags on images so screen readers can describe them for the user.
- ✓ Create larger buttons for people with fine motor skill difficulties (or those of us prone to "fat-finger" syndrome).

You can test your site's accessibility at WebAccessibility.com to see where it ranks. After reviewing the results, make the recommended modifications to ensure your website can be used by anyone—no matter how they view it.

○ USE RESPONSIVE DESIGN

Mobile usage has surpassed desktop usage and each year its share of online traffic rises. That's why you need to use a responsive design on your site so that users can easily read and navigate it on any device.

A website design that isn't mobile responsive may, for example, force users to enlarge the text on the screen to make it legible. Most often, readers who have to adjust the site on their cell phone to work around your non-responsive design will simply give up and leave your site.

Providing a responsive website doesn't only enhance the user experience. In fact, Google awards higher rankings to sites that are mobile-friendly, which translates into higher traffic on your site. (There's that SEO, again!)

Responsive websites load more quickly, look great on every screen, and are easier to update.

It's critical to make a positive first impression on users coming to your site from mobile devices. If you don't, you might lose them to competitors' more mobile-responsive sites.

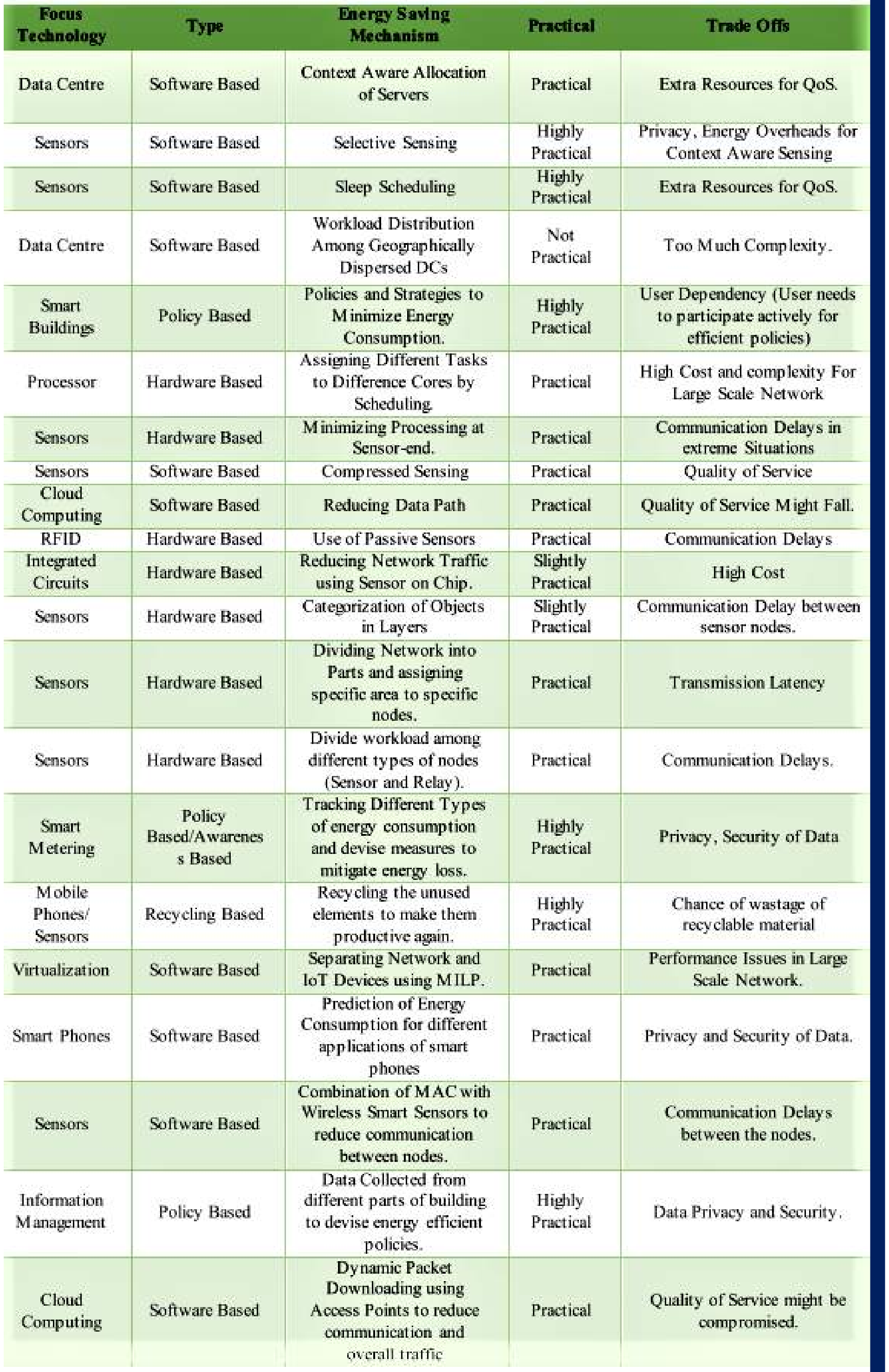
AltaFoodcraft, a company that provides office coffee and refreshment services, has a fully responsive website. If you look at the site on a cellphone, the home page is easy to read and navigate because it has been optimized with a responsive design.

We had **Group Discussion 1** on *Topic :*

“The impact of IoT technology on the environment: Need for green IoT”

For the Group Discussion we were to form groups of 4-5 among ourselves, and each group was expected to speak either in favour or against the topic. My group represented against the topic. We presented the tradeoffs faced when particular focused technology was applied with the particular type of Green IoT. It is tabulated as follows:

We all students were given enough time to present our views and hence the group discussion turned out to be healthy and insightful discussion.



Focus Technology	Type	Energy Saving Mechanism	Practical	Trade Offs
Data Centre	Software Based	Context Aware Allocation of Servers	Practical	Extra Resources for QoS.
Sensors	Software Based	Selective Sensing	Highly Practical	Privacy, Energy Overheads for Context Aware Sensing
Sensors	Software Based	Sleep Scheduling	Highly Practical	Extra Resources for QoS.
Data Centre	Software Based	Workload Distribution Among Geographically Dispersed DCs	Not Practical	Too Much Complexity.
Smart Buildings	Policy Based	Policies and Strategies to Minimize Energy Consumption.	Highly Practical	User Dependency (User needs to participate actively for efficient policies)
Processor	Hardware Based	Assigning Different Tasks to Difference Cores by Scheduling.	Practical	High Cost and complexity For Large Scale Network
Sensors	Hardware Based	Minimizing Processing at Sensor-end.	Practical	Communication Delays in extreme Situations
Sensors	Software Based	Compressed Sensing	Practical	Quality of Service
Cloud Computing	Software Based	Reducing Data Path	Practical	Quality of Service Might Fall.
RFID	Hardware Based	Use of Passive Sensors	Practical	Communication Delays
Integrated Circuits	Hardware Based	Reducing Network Traffic using Sensor on Chip.	Slightly Practical	High Cost
Sensors	Hardware Based	Categorization of Objects in Layers	Slightly Practical	Communication Delay between sensor nodes.
Sensors	Hardware Based	Dividing Network into Parts and assigning specific area to specific nodes.	Practical	Transmission Latency
Sensors	Hardware Based	Divide workload among different types of nodes (Sensor and Relay).	Practical	Communication Delays.
Smart Metering	Policy Based/Awarenes s Based	Tracking Different Types of energy consumption and devise measures to mitigate energy loss.	Highly Practical	Privacy, Security of Data
Mobile Phones/ Sensors	Recycling Based	Recycling the unused elements to make them productive again.	Highly Practical	Chance of wastage of recyclable material
Virtualization	Software Based	Separating Network and IoT Devices using MILP.	Practical	Performance Issues in Large Scale Network.
Smart Phones	Software Based	Prediction of Energy Consumption for different applications of smart phones	Practical	Privacy and Security of Data.
Sensors	Software Based	Combination of MAC with Wireless Smart Sensors to reduce communication between nodes.	Practical	Communication Delays between the nodes.
Information Management	Policy Based	Data Collected from different parts of building to devise energy efficient policies.	Highly Practical	Data Privacy and Security.
Cloud Computing	Software Based	Dynamic Packet Downloading using Access Points to reduce communication and overall traffic	Practical	Quality of Service might be compromised.

Group Discussion 2 on “How can IoT technology be leveraged for management of Covid 19 pandemic ? “

It was meant for us to gain knowledge on how IoT can benefit various sectors of professions during the pandemic and even after it. We all students were given enough time to present our views and hence the group discussion turned out to be healthy and insightful discussion.

The current applications of IoT during COVID-19:
Currently, IoT is already used to manage some aspects of the COVID-19. For example, drones are already used for public surveillance to ensure quarantine and the wearing of masks. AI has also been used to predict future outbreak areas.

Using IoT to dissect an outbreak:
With the numerous and diverse datasets collected by mobile devices, IoT can have many more applications during an epidemic.
IoT can be used to trace the origin of an outbreak. A recent study by researchers at MIT used aggregated mobile phone data to trace, in granular details of short distances and periods, the spread of dengue virus in Singapore during 2013 and 2014. Therefore, overlaying geographic information system (GIS) on IoT mobile data from infected patients can do two things. Upstream, it can assist epidemiologists in their search for patient zero; downstream, it can help identify all the persons who have come into contact with the infected patients and may, therefore, also be infected.

Using IoT to ensure compliance to quarantine:
IoT can also be used to ensure patient compliance once the potentially infected persons enter into quarantine. Public health personnel can monitor which patients remain quarantined, and which patients have breached the quarantine. The IoT data will also help them track down who else may be exposed due to the breach.

Using IoT to manage patient care:
The scalability of IoT also comes in handy for monitoring all the patients who are high-risk enough to warrant quarantine but not serious enough to warrant in-hospital care. Right now, the daily check-up of the patients is done manually by healthcare workers who go door-to-door. In one reported instance, a healthcare worker had patients standing in their apartment balconies, so that he could fly a drone up to take their temperatures with an infrared thermometer. With IoT, the patients can have their temperatures taken and upload the data with their mobile devices to the cloud for analysis. This way, healthcare workers can not only collect more data using less time but also reduce the chance for cross-infection with the patients.

WORK PROGRESS:

PHASE 1: DOCUMENTATION

Templates for the Documentation were provided to us in the Google Drive, and were explained in the session each and every part of the template, i.e. what to write for every part.

For the **Feature Documentation** part following were the subparts to be filled up:

1. Document title: Project name
2. Subtitle: Objective of the Project
3. Abstract: 4-6 lines describing design and testing
4. Contents: Index of the document
5. Revision history: Table containing every version's information like: Revision Number, Revision By (name), Revision Date, Revision Details (Additions & Modifications)
6. Disclaimer: Details like Copy-write, Legacy & Privacy, Contact & Address
7. Overview: It's an extended abstract. Contains requirements of clients

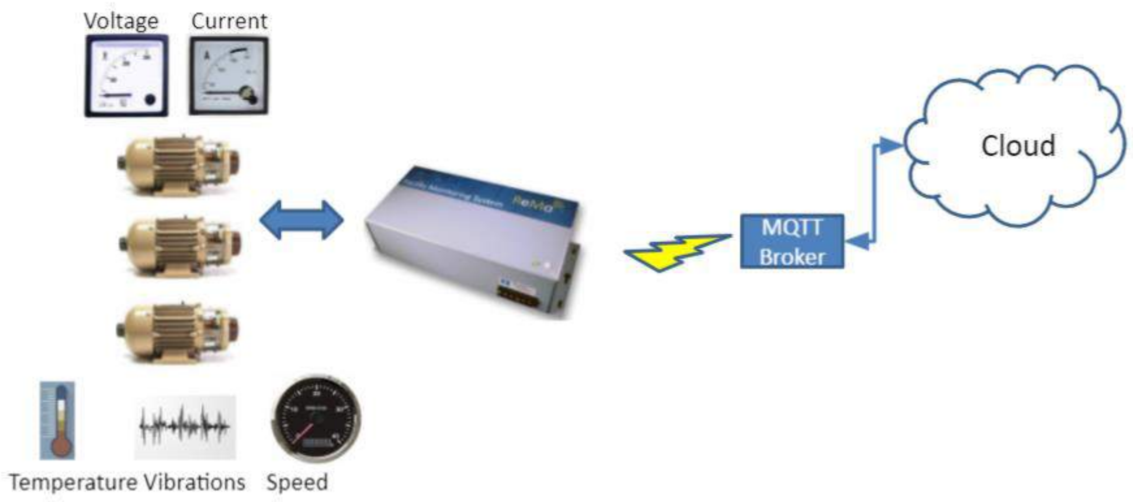
8. Features: Salient features for the product (in the bullet form)
    - It includes:
        - ✓ Compulsory/Mandatory feature
        - ✓ Nice to have feature
        - ✓ Functional feature (UI-User Interface part)
        - ✓ Non-functional feature (UX-User Experience part)
9. Description: Each feature in the above section is explained in detail with Algorithm, work flow, diagram, pictorial explanation, block diagram, etc.

For the **Design Documentation** part following were the subparts to be filled up:

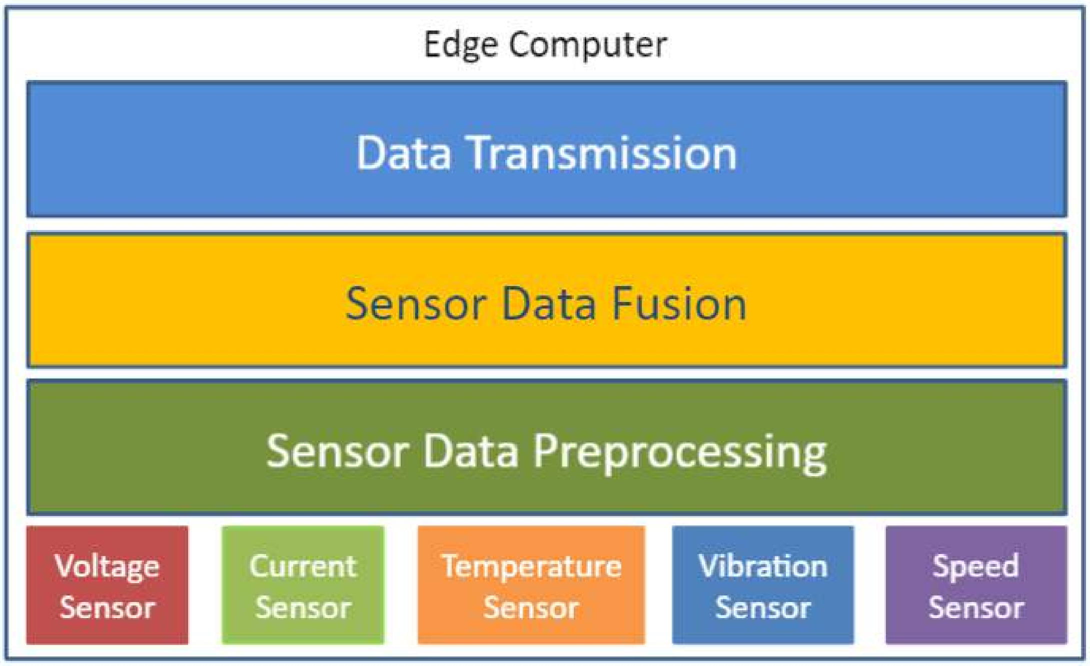
1. Overview of Design & Architecture: Contains Architectural Block Diagram and Entire Design (Modular structured design)
2. Modules: Listed Filename, Constants, Variables, Functions
3. Testing: Separate spreadsheet to be created based on the following details:
   - ✓ Testing Plan
   - ✓ Tests Description
   - ✓ Test(Number, Description)
   - ✓ Test Inputs
   - ✓ Test Expected Outputs
4. Appendix: Raw data or extra information, which provide readers the additional insights on the topic being discussed in the project document.

Following is briefly explained (with the help of diagrams) the Concept, Architecture and Implementation plan of the project to be executed.

CONCEPT



ARCHITECTURE



ARCHITECTURE

- Motor Basics
- Sensors

Voltage • Voltage 1 • Voltage 2 • Voltage 3	Temperature • Temperature 1 • Temperature 2 • Temperature 3
Current • Current 1 • Current 2 • Current 3	Vibrations • Vibration1 • Vibration 2 • Vibration 3

Speed • Speed 1 Direction 1 • Speed 2 Direction 2 • Speed 3 Direction 3

- Sensor Data Preprocessing
    - Data Sanitisation

  Sensor Data Fusion
    - Combine individual sensor data in a mathematical model to derive more information

  Data Transmission
    - Transmit the data over a Communication Interface upstream
        - MQTT Broker
        - Wireless Communication Interfaces
        - Wired Communication Interfaces

EDGE Computer:

- Get Data/Status from Device using Get API
- Store Data/Status locally
- Pre-process Data/Status (optional)

- Send the Data/Status to MQTT Broker
    - ✓ Publish to data or status as JSON string
- Get the Command (if any) from MQTT Broker
    - ✓ Subscribe to command Topic
- Send the Command to the Device using Set API

REVISION HISTORY

Revision Number	Prepared By	Revision Date	Revision Details
1.0	Nirzari Shah	8th July, 2021	Coded Frontend elements: Favicon, Banner, Menu Bar, Status Bar & Footer (On the basis of the UI/UX Design Feedback we received by Hemant Sir on 7th July, 2021)
2.0	Nirzari Shah	11th July, 2021	*Embedded the code for Current Day, Date and Time *Developed basic Client Area of three individual tabs: Home, Masters and Provision Drives
3.0	Nirzari Shah	16th July, 2021	Developed detailed Client Area of 3 individual tabs: Home, Masters and Provision Drives Revised on the basis of the edits suggested in Client Area, by Hemant Sir as per meet on 13th July,2021
4.0	Lakshya Pratap	17th July, 2021	Added horizontal scrolling attribute to the Drive Register Master Table
5.0	Lakshya Pratap	19th July, 2021	Linked MySQL Database to the Frontend part of the website
6.0	Nirzari Shah	20th July, 2021	Inserted iconic buttons (with mouse hover functions) in place of textual buttons for the tabular data on Masters Page: *Edit by Pencil *Save by Tick *Delete by Cross *Add Row by Plus Inserted real-time graph for the Data Analysis, which shows health status of the DC Motors on the Home Page Final Project Review was done by Hemant sir, on 21st July, 2021.

FEATURES

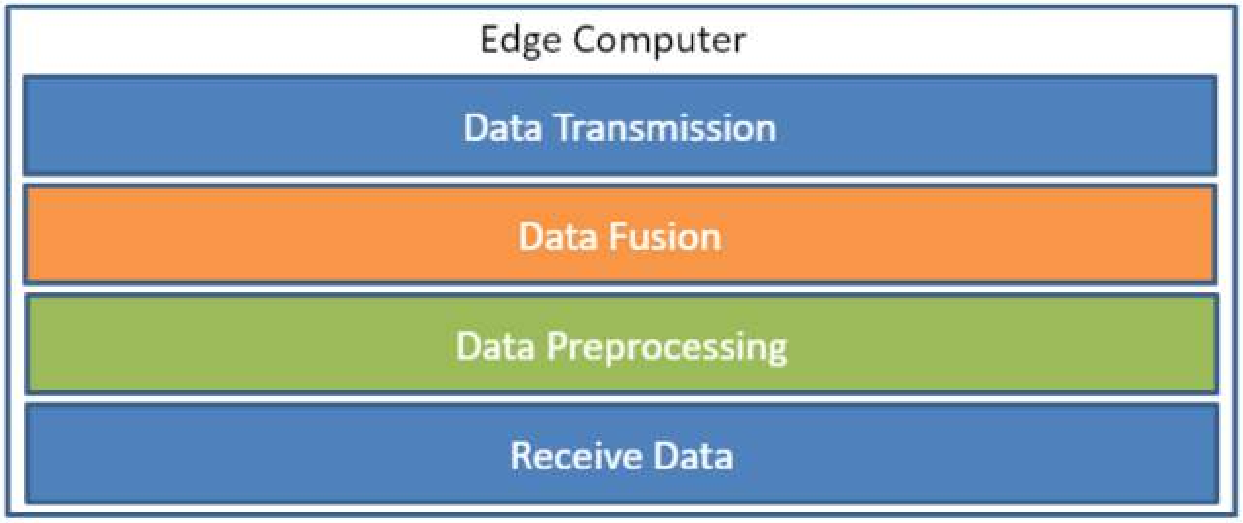
1. **Compulsory Feature:**
    a. Favicon of the company
    b. Logo of the company
    c. Banner of the company
    d. Status Bar
    e. Menu Bar
    f. Client Area
    g. Footer

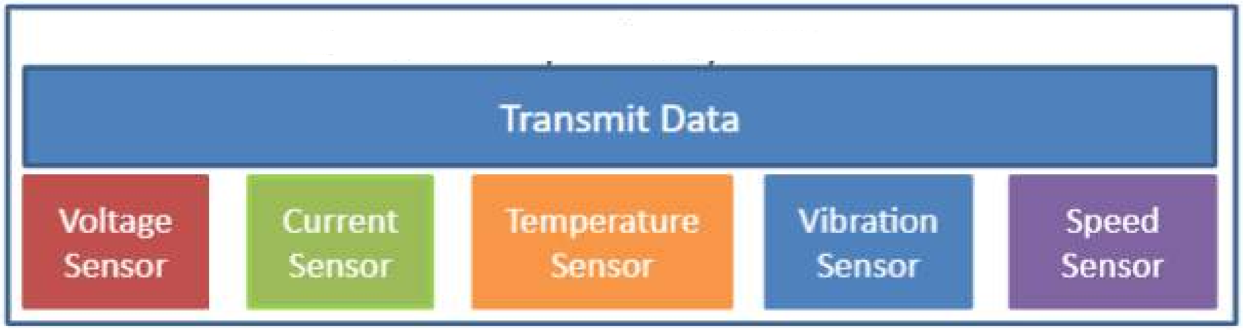
2. **Nice to have Feature:**
    a. Graphical presentation of Health Status of DC Motor on Home Page
    b. Current Day-Date-Time display on the Status Bar
    c. User Identity on the Status Bar

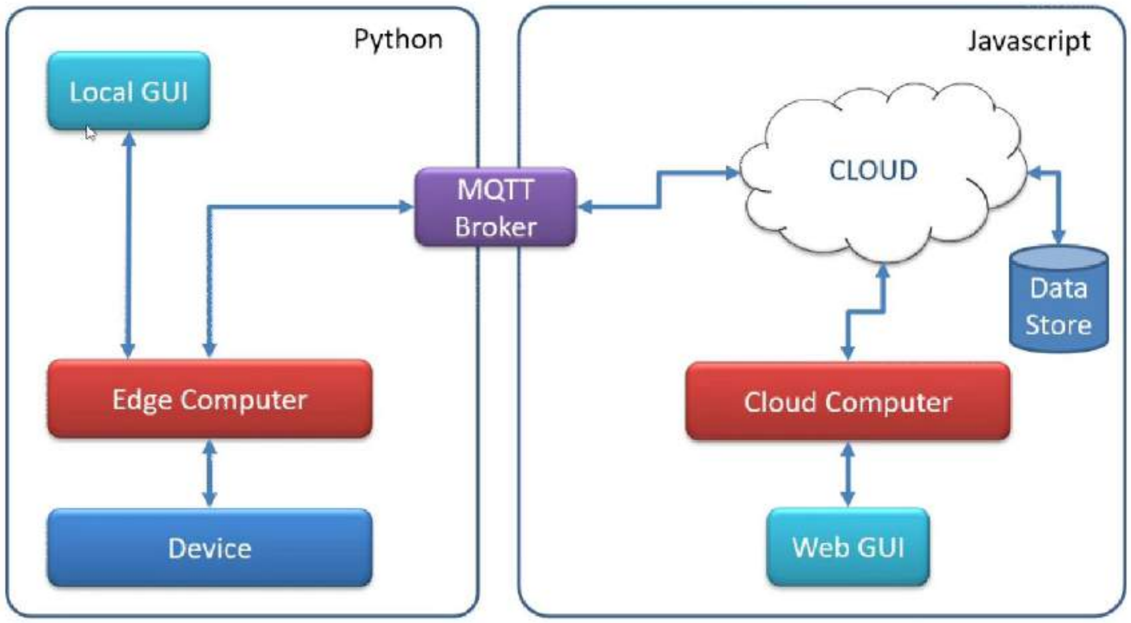
3. **Functional Feature:**
    a. Graphical presentation of Health Status of DC Motor on Home Page
    b. Edit: Row Editing of the database available on Masters Page by clicking on "pencil" button in the "Actions" column.
    c. Save: Saving of the database of row edited on Masters Page by clicking on "tick" button in the "Actions" column.
    d. Delete: Deleting a row of the database available on Masters Page by clicking on "cross" button in the "Actions" column.
    e. Add Row: Adding a row to the database on Masters Page by clicking on "plus" button in the "Actions" column.

4. **Non-Functional Feature:**
    a. The database of the website is maintained with the help of MySQL on local server
    b. The website is completely responsive in nature.

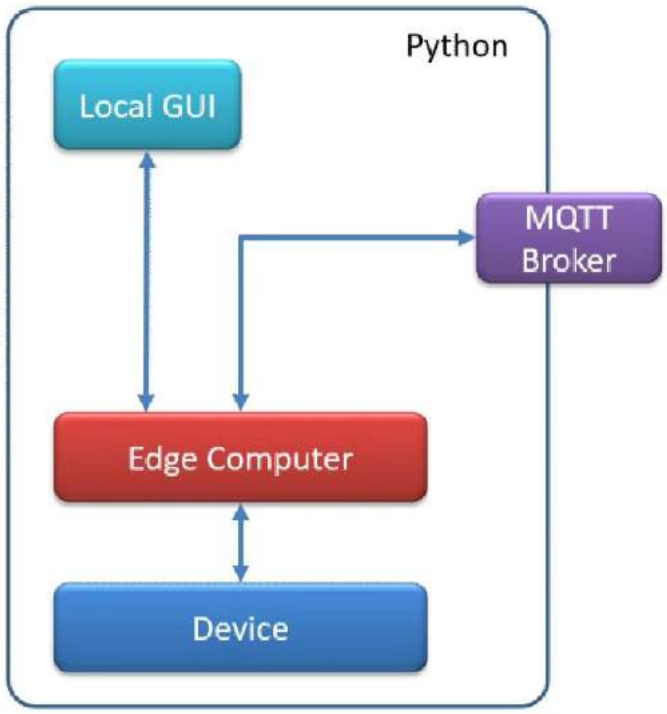
IMPLEMENTATION







PYTHON TEAM



Python team worked on Device Simulator function, which has three main modules, which are:

1. Local GUI
2. Edge Computer
3. Device

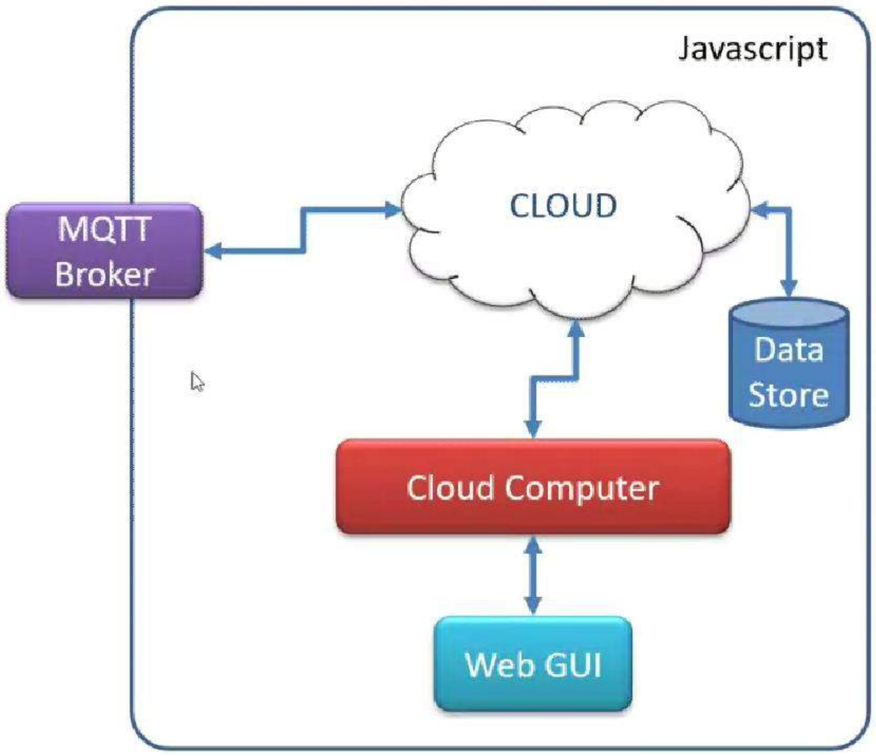
The diagram besides depicts the Device Simulator Architecture. The simulator consists of the Device Module as the core component. The module simulates the functionality of a physical Device vis-à-vis the Register Architecture. The registers of the device are internal memory locations with unique address and are 8-bits, 16-bits or 32-bits wide. The details of the registers are available in the datasheet of the device.

These registers are exposed to the application using the device module through a set of Application program Interface functions. These functions save or retrieve data to or from the register locations.

The device module is implemented as a python module. This can be imported in any Python application and used to simulate the Device. To test this feature of the device module, an Application is developed which exercises the device module. The Application communicates with the user through a Front End. The Front End User Interface may be a Command Line Interface (CLI) or a Graphical User Interface (GUI). The FE User Interface accepts verbose commands in the CLI or events from the GUI and maps them to appropriate command stream to the Application. This makes the application modular.

The Applications based on the user commands invokes the API function and communicates with the device module. The results of this function execution in the device module are published as appropriate to the Cloud based MQTT Broker using the MQTT protocol over TCP/IP. The data is published in JSON format.

JAVASCRIPT TEAM

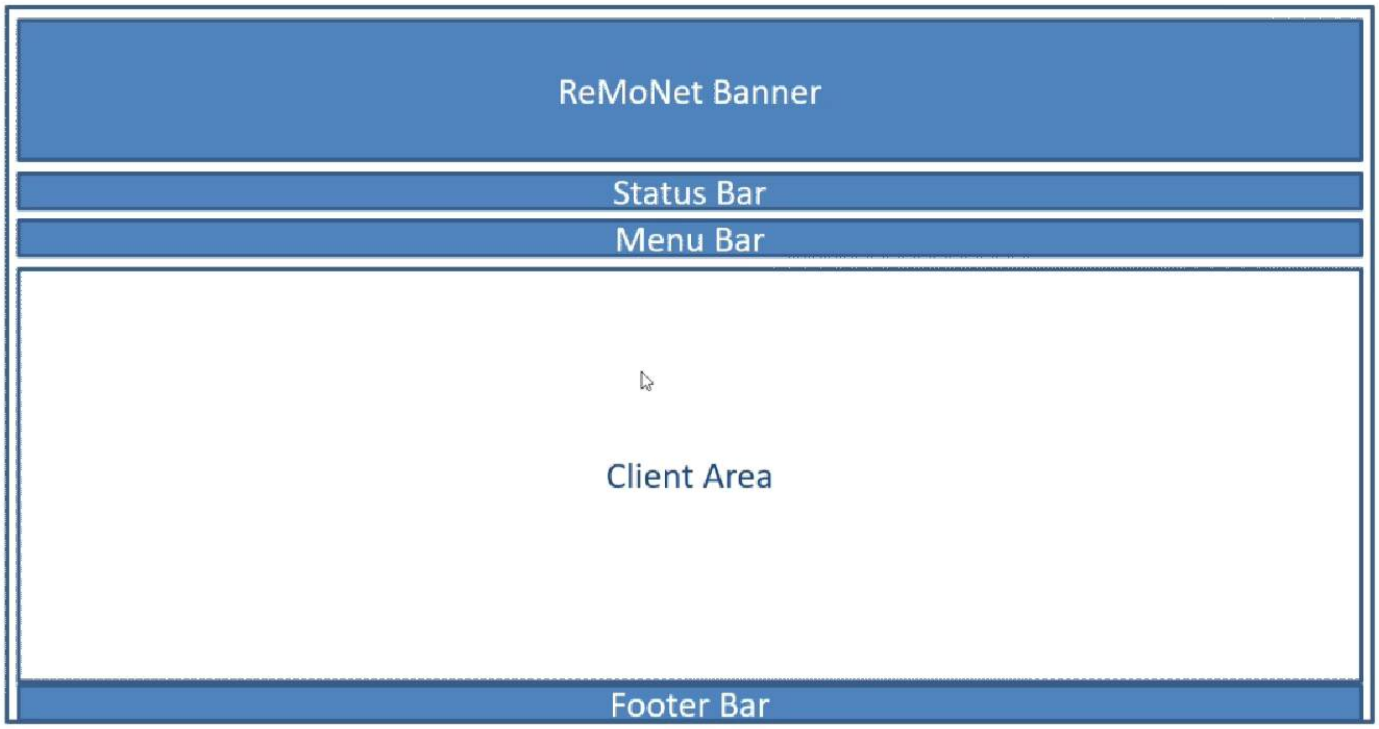


We were detailed about the layout as shown on the next page.

- Header: ReMoNet Banner, Status Bar, Menubar
- Middle: Client Area
- Footer: Footerbar

- **ReMoNet Banner** should include : Branding of ReMoNet, i.e. Logo, Favicon, etc
- **Status Bar** should include :
  - Current Date and Time- Timezone (on Left hand side)
  - User Location
  - User Info: Login ID and Role (on Right hand side)
  - SignIn/SignOut Link (Currently Invisible)
  - Profile Link (Currently Invisible)
  - Micro portal name (at Center)
- **Menubar** should include :
  - Command Menu (to be applied on the Client Area)
- **Client Area** should incude : Show Tabular Data
- **Footerbar** should include :
  - Communication Status
  - Copywrite message of Shalaka Connected Devices LLP

LAYOUT



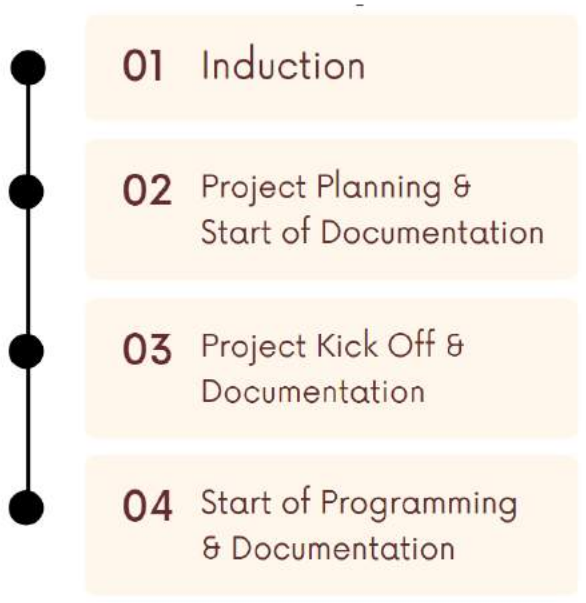
What does Client Area exactly contains:

Tabular Data

- Column 1: Gateway ID
  - ✓ Hover to show Location- Currently Inactive
  - ✓ Hyperlink to pop-up Gateway Config Panel-currently setup sampling interval
- Column 2: DID
  - ✓ Hover to show make and model
  - ✓ Hyperlink to pop-up Device Panel
- Column 3 to 6: Important Parameters
  - ✓ Speed
  - ✓ Direction
  - ✓ Temperature
  - ✓ Vibrations
  - ✓ Current
  - ✓ Voltage

PROJECT TIMELINE (we followed up for the project)

PROJECT TIMELINE ON WEEKLY BASIS:



01 Induction

02 Project Planning & Start of Documentation

03 Project Kick Off & Documentation

04 Start of Programming & Documentation

First week we were introduced to PS station

Second week we were allotted the respective projects in the groups of 4-5

Third week we were able to start the project’s documentation

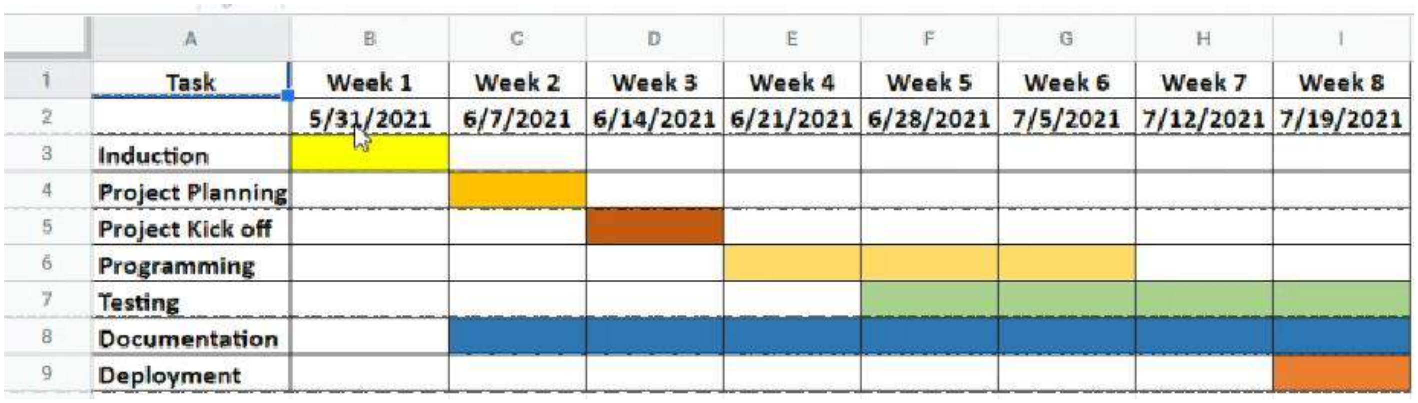
Fourth week we installed required tools and platforms and learnt programming

05 Start of Testing & Programming & Documentation

06 Documentation & Testing & End of Programming

07 Documentation & Testing

08 Winding up of Documentation & Testing & Deployment



	A	B	C	D	E	F	G	H	I
1	Task	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8
2		5/31/2021	6/7/2021	6/14/2021	6/21/2021	6/28/2021	7/5/2021	7/12/2021	7/19/2021
3	Induction	🟨							
4	Project Planning		🟧						
5	Project Kick off			🟫					
6	Programming				🟨	🟨	🟨		
7	Testing					🟩	🟩	🟩	🟩
8	Documentation		🟦	🟦	🟦	🟦	🟦	🟦	🟦
9	Deployment								🟧

PHASE 2: UI/UX Designing

Step 1: Information Architecture
User Experience (UX) Design

A website's information architecture has two main components: identification and definition of site content and functionality. the underlying organization, structure and nomenclature that define the relationships between a site's content/functionality.

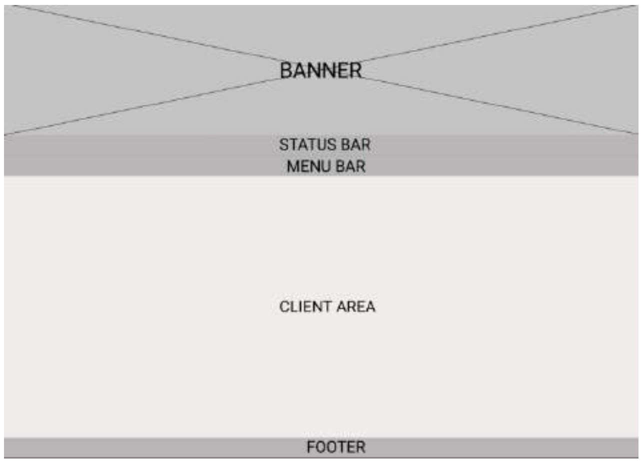
Software used for this purpose: **Figma**

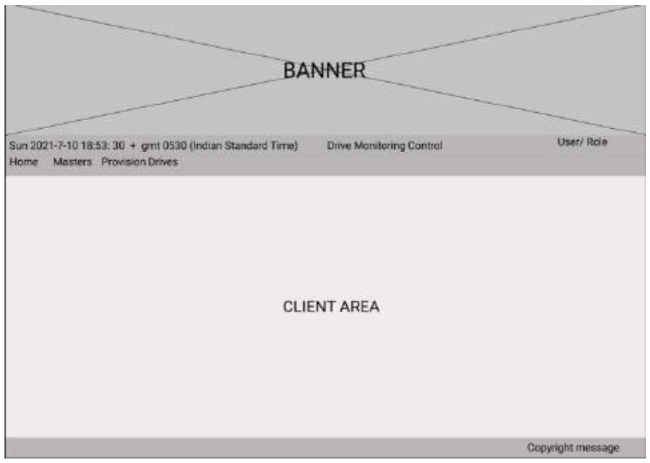
Wireframes were created in Figma which reflected the information architecture of the website.

A website wireframe, also known as a page schematic or screen blueprint, is a visual guide that represents the skeletal framework of a website. Wireframes are created for the purpose of arranging elements to best accomplish a particular purpose.

Link to the wireframes:
https://www.figma.com/proto/v4IHQPIUhzuKVMPF3wASVG/ReMoNet_DC_Motor_Control?node-id=201%3A103&scaling=scale-down&page-id=201%3A102&starting-point-node-id=201%3A103

Screenshot of the wireframes in Figma:





Step 2: Visual Designing

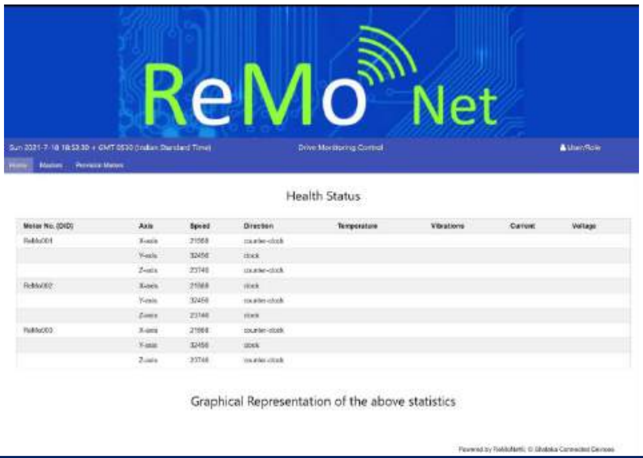
User Interface (UI) Design

Visual design is the use of imagery, color, shapes, typography, and form to enhance usability and improve the user experience. Visual design as a field has grown out of both UI design and graphic design.

Software used for this purpose: **Figma**

Designed the user interface by inserting graphics and visuals to the wireframes (in the step1).
Link to the Screen Designs:
https://www.figma.com/file/v4IHQPIUhzuKVMPF3wASVG/ReMoNet_DC_Motor_Control?node-id=0%3A1

Screenshot of the Screen Designs in Figma:

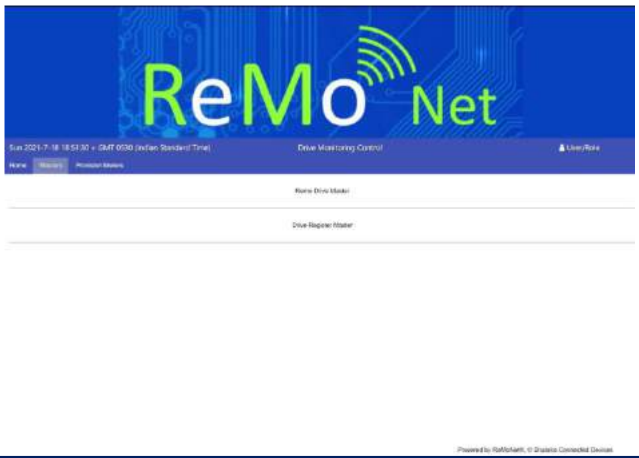


Health Status

Motor No. (DID)	Axis	Speed	Direction	Temperature	Vibrations	Current	Voltage
ReMo001	X-axis	21568	counter-clock				
	Y-axis	32456	clock				
	Z-axis	23746	counter-clock				
ReMo002	X-axis	21568	clock				
	Y-axis	32456	counter-clock				
	Z-axis	23746	clock				
ReMo003	X-axis	21568	counter-clock				
	Y-axis	32456	clock				
	Z-axis	23746	counter-clock				

Graphical Representation of the above statistics

Powered by ReMoNet, © Shalaka Connected Devices



Remo Drive Master

Drive Register Master

Powered by ReMoNet, © Shalaka Connected Devices

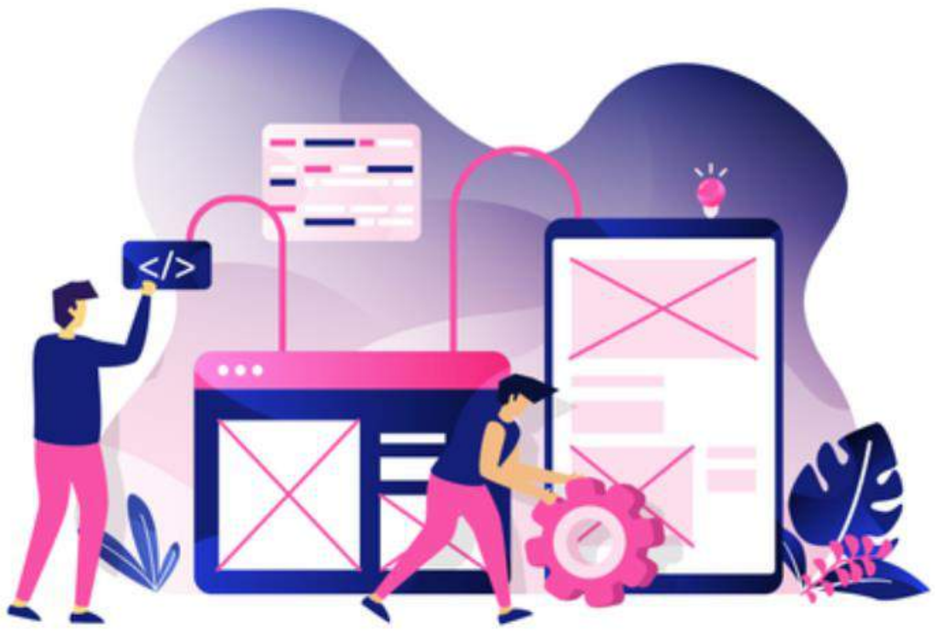
Step 3: Prototyping of website

A prototype is an early sample, model, or release of a product built to test a concept or process. It is a term used in a variety of contexts, including semantics, design, electronics, and software programming. A prototype is generally used to evaluate a new design to enhance precision by system analysts and users.

Software used for this purpose: **Figma, Google Sites**

Working prototype of the website was created in both Figma and Google Sites
Link to the Figma Prototype:
https://www.figma.com/proto/v4IHQPIUhzuKVMPF3wASVG/ReMoNet_DC_Motor_Control?node-id=201%3A69&scaling=scale-down&page-id=201%3A68&starting-point-node-id=201%3A69

Link to the Google Sites Prototype:
https://sites.google.com/view/dc-motor-control-project



PHASE 3: Website Development

Tabular summarization of all versions was done above in the
‘ “Revision History” in the Phase 1 : Documentation‘ part.
Following are the details of each of that version mentioned in the table.

Version 1.0:

Frontend coding for static elements was there in this version:

- Favicon
- Logo
- Banner
- Status Bar
- Menu Bar
- Footer

Tables in the Client Area are static.

Version 2.0:

In this version, improvements were made to the client area.

The tables were made dynamic in this version, which had clickable options: “Edit”, “Save”, “Delete” and “Add Row”

1. “Edit” button allow user to change details of that particular row of the table.

2. “Save” button allow user to save the changes made while editing the row.

3. “Delete” button allow user to delete that particular row from the table.

4. “Add Row” button allow the user to add a row with all column details filling up.

These all action buttons will reflect the changes not only in the table shown, but will also be reflected in the MySQL Database.

In this version, also embedded the code for Current Day, Date and Time in the Status Bar of website.

Version 3.0 & 4.0:

In this version, changes were made to the client area, Home’s and Master’s Page. The two tables on the Home Page were to be combined into one table. On the Master Page 3 accordions were to be removed and some detailing of the other 2 accordions was to be done.

Version 5.0:

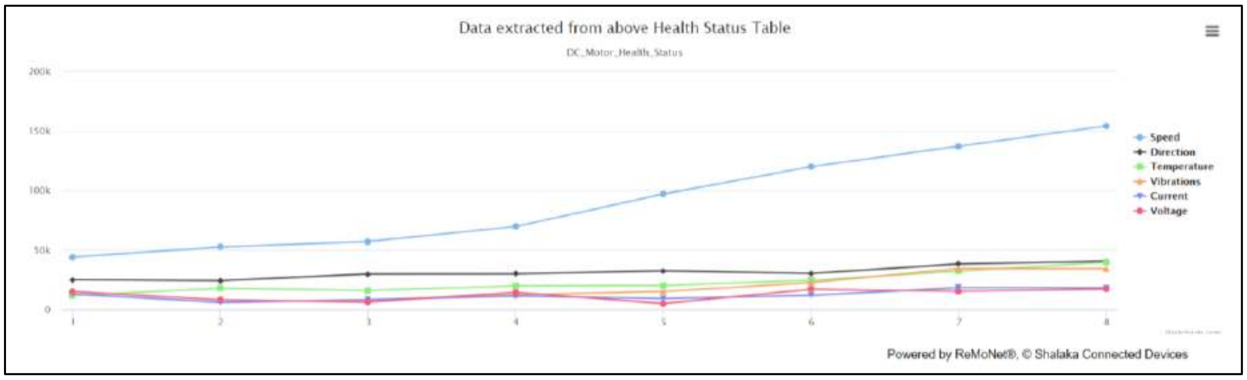
In this version, improvements were made to the client area. MySQL Database was created and linked to the website front-end code with the help of PHP backend language.

Version 6 – (Final Version):

Inserted and replaced the textual buttons (Edit, Save, Delete and Add Row) by iconic buttons (Pencil, Tick, Cross, Plus) and added real-time graphical representation of the health status of DC Motors on Home page.

Text Button	Icon Button (without Mouse Hover)	Icon Button (Mouse Hover)
Edit	✏	✏
Save	✓	✓
Delete	✕	✕
Add Row	+	+

Real-Time Graph based on the MySQL Database:



Process followed up in setting up the website:

- Favicon of the company added

  Favicon:

- Banner of the ReMoNet added

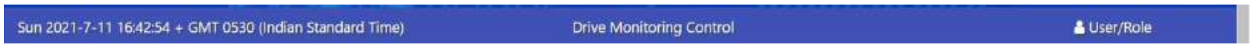
  Banner:

- Status Bar added, which included: Current Day, Date and Time (Left Side)

  Drive Monitoring Control (Micro portal) (Center)

  User/Role (Right Side)

  Status Bar:

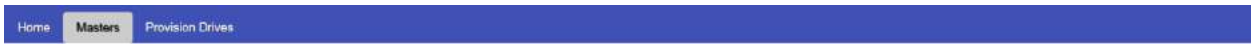


- Menu Bar added, which included: Home Page Tab

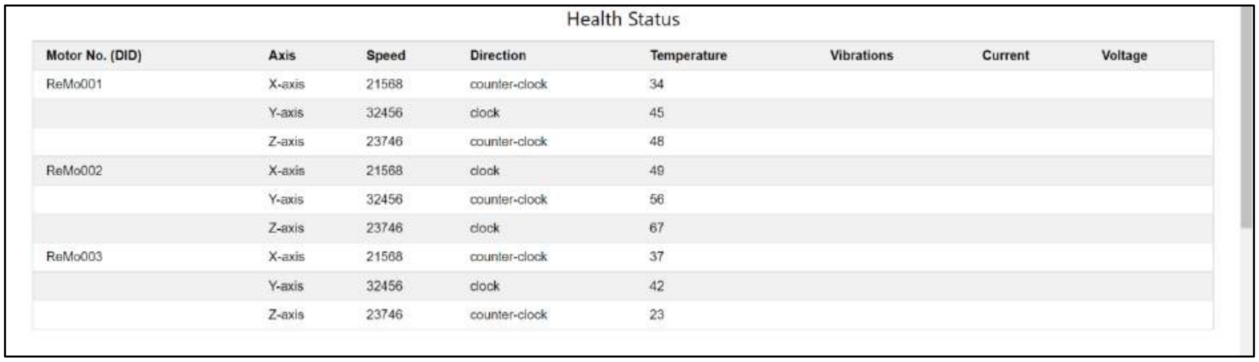
  Masters Page Tab

  Provision Drives Tab

  Menu Bar:

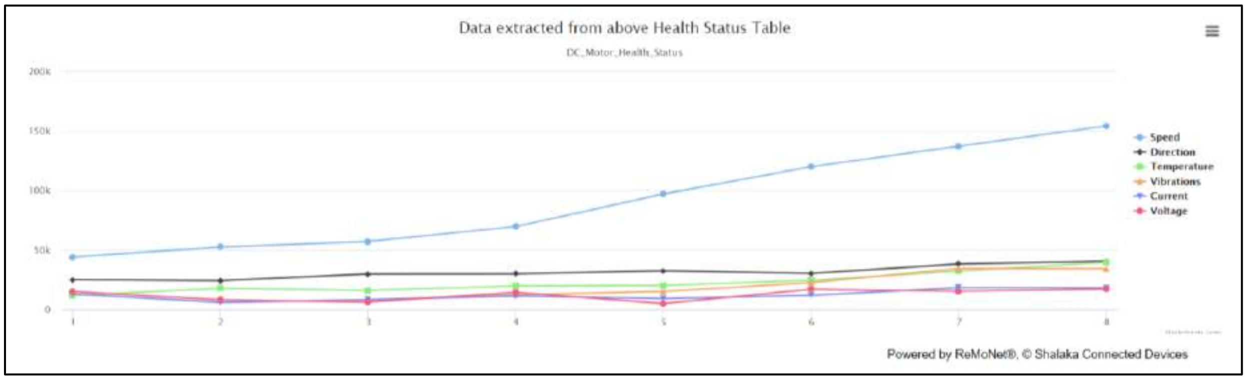


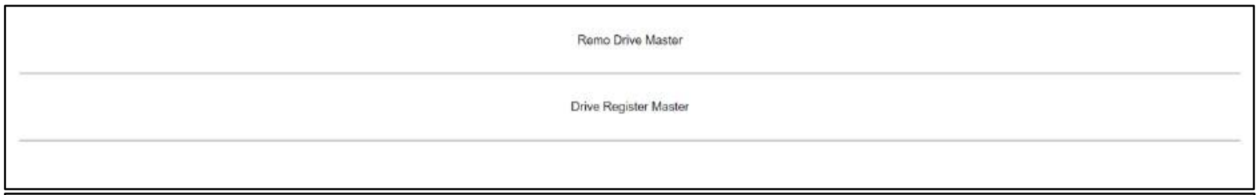
- Client Area, for all 3 pages added
  Client Area:

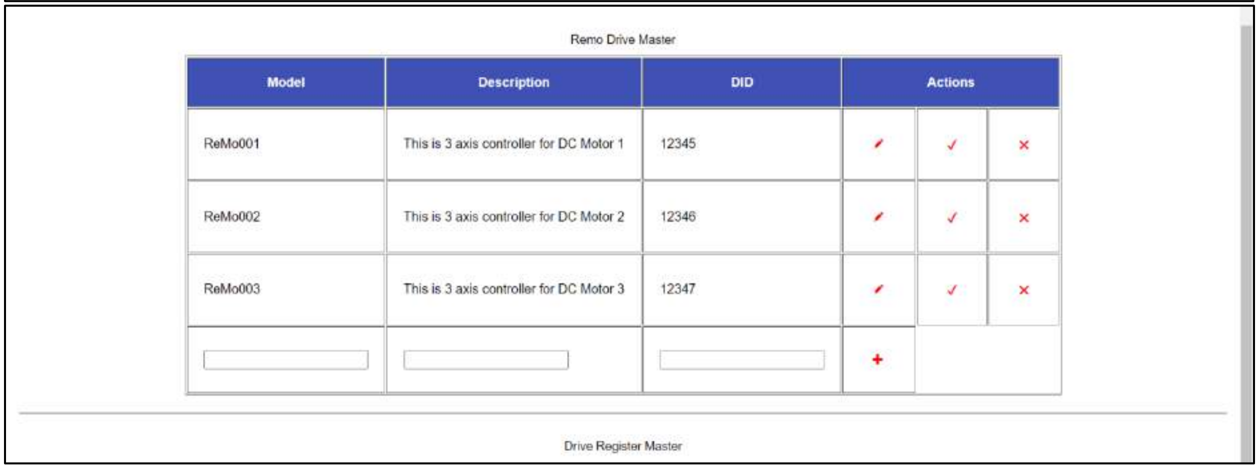


Health Status

Motor No. (DID)	Axis	Speed	Direction	Temperature	Vibrations	Current	Voltage
ReMo001	X-axis	21568	counter-clock	34			
	Y-axis	32456	clock	45			
	Z-axis	23746	counter-clock	48			
ReMo002	X-axis	21568	clock	49			
	Y-axis	32456	counter-clock	56			
	Z-axis	23746	clock	67			
ReMo003	X-axis	21568	counter-clock	37			
	Y-axis	32456	clock	42			
	Z-axis	23746	counter-clock	23			







Remo Drive Master

Model	Description	DID	Actions		
ReMo001	This is 3 axis controller for DC Motor 1	12345	✎	✓	✕
ReMo002	This is 3 axis controller for DC Motor 2	12346	✎	✓	✕
ReMo003	This is 3 axis controller for DC Motor 3	12347	✎	✓	✕
			+		

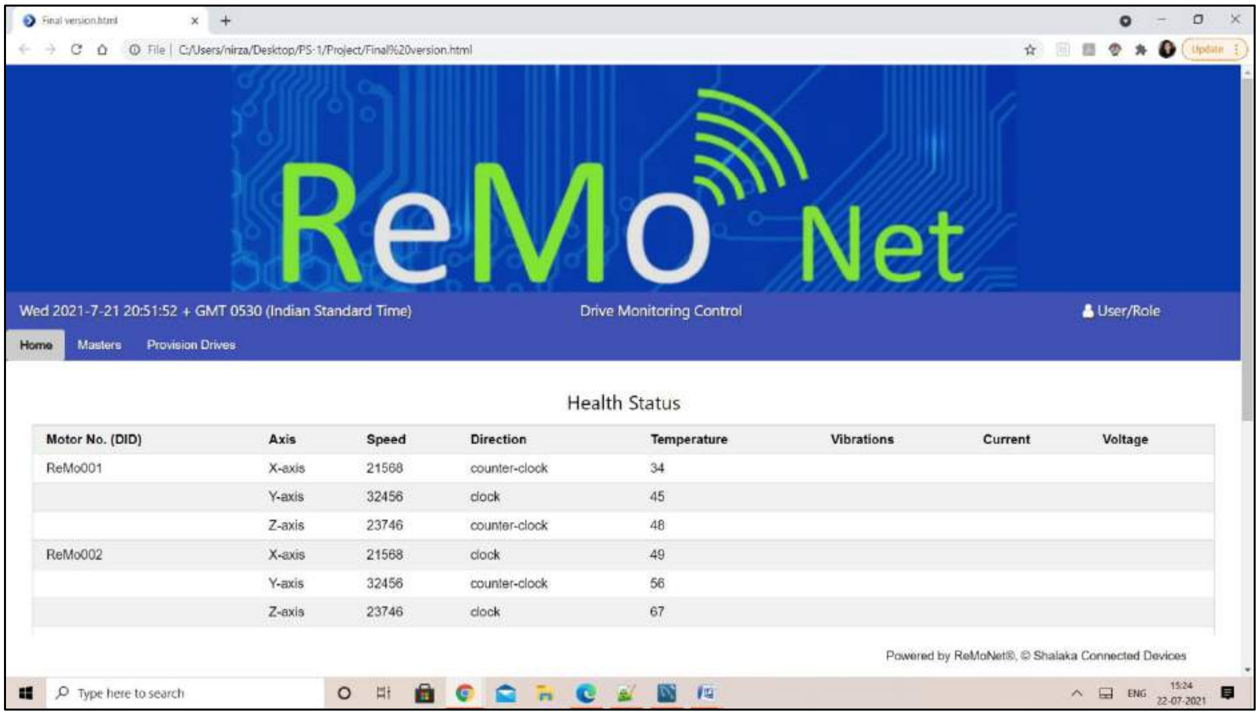
Drive Register Master

- Footer added
  Footer:

Powered by ReMoNet®, © Shalaka Connected Devices

Final version website screens:

Home Page: Health Status Table



Wed 2021-7-21 20:51:52 + GMT 0530 (Indian Standard Time)

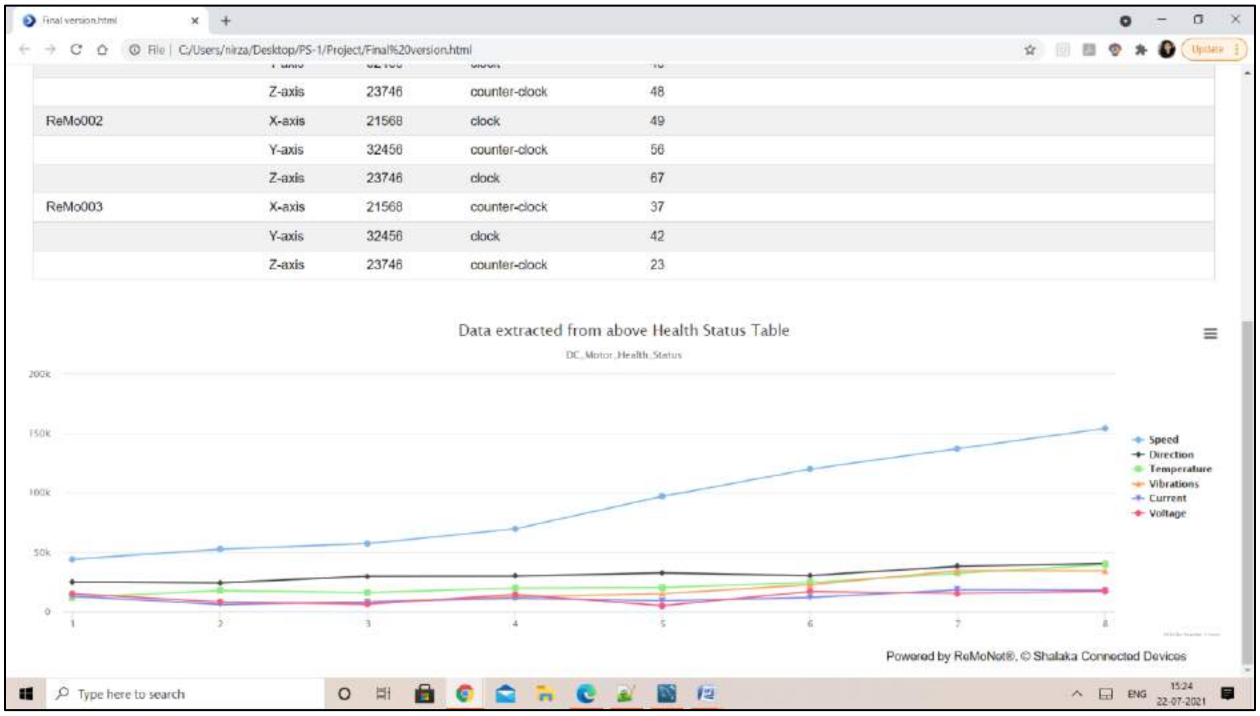
Drive Monitoring Control

Health Status

Motor No. (DID)	Axis	Speed	Direction	Temperature	Vibrations	Current	Voltage
ReMo001	X-axis	21568	counter-clock	34			
	Y-axis	32456	clock	45			
	Z-axis	23746	counter-clock	48			
ReMo002	X-axis	21568	clock	49			
	Y-axis	32456	counter-clock	56			
	Z-axis	23746	clock	67			

Powered by ReMoNet®, © Shalaka Connected Devices

Home Page: Real-time Graph for Data Analysis on Health Status



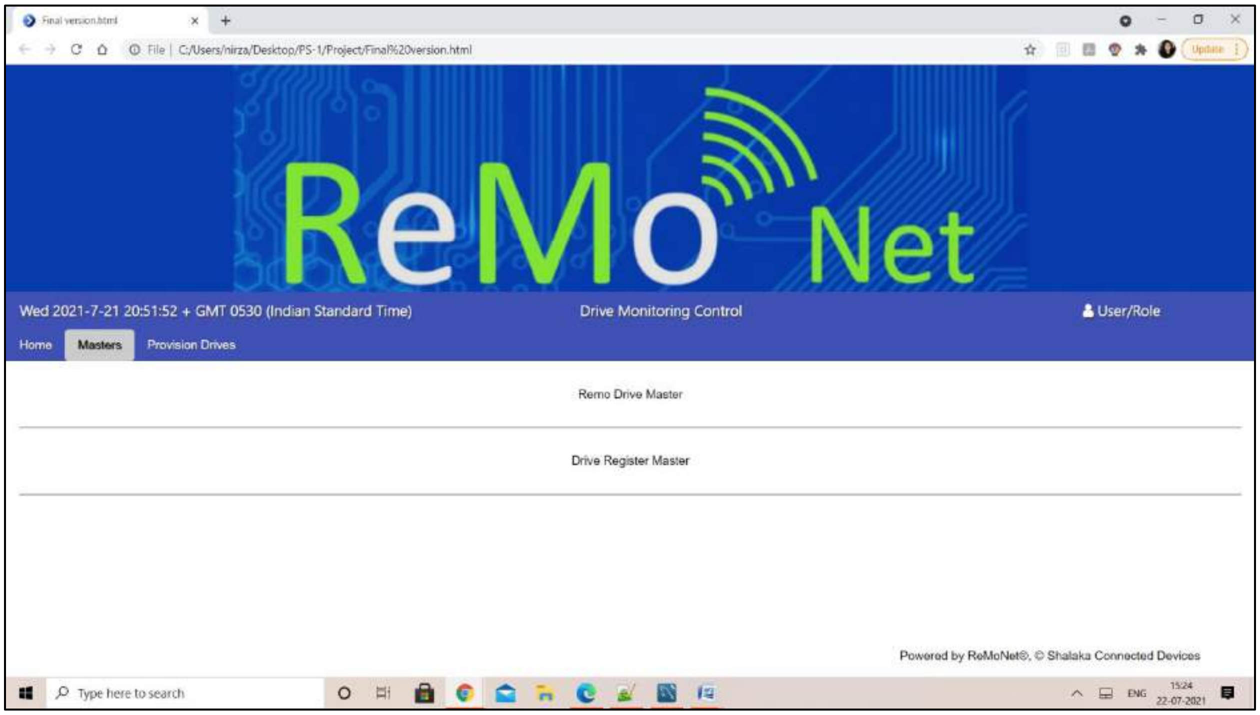
	Z-axis	23746	counter-clock	48			
ReMo002	X-axis	21568	clock	49			
	Y-axis	32456	counter-clock	56			
	Z-axis	23746	clock	67			
ReMo003	X-axis	21568	counter-clock	37			
	Y-axis	32456	clock	42			
	Z-axis	23746	counter-clock	23			

Data extracted from above Health Status Table

DC_Motor_Health_Status

Powered by ReMoNet®, © Shalaka Connected Devices

Masters Page: Two Accordions menu for two tabular database



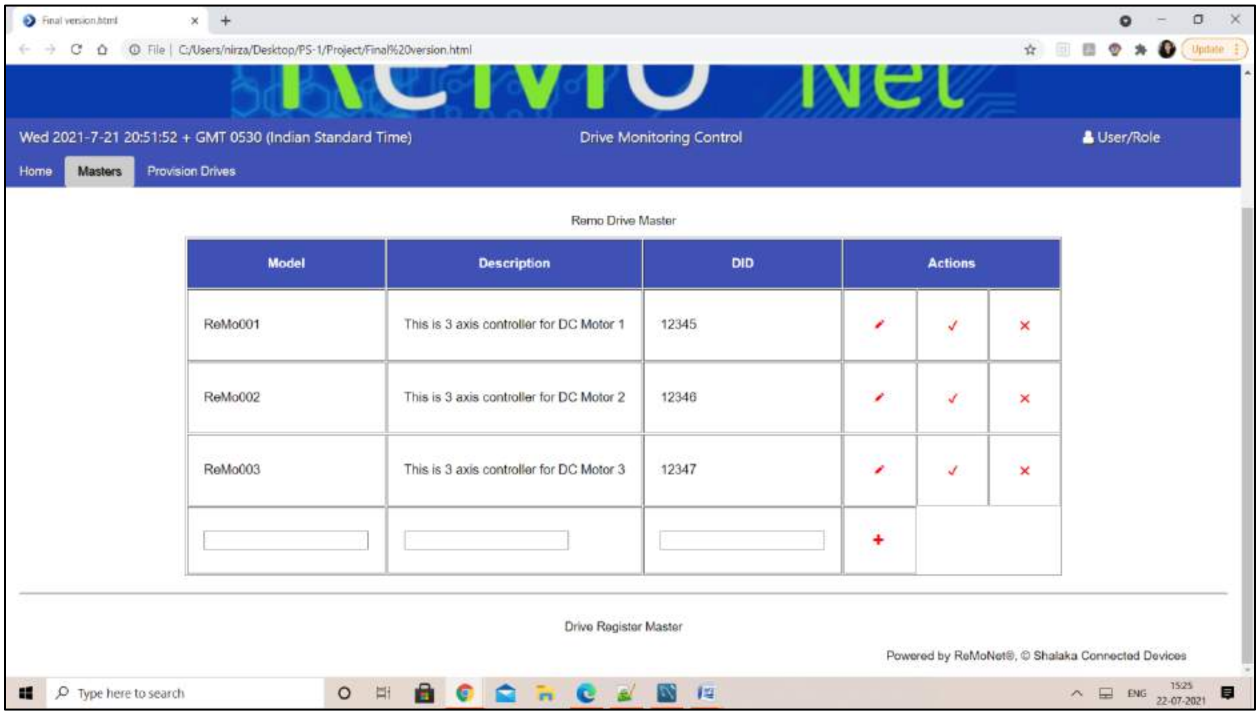
Wed 2021-7-21 20:51:52 + GMT 0530 (Indian Standard Time) Drive Monitoring Control User/Role

Remo Drive Master

Drive Register Master

Powered by ReMoNet®, © Shalaka Connected Devices

Home Page: Tabular database inside accordion



Wed 2021-7-21 20:51:52 + GMT 0530 (Indian Standard Time) Drive Monitoring Control User/Role

Remo Drive Master

Model	Description	DID	Actions		
ReMo001	This is 3 axis controller for DC Motor 1	12345	✎	✓	✕
ReMo002	This is 3 axis controller for DC Motor 2	12346	✎	✓	✕
ReMo003	This is 3 axis controller for DC Motor 3	12347	✎	✓	✕
			+		

Drive Register Master

Powered by ReMoNet®, © Shalaka Connected Devices

Conclusion ...

The two months of PS 1 program turned out to be satisfactory internship journey, It helped me learn many new skills, which are beneficial in my career prospects.

I have learnt the industry architecture and best industrial practices to be followed up. I learnt about what Industry 4.0 is, what IIoT is and about the Product Development Lifecycle and it's various stages. I understood what deliverables would be required and who all participants would be part of every stage of the Product Development Lifecycle. I have acquired in-depth knowledge of Documentation of two types: Feature documentation and Design documentation. I gained the knowledge of MQTT Broker. The two group discussions enlightened my knowledge in the field of IoT.

I have learnt implementation of HTML, CSS, Bootstrap and various classes present in them to make a responsive, user friendly and mobile compatible web page. I have also learnt implementation of Javascript functions based on user controls and various ready-to-use libraries present. I also practiced Angular, Node.js, My SQL, Mosquitto. I have gained the skills to work in the field of development of cloud web applications.

I have acquired skill of modularizing a project, which saves time and reduces rework and effort, while creating new version or while performing customization. It also helps in tracking and analyzing the project data faster. Throughout my PS 1 program, I have managed to set realistic expectations while setting a target goal for completing the project tasks.

I have also acquired many soft-skills useful in fulfilling any responsibility and handling tasks. This PS 1 program, being my first internship has resulted in quite good industry exposure.

Appendices ...

Appendix 1: Details of the Software Packages used in the project

- **HTML:** The HyperText Markup Language, or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets and scripting languages such as JavaScript.

- **CSS:** Cascading Style Sheets is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

- **Javascript:** JavaScript, often abbreviated as JS, is a programming language that conforms to the ECMAScript specification. JavaScript is high-level, often just-in-time compiled, and multi-paradigm. It has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions.

- **Python:** Python is an interpreted high-level general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant indentation.

- **Node.js:** Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser.

- **Angular:** Angular is a TypeScript-based free and open-source web application framework led by the Angular Team at Google and by a community of individuals and corporations. Angular is a complete rewrite from the same team that built AngularJS.

- **MySQL:** MySQL is an open-source relational database management system. Its name is a combination of "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language.

Appendix 2: Information on MQTT

- MQTT stands for MQ Telemetry Transport
    - Previously was known as Message Queuing Telemetry Transport.

  Designed by Andy Stanford-Clark (IBM) and Arlen Nipper in 1999 for connecting Oil Pipeline telemetry systems over satellite

  Started as a proprietary protocol

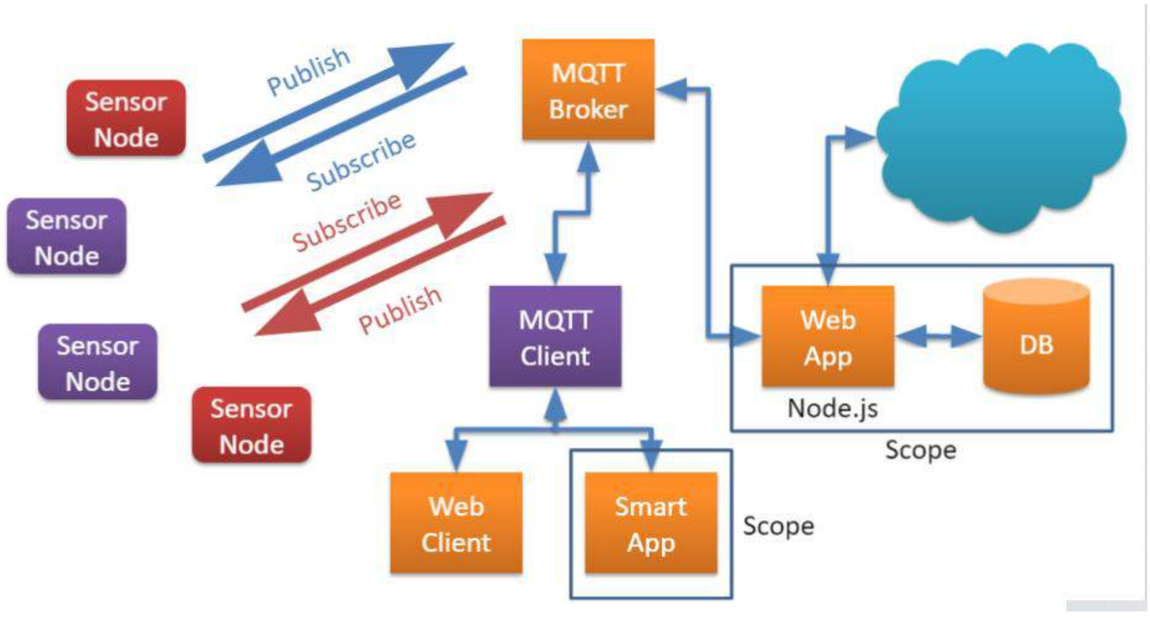
  Released Royalty free in 2010 and became an OASIS standard in 2014

Fast becoming one of the main protocols for IOT (internet of things) deployments

- MQTT Versions
    - Designed in 1999
        - Has been in use for many years
        - Designed for TCP/IP networks over TCP
        - Make it ideal for use in constrained environments
    - MQTT v3.1.0
        - Current Stable
    - MQTT v3.1.1
        - In Common Use
    - MQTT v5
        - Latest Approved
    - MQTT-SN
        - Designed to work over UDP, ZigBee and other transports
- Lightweight publish/subscribe messaging protocol
    - Designed for M2M (machine to machine) telemetry
        - In low bandwidth environments
        - Where network is expensive, has low bandwidth or is unreliable
        - When run on an embedded device with limited processor or memory resources (constrained environments)
- Features
    - The publish/subscribe message pattern to provide one-to-many message distribution and decoupling of applications
    - A messaging transport that is agnostic to the content of the payload
    - The use of TCP/IP to provide basic network connectivity
    - A small transport overhead (the fixed-length header is just 2 bytes), and protocol exchanges minimised to reduce network traffic

- A mechanism to notify interested parties to an abnormal disconnection of a client using the Last Will and Testament feature
- Three qualities of service for message delivery:
    - "At most once", where messages are delivered according to the best efforts of the underlying TCP/IP network
        - Message loss or duplication can occur
        - This level could be used, for example, with ambient sensor data where it does not matter if an individual reading is lost as the next one will be published soon after
    - "At least once", where messages are assured to arrive but duplicates may occur
    - "Exactly once", where message are assured to arrive exactly once
        - This level could be used, for example, with billing systems where duplicate or lost messages could lead to incorrect charges being applied

Industrial Internet Of Things

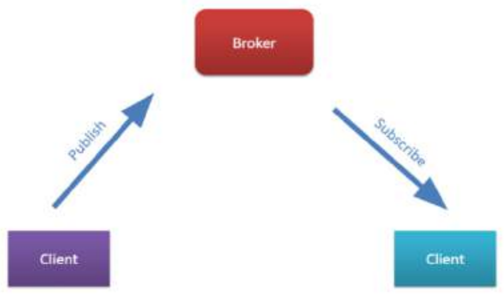


- MQTT Publish/Subscribe
    - A Publish Subscribe messaging protocol allowing a message to be published once and multiple consumers (applications / devices) to receive the message bu subscribing

        - Provides decoupling between the producer and consumer(s)
    - A producer sends (publishes) a message (publication) on a topic (subject)
    - A consumer subscribes (makes a subscription) for messages on a topic (subject)
    - A message server / broker matches publications to subscriptions
        - If no matches the message is discarded
        - If one or more matches the message is delivered to each matching subscriber/consume

- MQTT Clients
    - MQTT End devices
    - Can Publish and/or Subscribe to Topic(s)
        - A <u>Topic</u> forms the namespace
            - Is hierarchical with each "sub topic" separated by a /
    - Client(s) Publish Message for a Topic
        - A publication may be retained
            - A publisher can mark a publication as retained
            - The broker / server remembers the last known good message of a retained topic
            - The broker / server gives the last known good message to new subscribers
                - The new subscriber does not have to wait for a publisher to publish a message inorder to receive its first message
- MQTT Clients
    - A subscriber can subscribe to an absolute topic or can use wildcards
        - Single-level wildcards "+"can appear anywhere in the topic string
        - Multi-level wildcards "#"must appear at the end of the string
        - Wildcards must be next to a separator
        - Cannot use wildcards when publishing

MQTT Architecture



- MQTT Broker
  - A publisher/producer sends (publishes) a message (publication) on a topic (subject)
  - A consumer subscribes (makes a subscription) for messages on a topic (subject)
  - A message server / broker matches publications to subscriptions
    - If no matches the message is discarded
    - If one or more matches the message is delivered to each matching subscriber/consumer
- MQTT Client
  - Library available for
    - Python
      - Eclipse Paho (http://eclipse.org/paho/)
      - MQTT Python client library, which implements versions 3.1 and 3.1.1 of the MQTT protocol
      - Provides a client class which enable applications to connect to an MQTT broker to publish messages, and to subscribe to topics and receive published messages
      - Also provides some helper functions to make publishing one off messages to an MQTT server very straightforward
      - Supports Python 2.7.9+ or 3.4+, with limited support for Python 2.7 before 2.7.9.
- MQTT Client
  - Library available for
    - Javascript
      - mqtt npm (https://www.npmjs.com/package/mqtt)
      - MQTT.js is a client library for the MQTT protocol, written in JavaScript for node.js and the browser
    - Android
      - Paho Android Client (https://github.com/eclipse/paho.mqtt.android)
- MQTT Broker
  - mqtt.org

  MQTT Client
  - mqtt lens plug in for Eclipse
  - mqtt.fx Java client

Appendix 3: Code of Final Version 6.0 of website

- Favicon:

(HTML code: Inside <head> tag)

```
<link rel="shortcut icon" href="favicon.ico" type="image/x-icon">
```

- Banner:

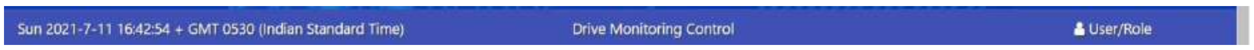
(CSS code)

```
/* Banner styling */
body, html {
  height: 100%;
  margin: 0;
  font-family: Arial, Helvetica, sans-serif;
}

.hero-image {
  background-image: linear-gradient(rgba(0, 0, 0, 0.1), rgba(0, 0, 0, 0.1)),
  url("Banner1.png");
  height: 37%;
  background-position: center;
  background-repeat: no-repeat;
  background-size: cover;
  position: relative;
}
```

(HTML code: Inside <body> tag)

```
<div class="hero-image">
</div>
```

- Status Bar:



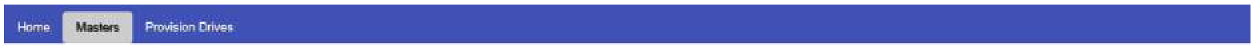
(HTML code: Inside <body> tag)

```
<div class="w3-container w3-indigo">

<h5 style="text-shadow:2px 2px 0 #444"; style="color: white"; id="displayDateTime">
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;Drive Monitoring Control
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
&nbsp;&nbsp;&nbsp; &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

<a href="#"><i class="fa fa-fw fa-user"></i></a>User/Role</h5>
```

(Javascript code)

```
<script type="text/javascript">

  var today = new Date();

  var day = today.getDay();

  var daylist = ["Sun","Mon","Tue","Wed","Thu","Fri","Sat"];

  var date = today.getFullYear()+'-'+(today.getMonth()+1)+'-'+today.getDate();

  var time = today.getHours() + ":" + today.getMinutes() + ":" + today.getSeconds();

  var dateTime = date+' '+time;

  document.getElementById("displayDateTime").innerHTML =  ' ' + daylist[day] + ' ' + dateTime + ' ' + '+ GMT 0530
(Indian Standard Time)';

 </script>
```

- Menu Bar:



(HTML code: Inside <body> tag)

```
<div class="w3-bar w3-indigo">
 <button style="text-shadow:1px 1px 0 #444"; class="w3-bar-item w3-button w3-round" onclick="opentab('Home')">Home</button>
 <button style="text-shadow:1px 1px 0 #444"; class="w3-bar-item w3-button w3-round" onclick="opentab('Masters')">Masters</button>
 <button style="text-shadow:1px 1px 0 #444"; class="w3-bar-item w3-button w3-round" onclick="opentab('Provision Drives')">Provision Drives</button>
</div>
```

(Javascript code)

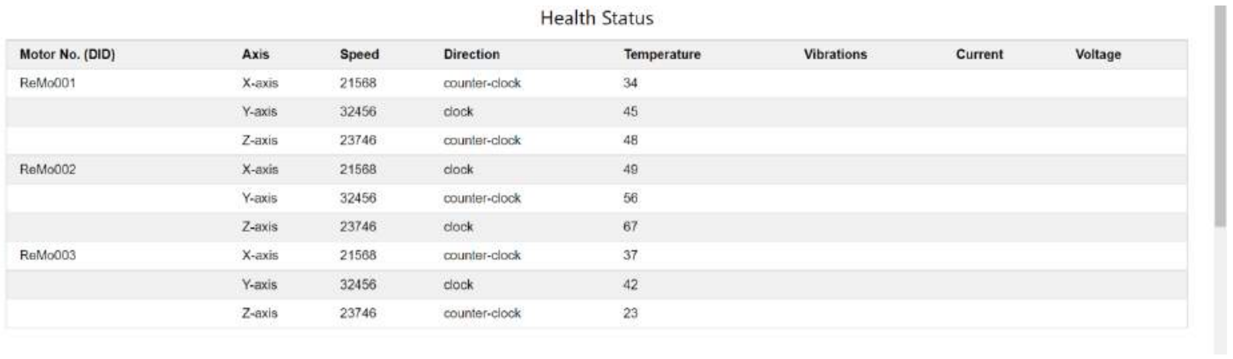
```
<script>
function opentab(tabName) {
  var i;
  var x = document.getElementsByClassName("tab");
  for (i = 0; i < x.length; i++) {
    x[i].style.display = "none";
  }
  document.getElementById(tabName).style.display = "block";
}
</script>
```

- Client Area:

Home Page tab:

(HTML code: Inside <body> tag)

```
<div id="Home" class="w3-container tab">
<div class="w3-container">
```



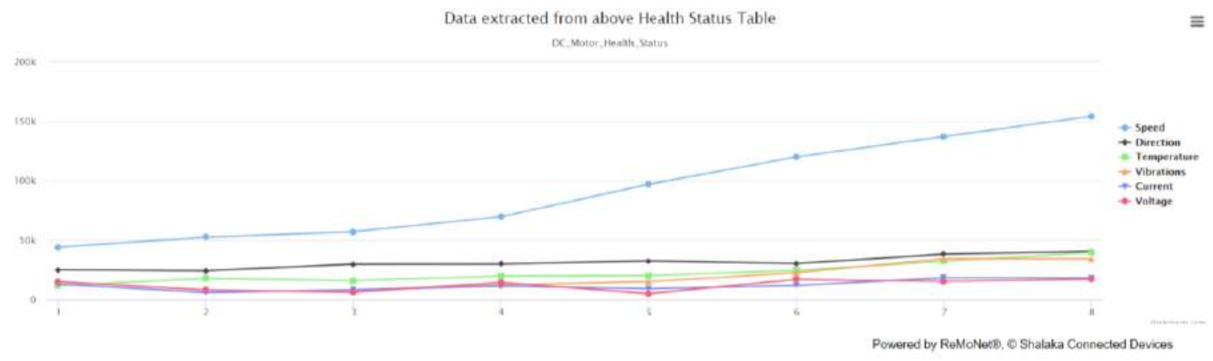
Health Status

Motor No. (DID)	Axis	Speed	Direction	Temperature	Vibrations	Current	Voltage
ReMo001	X-axis	21568	counter-clock	34			
	Y-axis	32456	clock	45			
	Z-axis	23746	counter-clock	48			
ReMo002	X-axis	21568	clock	49			
	Y-axis	32456	counter-clock	56			
	Z-axis	23746	clock	67			
ReMo003	X-axis	21568	counter-clock	37			
	Y-axis	32456	clock	42			
	Z-axis	23746	counter-clock	23			

```
<!—Tabular Health Status -->

<br><h3  style="text-align:center"> Health Status </h3>

 <table class="w3-table w3-striped w3-bordered w3-border w3-hoverable">

  <thead><tr class="w3-light-grey">

    <th>Motor No. (DID)</th>

    <th>Axis</th>

    <th>Speed</th>

    <th>Direction</th>

    <th>Temperature</th>

    <th>Vibrations</th>

    <th>Current</th>

     <th>Voltage</th>

   </tr></thead>

<--ReMo001 details -->      (Refer from table 1)

<--ReMo002 details -->      (Refer from table 1)

<--ReMo003 details -->      (Refer from table 1)

</table>

</div><br>
```

TABLE 1:

<--ReMo001 details -->	<--ReMo002 details -->	<--ReMo003 details -->
<tr>	<tr>	<tr>
<td>ReMo001</td>	<td>ReMo002</td>	<td>ReMo003</td>
<td>X-axis</td>	<td>X-axis</td>	<td>X-axis</td>
<td>21568</td>	<td>21568</td>	<td>21568</td>
<td>counter-clock</td>	<td>clock</td>	<td>counter-clock</td>
<td>34</td>	<td>49</td>	<td>37</td>
<td></td>	<td></td>	<td></td>
<td></td>	<td></td>	<td></td>
<td></td>	<td></td>	<td></td>
</tr> <tr>	</tr> <tr>	</tr> <tr>
<td></td>	<td></td>	<td></td>
<td>Y-axis</td>	<td>Y-axis</td>	<td>Y-axis</td>
<td>32456</td>	<td>32456</td>	<td>32456</td>
<td>clock</td>	<td>counter-clock</td>	<td>clock</td>
<td>45</td>	<td>56</td>	<td>42</td>
<td></td>	<td></td>	<td></td>
<td></td>	<td></td>	<td></td>
<td></td>	<td></td>	<td></td>
</tr> <tr>	</tr> <tr>	</tr> <tr>
<td></td>	<td></td>	<td></td>
<td>Z-axis</td>	<td>Z-axis</td>	<td>Z-axis</td>
<td>23746</td>	<td>23746</td>	<td>23746</td>
<td>counter-clock</td>	<td>clock</td>	<td>counter-clock</td>
<td>48</td>	<td>67</td>	<td>23</td>
<td></td>	<td></td>	<td></td>
<td></td>	<td></td>	<td></td>
<td></td>	<td></td>	<td></td>
</tr>	</tr>	</tr>



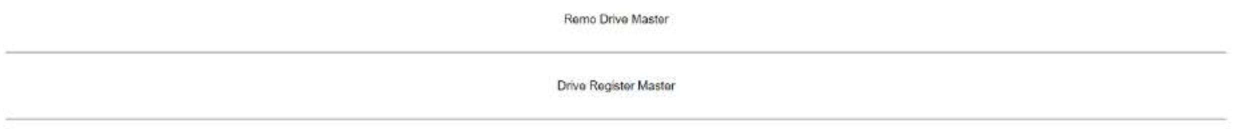
<!—Graphical Health Status -->

(Javascript code)

```
<script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
<script src="https://code.highcharts.com/highcharts.src.js"></script>
<script src="https://code.highcharts.com/modules/exporting.js"></script>
<script type="text/javascript">

  $(function() {
  Highcharts.chart('container', {
    title: {
      text: 'Data extracted from above Health Status Table'
    },
    subtitle: {
      text: 'DC_Motor_Health_Status'
    },
    yAxis: {
      title: {
        text: ''
      }
    },
    legend: {
      layout:'vertical',
      align: 'right',
      verticalAlign:'middle'
    },
    plotOptions: {
    series: {
      label: {
        connectorAllowed: false
      },
      pointStart: 001
     }
   },
   series: [{
    name: 'Speed',
    data: [43934, 52503, 57177, 69658, 97031, 119931, 137133, 154175]
    }, {
    name: 'Direction',
    data: [24916, 24064, 29742, 29851, 32490, 30282, 38121, 40434]
    }, {
    name: 'Temperature',
    data: [11744, 17722, 16005, 19771, 20185, 24377, 32147, 39387]
    }, {
    name: 'Vibrations',
    data: [null, null, 7988, 12169, 15112, 22452, 34400, 34227]
    }, {
    name: 'Current',
    data: [12908, 5948, 8105, 11248, 8989, 11816, 18274, 18111]
    }, {
    name: 'Voltage',
    data: [14908, 7948, 6105, 14248, 4989, 16816, 15274, 17111]
    }]
  });
  });
  </script>
        <div id="container" style="width: 100%; height: 400px;">
  </div><br><br>
<div>
```

Masters Page tab:

Picture 1



(HTML code: Inside <body> tag)

```
<div id="Masters" class="w3-container tab" style="display:none">
<br>
<!—Remo Drive Master accordion-->
<div class="w3-container">
<button onclick="myFunction('Demo1')" class="w3-btn w3-block w3-white w3-centre-align">Remo Drive
Master</button>
<div id="Demo1" class="w3-container w3-hide">
(Tabular Code 1 (code of this table noted after picture 2))
</div>
</div>
```

(Javascript code)

```
<script>
function myFunction(id) {
  var x = document.getElementById(id);
  if (x.className.indexOf("w3-show") == -1) {
    x.className += " w3-show";
  } else {
    x.className = x.className.replace(" w3-show", "");
  }}</script>
```

(HTML Code)

```
<hr class="solid">
<!—Remo Drive Master accordion END-->

<!—Drive Register Master accordion-->
<div class="w3-container" style="overflow-x:auto;">
<button onclick="myFunction('Demo2')" class="w3-btn w3-block w3-white w3-centre-align">Drive
Register Master</button>
<div id="Demo2" class="w3-container w3-hide">
(Tabular Code 2 (code of this table noted after picture 3))
</div>
</div>
```

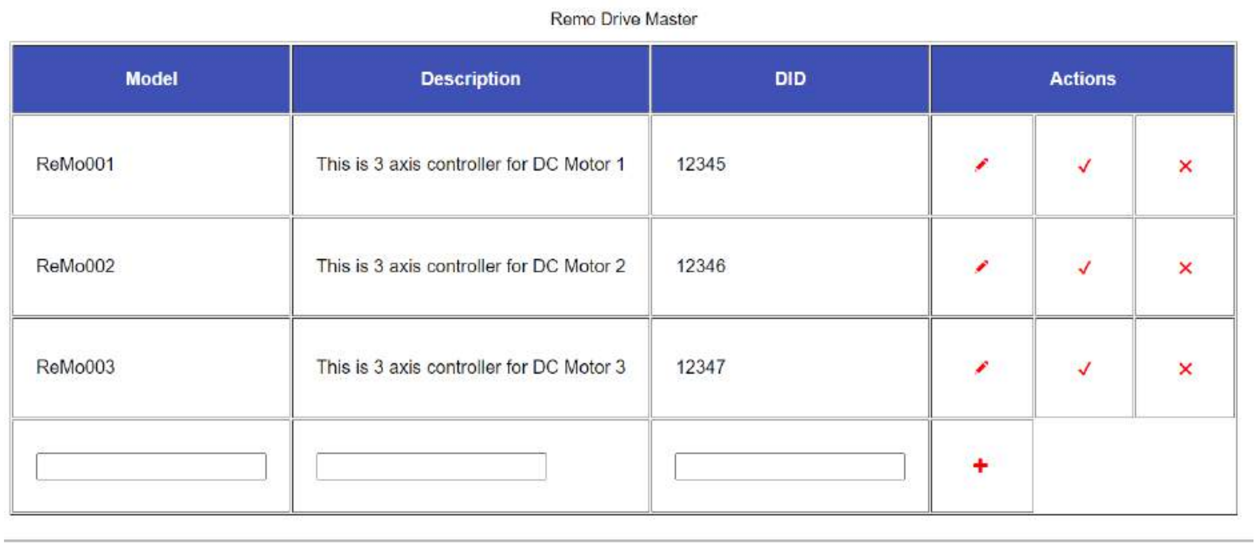
(Javascript Code)

```
<script>
function myFunction(id) {
  var x = document.getElementById(id);
  if (x.className.indexOf("w3-show") == -1) {
    x.className += " w3-show";
  } else {
    x.className = x.className.replace(" w3-show", "");
  }
}
</script>
```

(HTML code)

```
<hr class="solid">
</div>
<!—Drive Register Master accordion END-->
```

Picture 2



Remo Drive Master

Model	Description	DID	Actions		
ReMo001	This is 3 axis controller for DC Motor 1	12345	✎	✓	✕
ReMo002	This is 3 axis controller for DC Motor 2	12346	✎	✓	✕
ReMo003	This is 3 axis controller for DC Motor 3	12347	✎	✓	✕
			+		

Tabular Code 1:

(HTML code)

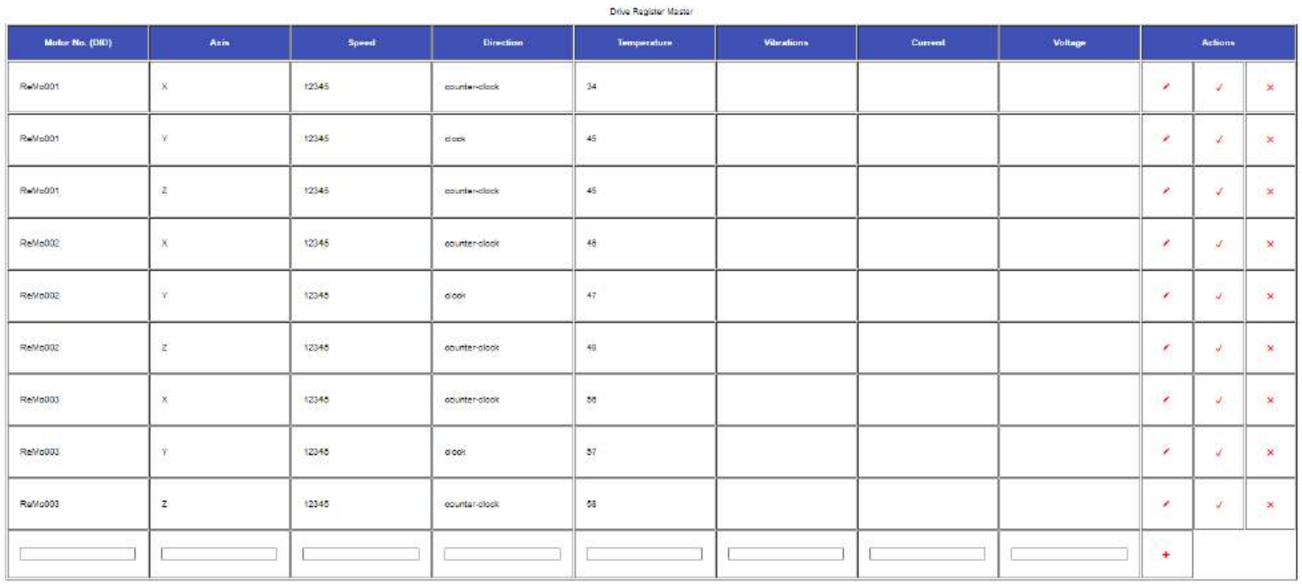
```
<div id="wrapper">
<table align='center' cellspacing=2 cellpadding=20 id="data_table" border=1 class="table_wrapper">
<tr bgcolor="#3f51b5">
<th style="color: white">Model</th>
<th style="color: white">Description</th>
<th style="color: white">DID</th>
<th style="color: white" colspan="3">Actions</th>
</tr>

<tr id="row1">
<td id="model_row1">ReMo001</td>
<td id="description_row1">This is 3 axis controller for DC Motor 1</td>
<td id="DID_row1">12345</td>
<td><button class="btnp" id="edit_button1" class="edit"
onclick="edit_row1('1')">&#x270E;</button></td>
<td><button class="btn" id="save_button1" class="save"
onclick="save_row1('1')">&#10004;</button></td>
<td><button class="btn" class="delete" onclick="delete_row('1')">&#10006;</button></td>
</tr>

<tr id="row2">
<td id="model_row2">ReMo002</td>
<td id="description_row2">This is 3 axis controller for DC Motor 2</td>
<td id="DID_row2">12346</td>
<td><button class="btnp" id="edit_button2" class="edit" onclick="edit_row1('2')">&#x270E;</button>
</td>
<td><button class="btn" id="save_button2" class="save" onclick="save_row1('2')">&#10004;</button>
```

```
</td>
<td><button class="btn" class="delete" onclick="delete_row('2')">&#10006;</button></td>
</tr>

<tr id="row3">
<td id="model_row3">ReMo003</td>
<td id="description_row3">This is 3 axis controller for DC Motor 3</td>
<td id="DID_row3">12347</td>
<td><button class="btnp" id="edit_button3" class="edit" onclick="edit_row1('3')">&#x270E;</button>
</td>
<td><button class="btn" id="save_button3" class="save" onclick="save_row1('3')">&#10004;</button>
</td>
<td><button class="btn" class="delete" onclick="delete_row('3')">&#10006;</button></td>
</tr>

<tr>
<td><input type="text" id="new_model"></td>
<td><input type="text" id="new_description"></td>
<td><input type="text" id="new_DID"></td>
<td><button class="btn" class="add" onclick="add_row1();"><i class="fa fa-plus"></i></button></td>
</tr>
</table>
</div>
```

(Javascript code)

```
<script>
function edit_row1(no)
{
 document.getElementById("edit_button"+no).style.display="none";
 document.getElementById("save_button"+no).style.display="block";

 var model=document.getElementById("model_row"+no);
 var description=document.getElementById("description_row"+no);
 var DID=document.getElementById("DID_row"+no);

 var model_data=model.innerHTML;
 var description_data=description.innerHTML;
 var DID_data=DID.innerHTML;

 model.innerHTML="<input type='text' id='model_text"+no+"' value='"+model_data+"'>";
 description.innerHTML="<input type='text' id='description_text"+no+"' value='"+description_data+"'>";
 DID.innerHTML="<input type='text' id='DID_text"+no+"' value='"+DID_data+"'>";
}

function save_row1(no)
{
 var model_val=document.getElementById("model_text"+no).value;
 var description_val=document.getElementById("description_text"+no).value;
 var DID_val=document.getElementById("DID_text"+no).value;
 document.getElementById("model_row"+no).innerHTML=model_val;
 document.getElementById("description_row"+no).innerHTML=description_val;
 document.getElementById("DID_row"+no).innerHTML=DID_val;

 document.getElementById("edit_button"+no).style.display="block";
 document.getElementById("save_button"+no).style.display="block";
}

function delete_row(no)
{
 document.getElementById("row"+no+"").outerHTML="";
}
function add_row1()
{
 var new_model=document.getElementById("new_model").value;
 var new_description=document.getElementById("new_description").value;
 var new_DID=document.getElementById("new_DID").value;

 var table=document.getElementById("data_table");
 var table_len=(table.rows.length)-1;
 var row = table.insertRow(table_len).outerHTML="<tr id='row"+table_len+"'><td
id='model_row"+table_len+"'>"+new_model+"</td><td
id='description_row"+table_len+"'>"+new_description+"</td><td
id='DID_row"+table_len+"'>"+new_DID+"</td><td><input type='button' class='btnp'
id='edit_button"+table_len+"' value='&#x270E;' class='edit' onclick='edit_row1("+table_len+")'></td>
<td><input type='button' class='btn' id='save_button"+table_len+"' value='&#10004;' class='save'
onclick='save_row1("+table_len+")'></td> <td><input type='button' class='btn' value=' &#10006;'
class='delete' onclick='delete_row("+table_len+")'></td></tr>";

 document.getElementById("new_model").value="";
 document.getElementById("new_description").value="";
 document.getElementById("new_DID").value="";

}
</script>
```

Picture 3



Drive Register Master

Motor No. (DID)	Axis	Speed	Direction	Temperature	Vibrations	Current	Voltage	Actions		
ReMo001	X	12345	counter-clock	34				✎	✓	✕
ReMo001	Y	12345	clock	45				✎	✓	✕
ReMo001	Z	12345	counter-clock	45				✎	✓	✕
ReMo002	X	12345	counter-clock	46				✎	✓	✕
ReMo002	Y	12345	clock	47				✎	✓	✕
ReMo002	Z	12345	counter-clock	48				✎	✓	✕
ReMo003	X	12345	counter-clock	56				✎	✓	✕
ReMo003	Y	12345	clock	57				✎	✓	✕
ReMo003	Z	12345	counter-clock	58				✎	✓	✕
								+		

Tabular Code 2:

(HTML code)

```
<div id="wrapper">
<table align='center' cellspacing=2 cellpadding=20 id="data_tablen" border=1 class="table_wrapper">
<tr bgcolor="#3f51b5">
<th style="color: white">Motor No. (DID)</th>
<th style="color: white">Axis</th>
<th style="color: white">Speed</th>
<th style="color: white">Direction</th>
<th style="color: white">Temperature</th>
<th style="color: white">Vibrations</th>
<th style="color: white">Current</th>
<th style="color: white">Voltage</th>
<th style="color: white" colspan="3">Actions</th>
</tr>

<tr id="row1">
<td id="motor_row1">ReMo001</td>
<td id="axis_row1">X</td>
<td id="speed_row1">12345</td>
<td id="direction_row1">counter-clock</td>
<td id="temperature_row1">34</td>
<td id="vibrations_row1"></td>
<td id="current_row1"></td>
<td id="voltage_row1"></td>
<td><button class="btnp" id="edit_button1" class="edit" onclick="edit_row('1')">&#x270E;</button>
</td>
<td><button class="btn" id="save_button1" class="save" onclick="save_row('1')">&#10004;</button>
```

```
</td>
<td><button class="btn" class="delete" onclick="delete_row('1')">&#10006;</button></td>
</tr>

<tr id="row2">
<td id="motor_row2">ReMo001</td>
<td id="axis_row2">Y</td>
<td id="speed_row2">12345</td>
<td id="direction_row2">clock</td>
<td id="temperature_row2">45</td>
<td id="vibrations_row2"></td>
<td id="current_row2"></td>
<td id="voltage_row2"></td>
<td><button class="btnp" id="edit_button2" class="edit" onclick="edit_row('2')">&#x270E;</button>
</td>
<td><button class="btn" id="save_button2" class="save" onclick="save_row('2')">&#10004;</button>
</td>
<td><button class="btn" class="delete" onclick="delete_row('2')">&#10006;</button></td>
</tr>

<tr id="row3">
<td id="motor_row3">ReMo001</td>
<td id="axis_row3">Z</td>
<td id="speed_row3">12345</td>
<td id="direction_row3">counter-clock</td>
<td id="temperature_row3">45</td>
<td id="vibrations_row3"></td>
<td id="current_row3"></td>
<td id="voltage_row3"></td>
<td><button class="btnp" id="edit_button3" class="edit" onclick="edit_row('3')">&#x270E;</button>
</td>
<td><button class="btn" id="save_button3" class="save" onclick="save_row('3')">&#10004;</button>
</td>
<td><button class="btn" class="delete" onclick="delete_row('3')">&#10006;</button></td>
</tr>

<tr id="row4">
<td id="motor_row4">ReMo002</td>
<td id="axis_row4">X</td>
<td id="speed_row4">12345</td>
<td id="direction_row4">counter-clock</td>
<td id="temperature_row4">46</td>
<td id="vibrations_row4"></td>
<td id="current_row4"></td>
<td id="voltage_row4"></td>
<td><button class="btnp" id="edit_button4" class="edit" onclick="edit_row('4')">&#x270E;</button>
</td>
<td><button class="btn" id="save_button4" class="save" onclick="save_row('4')">&#10004;</button>
</td>
```

```
<td><button class="btn" class="delete" onclick="delete_row('4')">&#10006;</button></td>
</tr>

<tr id="row5">
<td id="motor_row5">ReMo002</td>
<td id="axis_row5">Y</td>
<td id="speed_row5">12345</td>
<td id="direction_row5">clock</td>
<td id="temperature_row5">47</td>
<td id="vibrations_row5"></td>
<td id="current_row5"></td>
<td id="voltage_row5"></td>
<td><button class="btnp" id="edit_button5" class="edit" onclick="edit_row('5')">&#x270E;</button>
</td>
<td><button class="btn" id="save_button5" class="save" onclick="save_row('5')">&#10004;</button>
</td>
<td><button class="btn" class="delete" onclick="delete_row('5')">&#10006;</button></td>
</tr>

<tr id="row6">
<td id="motor_row6">ReMo002</td>
<td id="axis_row6">Z</td>
<td id="speed_row6">12345</td>
<td id="direction_row6">counter-clock</td>
<td id="temperature_row6">48</td>
<td id="vibrations_row6"></td>
<td id="current_row6"></td>
<td id="voltage_row6"></td>
<td><button class="btnp" id="edit_button6" class="edit" onclick="edit_row('6')">&#x270E;</button>
</td>
<td><button class="btn" id="save_button6" class="save" onclick="save_row('6')">&#10004;</button>
</td>
<td><button class="btn" class="delete" onclick="delete_row('6')">&#10006;</button></td>
</tr>

<tr id="row7">
<td id="motor_row7">ReMo003</td>
<td id="axis_row7">X</td>
<td id="speed_row7">12345</td>
<td id="direction_row7">counter-clock</td>
<td id="temperature_row7">56</td>
<td id="vibrations_row7"></td>
<td id="current_row7"></td>
<td id="voltage_row7"></td>
<td><button class="btnp" id="edit_button7" class="edit" onclick="edit_row('7')">&#x270E;</button>
</td>
<td><button class="btn" id="save_button7" class="save" onclick="save_row('7')">&#10004;</button>
</td>
<td><button class="btn" class="delete" onclick="delete_row('7')">&#10006;</button></td>
```

```
</tr>

<tr id="row8">
<td id="motor_row8">ReMo003</td>
<td id="axis_row8">Y</td>
<td id="speed_row8">12345</td>
<td id="direction_row8">clock</td>
<td id="temperature_row8">57</td>
<td id="vibrations_row8"></td>
<td id="current_row8"></td>
<td id="voltage_row8"></td>
<td><button class="btnp" id="edit_button8" class="edit" onclick="edit_row('8')">&#x270E;</button>
</td>
<td><button class="btn" id="save_button8" class="save" onclick="save_row('8')">&#10004;</button>
</td>
<td><button class="btn" class="delete" onclick="delete_row('8')">&#10006;</button></td>
</tr>

<tr id="row9">
<td id="motor_row9">ReMo003</td>
<td id="axis_row9">Z</td>
<td id="speed_row9">12345</td>
<td id="direction_row9">counter-clock</td>
<td id="temperature_row9">58</td>
<td id="vibrations_row9"></td>
<td id="current_row9"></td>
<td id="voltage_row9"></td>
<td><button class="btnp" id="edit_button9" class="edit" onclick="edit_row('9')">&#x270E;</button>
</td>
<td><button class="btn" id="save_button9" class="save" onclick="save_row('9')">&#10004;</button>
</td>
<td><button class="btn" class="delete" onclick="delete_row('9')">&#10006;</button></td>
</tr>

<tr>
<td><input type="text" id="new_motor"></td>
<td><input type="text" id="new_axis"></td>
<td><input type="text" id="new_speed"></td>
<td><input type="text" id="new_direction"></td>
<td><input type="text" id="new_temperature"></td>
<td><input type="text" id="new_vibrations"></td>
<td><input type="text" id="new_current"></td>
<td><input type="text" id="new_voltage"></td>

<td><button class="btn" class="add" onclick="add_row();"><i class="fa fa-plus"></i></button></td>
</tr>

</table>
</div>
```

(Javascript code)

```
function edit_row(no)
{
 document.getElementById("edit_button"+no).style.display="none";
 document.getElementById("save_button"+no).style.display="block";

 var motor=document.getElementById("motor_row"+no);
 var axis=document.getElementById("axis_row"+no);
 var speed=document.getElementById("speed_row"+no);
 var direction=document.getElementById("direction_row"+no);
 var temperature=document.getElementById("temperature_row"+no);
 var vibrations=document.getElementById("vibrations_row"+no);
 var current=document.getElementById("current_row"+no);
 var voltage=document.getElementById("voltage_row"+no);

 var motor_data=motor.innerHTML;
 var axis_data=axis.innerHTML;
 var speed_data=speed.innerHTML;
 var direction_data=direction.innerHTML;
 var temperature_data=temperature.innerHTML;
 var vibrations_data=vibrations.innerHTML;
 var current_data=current.innerHTML;
 var voltage_data=voltage.innerHTML;

 motor.innerHTML="<input type='text' id='motor_text"+no+"' value='"+motor_data+"'>";
 axis.innerHTML="<input type='text' id='axis_text"+no+"' value='"+axis_data+"'>";
 speed.innerHTML="<input type='text' id='speed_text"+no+"' value='"+speed_data+"'>";
 direction.innerHTML="<input type='text' id='direction_text"+no+"' value='"+direction_data+"'>";
 temperature.innerHTML="<input                    type='text'                    id='temperature_text"+no+"'
value='"+temperature_data+"'>";
 vibrations.innerHTML="<input type='text' id='vibrations_text"+no+"' value='"+vibrations_data+"'>";
 current.innerHTML="<input type='text' id='current_text"+no+"' value='"+current_data+"'>";
 voltage.innerHTML="<input type='text' id='voltage_text"+no+"' value='"+voltage_data+"'>";

}

function save_row(no)
{
 var motor_val=document.getElementById("motor_text"+no).value;
 var axis_val=document.getElementById("axis_text"+no).value;
 var speed_val=document.getElementById("speed_text"+no).value;
 var direction_val=document.getElementById("direction_text"+no).value;
 var temperature_val=document.getElementById("temperature_text"+no).value;
 var vibrations_val=document.getElementById("vibrations_text"+no).value;
 var current_val=document.getElementById("current_text"+no).value;
 var voltage_val=document.getElementById("voltage_text"+no).value;

 document.getElementById("motor_row"+no).innerHTML=motor_val;
 document.getElementById("axis_row"+no).innerHTML=axis_val;
 document.getElementById("speed_row"+no).innerHTML=speed_val;
 document.getElementById("direction_row"+no).innerHTML=direction_val;
 document.getElementById("temperature_row"+no).innerHTML=temperature_val;
 document.getElementById("vibrations_row"+no).innerHTML=vibrations_val;
 document.getElementById("current_row"+no).innerHTML=current_val;
 document.getElementById("voltage_row"+no).innerHTML=voltage_val;

 document.getElementById("edit_button"+no).style.display="block";
 document.getElementById("save_button"+no).style.display="block";
}

function delete_row(no)
{
 document.getElementById("row"+no+"").outerHTML="";
}

function add_row()
{
 var new_motor=document.getElementById("new_motor").value;
 var new_axis=document.getElementById("new_axis").value;
 var new_speed=document.getElementById("new_speed").value;
 var new_direction=document.getElementById("new_direction").value;
 var new_temperature=document.getElementById("new_temperature").value;
 var new_vibrations=document.getElementById("new_vibrations").value;
 var new_current=document.getElementById("new_current").value;
 var new_voltage=document.getElementById("new_voltage").value;

 var tablen=document.getElementById("data_tablen");
 var tablen_len=(tablen.rows.length)-1;
 var row = tablen.insertRow(tablen_len).outerHTML="<tr id='row"+tablen_len+"'><td
id='motor_row"+tablen_len+"'>"+new_motor+"</td><td
id='axis_row"+tablen_len+"'>"+new_axis+"</td><td
id='speed_row"+tablen_len+"'>"+new_speed+"</td><td
id='direction_row"+tablen_len+"'>"+new_direction+"</td><td
id='temperature_row"+tablen_len+"'>"+new_temperature+"</td><td
id='vibrations_row"+tablen_len+"'>"+new_vibrations+"</td><td
id='current_row"+tablen_len+"'>"+new_current+"</td><td
id='voltage_row"+tablen_len+"'>"+new_voltage+"</td><td><input type='button' class='btnp'
id='edit_button"+tablen_len+"' value='&#x270E;' class='edit' onclick='edit_row("+tablen_len+")'></td>
<td><input type='button' class='btn' id='save_button"+tablen_len+"' value='&#10004;' class='save'
onclick='save_row("+tablen_len+")'></td> <td><input type='button' class='btn' value=' &#10006;'
class='delete' onclick='delete_row("+tablen_len+")'></td></tr>";

 document.getElementById("new_motor").value="";
 document.getElementById("new_axis").value="";
 document.getElementById("new_speed").value="";
 document.getElementById("new_direction").value="";
 document.getElementById("new_temperature").value="";
 document.getElementById("new_vibrations").value="";
 document.getElementById("new_current").value="";
 document.getElementById("new_voltage").value="";
}
</script><br><br>
```

Provision Drives Page tab:

(HTML code: Inside <body> tag)

```
<div id="Provision Drives" class="w3-container tab" style="display:none">
 <h2>Provision Drives</h2>
</div>
```

- Footer:

Powered by ReMoNet®, © Shalaka Connected Devices

(CSS code)

```
/* Footer */
.footer {
 position: fixed;
 left: 0;
 bottom: 0;
 width: 100%;
background-color: white;
color: black;
 text-align: right;
}
```

(HTML code: Inside <body> tag)

```
<div class="footer">
 <p>Powered by ReMoNet®, © Shalaka Connected Devices &nbsp;&nbsp;&nbsp;&nbsp;
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</p>
</div>
```

Appendix 4: About ReMoNet

ReMoNet assists in safety, security, comfort and good health of occupants and capital goods of the organizations. It can monitor climate, hazardous conditions, assets and many other parameters. It can take a step ahead to monitor and raise alerts, take preventive measures and store data for future analysis. It is a network of devices that communicate with one another to create a mesh of devices for the benefit of the organization.

ReMo remote monitoring system software is a versatile embedded solution that facilitates offices, data centers, industries and homes in monitoring critical parameters such as temperature, humidity, utility/backup power, smoke/fire, door safety (open/close) and alert via SMS/E-mail.

Host side user friendly software facilitates in setting up and configuring the ReMo device. Absence of local display and keypad ensures secure functioning. However, an optional display/keypad console can be provided where required.

References ...

- ✓ https://searchcloudcomputing.techtarget.com/definition/cloud-application
- ✓ https://www.i-scoop.eu/internet-of-things-guide/industrial-internet-things-iiot-saving-costs-innovation/industrial-internet-things-iiot/
- ✓ https://www.figma.com/resources/learn-design/
- ✓ https://www.freepik.com/home
- ✓ https://www.w3schools.com/
- ✓ https://media-exp3.licdn.com/dms/image/C4E1BAQFkNucHaTSz5A/company-background_10000/0/1519798297732?e=2159024400&v=beta&t=EOU0hy2itSCu7xZ9rFeSkWpYOsabq3R9ZHoxabWc9KM
- ✓ https://www.digikey.in/-/media/Images/Design%20Service%20Providers/S/Shalaka%20Logo/Shalaka%20Logo-200.jpg?la=en-IN&ts=cf2d1bf3-a6b1-43c2-ad7c-6f55c5a4b4f4
- ✓ http://shalaka.com/v2/
- ✓ https://dev.mysql.com/doc/refman/8.0/en/creating-tables.html

Glossary …

MQTT **:** The Message Queuing Telemetry Transport is a lightweight, publish-subscribe network protocol that transports messages between devices.

DC motor controller **:** DC motor controller is any device that can manipulate the position, speed, or torque of a DC-powered motor.

UI (User Interface) **:** User interfaces are the access points where users interact with designs. They come in three formats: Graphical user interfaces (GUIs)—Users interact with visual representations on digital control panels. A computer's desktop is a GUI. Voice-controlled interfaces (VUIs)—Users interact with these through their voices.

Green IoT **:** Green IoT is defined as the energy efficient ways in IoT either to reduce the green-house effect caused by existing applications or to eradicate the same in IoT itself.

API **:** API is the acronym for Application Programming Interface, which is a software intermediary that allows two applications to talk to each other.

ReMoNet**:** Remote Monitoring Net (More details about it in Appendix 4)

DID **:** DID stands for Drive Identification (Drive ID). It's 5 - digit device unique ID

Responsiveness **:** Responsive web design is the approach that suggests that design and development should respond to the user's behavior and environment based on screen size, platform and orientation.

Accordion menu **:** An accordion menu is **a vertically stacked list of headers** that can be clicked to reveal or hide content associated with them. It is one of many ways you can expose content to users in a progressive manner.

THANK YOU