Heaven's light is our guide

Rajshahi University of Engineering &Technology, Bangladesh

Department of Mechatronics Engineering

A Project & Thesis Report on

Environment Condition Monitoring and Long Distance Actuator Control using Wireless Sensor Network (WSN)

Supervised by

Md. Manirul Islam
Assistant Professor
Department of Mechatronics Engineering
Rajshahi University of Engineering & Technology

Submitted by

Md. Asif Bin Karim
Roll: 148004
Saquib Shahriar
Roll: 148007

August-2019

Heaven's light is our guide

Rajshahi University of Engineering &Technology, Bangladesh
Department of Mechatronics Engineering

Certificate

This is to certify that the thesis entitled "**Environment Condition Monitoring and Long Distance Actuator Control using Wireless Sensor Network (WSN)**" has been carried out by **Md. Asif Bin Karim, Roll: 148004** and **Saquib Shahriar, Roll: 148007** under my supervision in the Department of Mechatronics Engineering at Rajshahi University of Engineering & Technology, Rajshahi-6204, Bangladesh.

Signature of the Supervisor

……...

Md. Manirul Islam
Assistant Professor
Department of Mechatronics Engineering
Rajshahi University of Engineering & Technology

Signature of the student

……

Md. Asif Bin Karim
Roll No.: 148004

……

Saquib Shahriar
Roll No.: 148007

Countersigned

……...

Dr. Sajal Kumar Das
Head
Department of Mechatronics Engineering
Rajshahi University of Engineering & Technology

Signature of the External

……...

Md. Firoj Ali
Assistant Professor
Department of Mechatronics Engineering
Rajshahi University of Engineering & Technology

Table of Contents

Index of Figures

Index of Tables

Declaration

We, Md. Asif Bin Karim and Saquib Shahriar, hereby declare that we are the sole author of the thesis titled " Environment Condition Monitoring and Long Distance Actuator Control using Wireless Sensor Network (WSN)". This is written and submitted by us to the Department of Mechatronics Engineering, Rajshahi University of Engineering & Technology, in partial fulfillment of the requirements for acquiring the degree of Bachelor of Science in Mechatronics Engineering. This work was done under the supervision of Md. Manirul Islam, Assistant Professor, Department of Mechatronics Engineering, Rajshahi University of Engineering & Technology.

We certify that, to the best of our knowledge, our thesis does not infringe upon anyone's copyright nor violate proprietary rights, and any ideas, techniques, quotations or any other material from the work of other people included in our thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices.

We further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this university or any other institutes or universities.

Authors

Acknowledgment

All the praises and the supreme thanks belong to Allah, “Lord of All the Worlds, Most Beneficent, and Ever-Merciful”.

It is needless to express the humble submission and indebtedness in language to Dr. Md. Sajal Kumar Das, Head, Department of Mechatronics Engineering. His endless patience, scholarly guidance, continual encouragement, cordial supervision, constructive criticism and valuable advice at all stage have made it possible to complete this project. Authors will never forget him for being kind enough to spend his valuable time in inspecting and helping them by all means in every phase of their work. This enables them to overcome the difficulties at different stages.

All the honorable teachers from the Department of Mechatronics Engineering, RUET are also remembered with great gratitude for their cordial well wishes towards the authors.

Thanks, are also due to all the staff members of Department of Mechatronics Engineering for their help and co-operation.

Authors are thankful to their classmate and all who helped them directly and indirectly at many stages to complete the project.

Finally, authors must express their very profound gratitude to their parents for providing them with unfailing support and continuous encouragement throughout their years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

Authors

Md. Asif Bin Karim

Roll No: 148004

Saquib Shahriar

Roll No: 148007

RUET, Rajshahi

August 2019

Abstract

In this project environment monitoring systems is implemented by using sensors and then sensors data are sent to a My SQL server using Wi-Fi. For enchanting data, DHT-11 (Temperature and Humidity sensor) and MQ-6 (LPG gas sensor) are being used. The basic objective of this research is to monitor and to develop a real-time monitoring of humidity and temperature, as well as the availability of gas using the very available DHT-11 sensor, MQ-6 sensor, and ESP-8266 NodeMCU module and then observe the data from a database. We can control the actuator from the server depending on the sensor value. Although great leap has been made in the control area, the precision motion control is challenging the control engineering to a greater extent. The control engineer needs to design a suitable controller which will effectively achieve the desired system characteristics, such as high precision, high speed requirements in precision motion control. here are two control schemes which have been proposed. In this research, both feedback and feedforward methods of control have been applied. This paper also makes a compact distinction between conventional and the local IP based observing system of an environment. The sensor's data are saved in a database by which we can monitor the sensor’s data without any access to the internet. In this research, the Arduino based ESP-8266 based NodeMCU was used. The various data attainment system of Arduino or Raspberry-pi is mother controller but using NodeMCU gives the benefit of using an Arduino along with a 2.4 GHz Wi-Fi module. As this was a demo project and needed far more inquiry in the real practice so, breadboards and jumper wires were used to test the project.

Chapter1
Introduction

1.1 Overview of Wireless Sensor Network

Wireless sensor network (WSN) refers to a group of spatially dispersed and dedicated sensors for monitoring and recording the physical conditions of the environment and organizing the collected data at a central location. The propagation technique between the hops of the network can be routing or flooding. In computer science and telecommunications, wireless sensor networks are an active research area with numerous workshops and conferences arranged each year, for example IPSN, SenSys, and EWSN. Each such sensor network node has typically several parts: a radio transceiver with an internal antenna or connection to an external antenna, a microcontroller, an electronic circuit for interfacing with the sensors and an energy source, usually a battery or an embedded form of energy harvesting.

Enabled by recent advances in microelectronic mechanical systems (MEMS) and wireless communication technologies, tiny, cheap, and smart sensors deployed in a physical area and networked through wireless links and the Internet provide unprecedented opportunities for a variety of civilian and military applications, for example, environmental monitoring, battle field surveillance, and industry process control. Distinguished from traditional wireless communication networks, for example, cellular systems and mobile ad hoc networks (MANET), WSNs have unique characteristics, for example, denser level of node deployment, higher unreliability of sensor nodes, and severe energy, computation, and storage constraints, which present many new challenges in the development and application of WSNs. A large amount of research activities has been carried out to explore and solve various design and application issues, and significant advances have been made in the development and deployment of WSNs. It is envisioned that in the near future WSNs will be widely used in various civilian and military fields, and revolutionize the way we live, work, and interact with the physical world.

1.2 Contribution and Motivation

Safety is a less concerned issue in our country. Workers safety gets less priority and several fire accidents occur in the overcrowded rural areas in Bangladesh almost every year. No precautions are taken by the industries or the owners of the buildings as they don't want to pay extra for the safety issues. So, we tried to build a system which can monitor any system, warn people and can take actions if any accident especially the fire accident which would be not only effective but also cheap.

We have chosen Wireless Sensor Network Technology because of its flexibility and easy implementation capability. Moreover, we have attached a solar panel as a secondary power source to the system for the implementation of renewable energy. We believe our system is capable of after-effects of any kind fire accident and can monitor the humidity, temperature of the home, patients rooms or food or chemical industries where these factors should be strictly maintained.

1.3 Characteristics of Wireless Sensor Network

- Due to the large number of sensor nodes, it is usually not possible to build a global addressing scheme for a sensor network because it would introduce a high overhead for the identification maintenance.
- Once deployed, sensor nodes have to autonomously configure themselves into a communication network.
- Sensor nodes are usually deployed in harsh or hostile environments and operate without attendance.
- The number of sensor nodes in a sensor network can be several orders of magnitude higher than that in a MANET.
- Sensor nodes are usually randomly deployed without careful planning and engineering.
- Sensor nodes are highly limited in energy, computation, and storage capacities.
- In most sensor network applications, the data sensed by sensor nodes flow from multiple source sensor nodes to a particular sink, exhibiting a many - to - one traffic pattern.

- In most sensor network applications, sensor nodes are densely deployed in a region of interest and collaborate to accomplish a common sensing task.
- The data sensed by multiple sensor nodes typically have a certain level of correlation or redundancy.

1.4 Network Application

WSNs were originally motivated by military applications, which range from large - scale acoustic surveillance systems for ocean surveillance to small networks of unattended ground sensors for ground target detection. Sensors can be used to detect or monitor a variety of physical parameters or conditions, for example:

• Light

• Sound

• Humility

• Pressure

• Temperature

• Soil composition

• Air or water quality

• Attributes of an object such as size, weight, position, speed, and direction.

They can not only reduce the cost and delay in deployment, but also be applied to any environment, especially those in which conventional wired sensor networks are impossible to be deployed, for example, inhospitable terrains, battlefields, outer space, or deep oceans. However, the availability of low - cost sensors and wireless communication has promised the development of a wide range of applications in both civilian and military fields.

- **Environment Monitoring**

In environmental monitoring, sensors are used to monitor a variety of environmental parameters or conditions. Environmental monitoring is one of the earliest applications of sensor networks.

Sensors can be deployed on the ground or under water to monitor air or water quality. For example, water quality monitoring can be used in the hydrochemistry field. Sensors can be used to monitor biological or chemical hazards in locations, for example, a chemical plant or a battlefield. Sensors can be densely deployed in an intended region to detect natural or non - natural disasters. For example, sensors can be scattered in forests or revivers to detect forest fi res or floods Air or Water Quality Monitoring. from the University of California at Berkeley and the college of the Atlantic in Bar Harbor, conducted an experiment to monitor the habitat of the nesting petrels on Great Duck Land in Maine by deploying 190 wireless sensors, including humidity, pressure, temperature, and radiation.

- **Military Application**

Due to ease of deployment, self - configurability, untended operation, and fault tolerance, sensor networks will play more important roles in future military C3I systems and make future wars more intelligent with less human involvement. Sensors can be mounted on unmanned robotic vehicles, tanks, fighter planes, submarines, missiles, or torpedoes to guide them around obstacles to their targets and lead them to coordinate with one another to accomplish more effective attacks or defenses. Sensor nodes can be deployed around sensitive objects, for example, atomic plants, strategic bridges, oil and gas pipelines, communication centers, and military headquarters, for protection purpose. Sensors can be deployed for remote sensing of nuclear, biological, and chemical weapons, detection of potential terrorist attacks, and reconnaissance. Sensors can be deployed in a battlefield to monitor the presence of forces and vehicles, and track their movements, enabling close surveillance of opposing forces.

- **Health Care Application**

WSNs can be used to monitor and track elders and patients for health care purposes, which can significantly relieve the severe shortage of health care personnel and reduce the health care expenditures in the current health care systems. Wearable sensors can be integrated into a wireless body area network (WBAN) to monitor vital signs, environmental parameters, and geographical locations, and thus allow long - term, noninvasive, and ambulatory monitoring of patients or elderly people with instantaneous alerts to health care personal in case of emergency, immediate reports to users about their current health statuses, and real - time updates of users ’ medical records.

- **Industrial Process Control**

Tiny sensors can be embedded into the regions of a machine that are inaccessible by humans to monitor the condition of the machine and alert for any failure. For example, wireless sensors can be instrumented to production and assembly lines to monitor and control production processes. Chemical plants or oil refiners can use sensors to monitor the condition of their miles of pipelines. In industry, WSNs can be used to monitor manufacturing processes or the condition of manufacturing equipment.

- **Security and Surveillance**

For example, acoustic, video, and other kinds of sensors can be deployed in buildings, airports, subways, and other critical infrastructure, for example, nuclear power plants or communication centers to identify and track intruders, and provide timely alarms and protection from potential attacks.

- **Home Intelligence**

WSNs can be used to provide more convenient and intelligent living environments for human beings. Wireless sensors can be embedded into a home and connected to form an autonomous home network. Wireless sensors can be used to remotely read utility meters in a home, for example, water, gas, or electricity, and then send the readings to a remote center through wireless communication. In addition to the above applications, self - configurable WSNs can be used in many other areas, for example, disaster relief, traffic control, warehouse management, and civil engineering.

1.5 Network Design Objectives

- **Small Node Size:** Reducing node size is one of the primary design objectives of sensor networks. Sensor nodes are usually deployed in a harsh or hostile environment in large numbers. Reducing node size can facilitate node deployment, and also reduce the cost and power consumption of sensor nodes.
- **Low Node Cost:** Reducing node cost is another primary design objective of sensor network. Since sensor nodes are usually deployed in a harsh or hostile environment in large

numbers and cannot be reused, it is important to reduce the cost of sensor nodes so that the cost of the whole network is reduced.

- **Low Power Consumption:** Reducing power consumption is the most important objective in the design of a sensor network. Since sensor nodes are powered by battery and it is often very difficult or even impossible to change or recharge their batteries, it is crucial to reduce the power consumption of sensor nodes so that the lifetime of the sensor nodes, as well as the whole network is prolonged.
- **Self – Configurability:** In sensor networks, sensor nodes are usually deployed in a region of interest without careful planning and engineering. Once deployed, sensor nodes should be able to autonomously organize themselves into a communication network and reconfigure their connectivity in the event of topology changes and node failures.
- **Scalability:** In sensor networks, the number of sensor nodes may be on the order of tens, hundreds, or thousands. Thus, network protocols designed for sensor networks should be scalable to different network sizes.
- **Adaptability:** In sensor networks, a node may fail, join, or move, which would result in changes in node density and network topology. Thus, network protocols designed for sensor networks should be adaptive to such density and topology changes.
- **Reliability:** For many sensor network applications, it is required that data be reliably delivered over noisy, error - prone, and time - varying wireless channels. To meet this requirement, network protocols designed for sensor networks must provide error control and correction mechanisms to ensure reliable data delivery.
- **Fault Tolerance:** Sensor nodes are prone to failures due to harsh deployment environments and unattended operations. Thus, sensor nodes should be fault tolerant and have the abilities of self - testing, self - calibrating, self-repairing, and self - recovering.
- **Security:** In many military applications, sensor nodes are deployed in a hostile environment and thus are vulnerable to adversaries. In such situations, a sensor network should introduce effective security mechanisms to prevent the data information in the network or a sensor node from unauthorized access or malicious attacks.
- **Channel Utilization:** Sensor networks have limited bandwidth resources. Thus, communication protocols designed for sensor networks should efficiently make use of the bandwidth to improve channel utilization.

- **QoS Support:** In sensor networks, different applications may have different quality - of - service (QoS) requirements in terms of delivery latency and packet loss. For example, some applications, for example, fi re monitoring, are delay sensitive and thus require timely data delivery. Some applications, for example, data collection for scientific exploration, are delay tolerant but cannot stand packet loss. Thus, network protocol design should consider the QoS requirements of specific applications.

1.6 Network Design Challenges

- **Limited Hardware Resources:** Sensor nodes have limited processing and storage capacities, and thus can only perform limited computational functionalities. These hardware constraints present many challenges in software development and network protocol design for sensor networks, which must consider not only the energy constraint in sensor nodes, but also the processing and storage capacities of sensor nodes.
- **Massive and Random Deployment:** Most sensor networks consist of a large number of sensor nodes, from hundreds to thousands or even more. Node deployment is usually application dependent, which can be either manual or random. In most applications, sensor nodes can be scattered randomly in an intended area or dropped massively over an inaccessible or hostile region. The sensor nodes must autonomously organize themselves into a communication network before they start to perform a sensing task.
- **Dynamic and Unreliable Environment:** A sensor network usually operates in a dynamic and unreliable environment. On one hand, the topology of a sensor network may change frequently due to node failures, damages, additions, or energy depletion. On the other hand, sensor nodes are linked by a wireless medium, which is noisy, error prone, and time varying. The connectivity of the network may be frequently disrupted because of channel fading or signal attenuation.
- **Diverse Application:** Sensor networks have a wide range of diverse applications. The requirements for different applications may vary significantly. No network protocol can

meet the requirements of all applications. The design of sensor networks is application specific.

- **Limited Energy Capacity:** Sensor nodes are battery powered and thus have very limited energy capacity. This constraint presents many new challenges in the development of hardware and software, and the design of network architectures and protocols for sensor networks. To prolong the operational lifetime of a sensor network, energy efficiency should be considered in every aspect of sensor network design, not only hardware and software, but also network architectures and protocols.

1.7 Wireless Communication Technology

At higher layers, efficient communication protocols have been developed to address various networking issues, for example, medium access control, routing, QoS, and network security. Wireless communication has been extensively studied for conventional wireless networks in the last couple of decades and significant advances have been obtained in various aspects of wireless communication. Today most conventional wireless networks use radio frequency (RF) for communication, including microwave and millimeter wave. However, RF has some limitations, for example, large radiators and low transmission efficiencies, which make RF not the best communication medium for tiny energy - constrained sensor nodes. On the other hand, most communication protocols for conventional wireless networks, for example, cellular systems, wireless local area networks (WLANs), wireless personal area networks (WPANs), and MANETs, do not consider the unique characteristics of sensor networks, in particular, the energy constraint in sensor nodes. The high directivity of optical communication enables the use of spatial division multiple access (SDMA), which requires no communication overhead and has the potential to be more energy efficient than the medium access schemes used in RF, such as time, frequency, and code division multiple access (TDMA, FDMA, and CDMA).

It has been shown that DVS based power management has significantly higher energy efficiency compared to shut down - based power management. Meanwhile, power consumption can further be reduced through efficiently operating various system resources using some dynamic power management (DPM) technique. On the other hand, energy efficiency can significantly be enhanced if energy awareness is incorporated in the design of system software, including the operating

system, and application and network protocols. To achieve low - power consumption at the node level, it is necessary to incorporate power awareness and energy optimization in hardware design for sensor networks. For example, a commonly used DPM technique is to shutdown idle components or put them in a low - power state when there is little or no load to process, which can significantly reduce power consumption. Low - power circuit and system design has enabled the development of ultralow power hardware components, for example, microprocessors and microcontrollers.

1.7.1 Hardware Platform

This class of platforms include the Berkeley mote family, the UCLA Medusa family, and MIT μ AMP, which typically use commercial off - the - shelf chips and are characterized by small form factors, low - power processing and communication, and simple sensor interfaces. Compared with dedicated and SoC sensor nodes, these PC - like platforms have higher processing capability and thus can incorporate a richer set of networking protocols, popular programming languages, middleware, application programming interfaces (APIs), and other off - the - shelf software. This class of platforms include various low - power embedded PCs (e.g., PC104) and personal digital assistants (PDAs), which typically run off - the - shelf operating systems, for example, Win CE, Linux, or real - time operating systems, and use standard wireless communication protocols, for example, IEEE 802.11 or Bluetooth. Sensor node hardware platforms can be classified into three categories: augmented general - purpose personal computers (PCs), dedicated sensor nodes, and system - on - chip (SoC) sensor nodes.

1.7.2 Software Platform

A software platform can be an operating system that provides a set of services for applications, including fi le management, memory allocation, task scheduling, peripheral device drivers, and networking, or it can be a language platform that provides a library of components to programmers. Tiny OS is one of the earliest operating systems supporting sensor network applications on resource constrained hardware platforms, for example, the Berkeley motes. It defines virtual machine instructions to abstract those common operations, for example, polling sensors and accessing internal states. Tiny GALS is a language for Tiny OS, which provides a way of building event - triggered concurrent execution from thread - unsafe components. Therefore, software

written in Mot é instructions does not have to be rewritten to accommodate a new hardware platform with support for the virtual machine. It provides a set of language constructs and restrictions to implement Tiny OS components and applications.

The IEEE 802.15.4 Standard: The IEEE 802.15.4 is a standard developed by IEEE 802.15 Task Group 4, which specifies the physical and MAC layers for low - rate WPANs. As defined in its Project Authorization Request, the goal of Task Group 4 is to "provide a standard for ultralow complexity, ultralow cost, ultralow power consumption, and low - data rate wireless connectivity among inexpensive devices". The first release of the IEEE 802.15.4 standard was delivered in 2003 and is freely distributed. This release was revised in 2006, but the new release is not yet freely distributed. Its protocol stack is simple and flexible, and does not require any infrastructure. The standard has the following features:

- Data rates of 250 kbps, 40 kbps, and 20 kbps.
- Two addressing modes: 16 - bit short and 64 - bit IEEE addressing. • Support for critical latency devices, for example, joysticks.
- The CSMA - CA channel access.
- Automatic network establishment by the coordinator.
- Fully handshaking protocol for transfer reliability.
- Power management to ensure low - power consumption.
- Some 16 channels in the 2.4 - GHz ISM band, 10 channels in the 915 - MHz band, and 1 channel in the 868 - MHz band.

The ZigBee Standard: The IEEE 802.15.4 standard only defines the physical and MAC layers without specifying the higher protocol layers, including the network and application layers. The ZigBee standard is developed on top of the IEEE 802.15.4 standard and defines the network and application layers. The network layer provides networking functionalities for different network topologies, and the application layer provides a framework for distributed application development and communication. The two protocol stacks can be combined together to support short - range low data rate wireless communication with battery - powered wireless devices. The potential applications of these standards include sensors, interactive toys, smart badges, remote controls, and home automation. The ZigBee protocol stack was proposed at the end of 2004 by the ZigBee

Alliance, an association of companies working together to enable reliable, cost - effective, low - power, wirelessly networked, monitoring, and control products based on an open global standard. The first release of ZigBee was revised at the end of 2006, which introduces extensions on the standardization of application profiles and some minor improvements to the network and application layers.

The IEEE 1451 Standard: The IEEE 1451 standards are a family of Smart Transducer Interface Standards that defines a set of open, common, network - independent communication interfaces for connecting transducers (i.e., sensors or actuators) to microprocessors, instrumentation systems, and control/ field networks. Transducers have a wide variety of applications in industry, for example, manufacturing, industrial control, automotive, aerospace, building, biomedicine

IEEE 1451 standards:

- IEEE P1451.0 defines a set of common commands, common operations, and TEDS for the family of IEEE 1451 smart transducer standards. Through this command set, one can access any sensors or actuators in the 1451 - based wired and wireless networks.
- IEEE 1451.1 defines a common object model describing the behavior of smart transducers, a measurement model that streamlines measurement processes, and the communication models used for the standard, which includes the client - server and publish - subscribe models.
- IEEE 1451.2 defines a transducer - to - NCAP interface and TEDS for a point - to - point configuration.
- IEEE 1451.3 defines a transducer - to - NCAP interface and TEDS for multidrop transducers using a distributed communication architecture. It allowed many transducers to be arrayed as nodes, on a multidrop transducer network, sharing a common pair of wires.
- IEEE 1451.4 defines a mixed - mode interface for analog transducers with analog and digital operating modes.
- IEEE P1451.5 defines a transducer - to - NCAP interface and TEDS for wireless transducers. Protocol standards for wireless networks, for example, 802.11 (WIFI), 802.15.1 (Bluetooth), and 802.15.4 (ZigBee), are being considered as some of the physical interfaces for IEEE P1451.5.

Chapter 2
Literature Review

“Design and Implementation of an Economic Gas Leakage Detector”, - this project was developed to detect the gas leakage and providing immediate alarm or intimation to the user. This project was developed using microcontroller ARM version7 processor and simulated using Keil software.[1]

Providing the accurate calculation of the ratio is very important to increase the Electronic nose performance, and the result of this experiment showed that the Artificial neural network method achieves a good performance with smaller RMSE of 0.0433 compared with the existing methods [2].E-nose with MQ Family produces the ratio of existing air and base line air resistance, and it is usually equipped with a datasheet containing the consecration of detected gas in a certain value of the sensor to convert the output to the concentration of detected gas.

The prior is connected with cameras to undergo plant recognition, and utilizes database to find the suitable amount of water; the latter controls the amount of water that is able to flow out [15]. The researchers had developed an irrigation system, with the use of deep learning, that is able to adjust the amounts of water foe each type of plant through plants recognition.

This paper consists same like previous paper but here Arduino controller is used and detects the LPG gas leakage by gas sensor MQ-2 and within a few minutes automatically switch on the exhaust fan and buzzer to provide the alertness to the people. Here GSM SIM900A module is used to send the message to the predefined number so information can be sent and can prevent fire accidents [11].

Accordingly, this paper presents a brief overview of solid-state gas sensors, which can be classified into semiconductor, capacitor, and solid-electrolyte type sensors, based on their sensing mechanisms and a simple NDIR instrument. Furthermore, the sensing properties of solid-state gas sensors to environmental gases, such as NOX, SOX, CO2, volatile organic compounds (VOCs), plus certain other gases, are also classified and summarized [13].

The results proven that the step response of the second-order transfer function model can fit to the output signal of the gas sensors for different concentration when the parameters of ω/n (Natural Frequency), ζ (Overshoot) and K (Gain), chosen optimally. This paper presents, the second order

transfer function model for the gas sensor's transient response based on a smaller number of coefficients and equations.[14]

The importance of using such a system in Ghana is due to the increased spate of fire outbreaks in homes where at least one or more gas cylinders can be found [17] in 2017 published a research article where the design of an electronic gas leakage monitoring system in real-time is proposed, to alert people in homes and industries in Ghana of gas leakages through persistent text messages with accompanying beeps.

[19] proposed an Internet of Things-based cargo monitoring system (IoT-CMS) to monitor any environmental changes of ESPs in order to ensure their functional quality throughout the entire cold chain operational environment. Through applying (i) a wireless sensor network to collect real-time product information, together with (ii) fuzzy logic and case-based reasoning techniques to suggest appropriate storage conditions for various ESPs, effective storage guidance can be established.

[20] The controller is a closed-loop design which uses the soil moisture data to turn on and off the irrigation system based on specified thresholds. The system was developed at the Sensor and Automation Lab of Edisto-REC, Clemson University and then field tested at McCorkle Nurseries in Dearing, GA during summer 2017.The goal of this project was to design a prototype irrigation controller using the internet of things (IoT).

In this research it also showed how to control actuators over the net by just changing the state wise parameters from "0" to "1" and vice versa for controlling [16]. A research had been done on IoT based cold storage temperature and humidity monitoring, where NodeMCU and Thingspeak platform was used.

Chapter 3
Technical Specifications of Hardware and Software

3.1 Hardware requirements:

- NodeMCU
- LPG Gas sensor Module MQ-6
- DHT-11 Sensor
- Solenoid Valve (12V)
- 300 MBPS Router
- Jumper Wires
- Relay Modules and etc.

3.2 Specifications of components

3.2.1 NodeMCU Specifications

NodeMCU is an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. The term "NodeMCU" by default refers to the firmware rather than the development kits. The firmware uses the Lua scripting language. It is based on the eLua project, and built on the Express if Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson and spiffs [36]. Technical specifications are given below:

- Voltage:3.3V.
- Wi-Fi Direct (P2P), soft-AP.
- Current consumption: 10uA~170mA.
- Flash memory attachable: 16MB max (512K normal).
- Integrated TCP/IP protocol stack.
- Processor: Tensilica L106 32-bit.
- Processor speed: 80~160MHz.

- RAM: 32K + 80K.
- GPIOs: 17 (multiplexed with other functions).
- Analog to Digital: 1 input with 1024 step resolution.
- +19.5dBm output power in 802.11b mode
- 802.11 support: b/g/n.

A graphical schematic of NodeMCU is given below in Figure-2.1 [36].

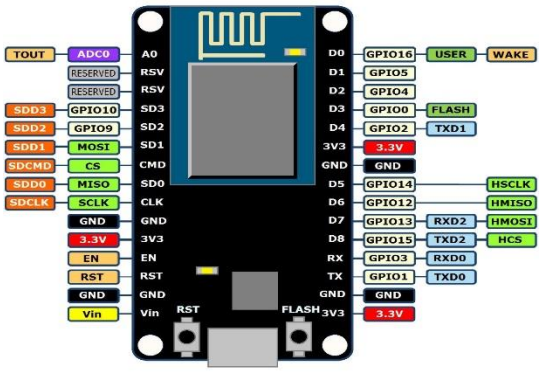


Figure-3. 1 NodeMCU

While writing GPIO code on NodeMCU, you can't address them with actual GPIO Pin Numbers. There are different I/O Index numbers assigned to each GPIO Pin which is used for GPIO Pin addressing. Figure-2.2 shows a graphical representation of NodeMCU pin settings. Refer following Table-1 to check I/O Index of NodeMCU GPIO Pins –

Table-1: 1 NodeMCU pin settings

GPIO Pin	**I/O Index Number**
GPIO0	3
GPIO1	10
GPIO2	4
GPIO3	9
GPIO4	2
GPIO5	1
GPIO6	N/A
GPIO7	N/A
GPIO8	N/A
GPIO9	11
GPIO10	12
GPIO11	N/A
GPIO12	6
GPIO13	7
GPIO14	5
GPIO15	8
GPIO16	0

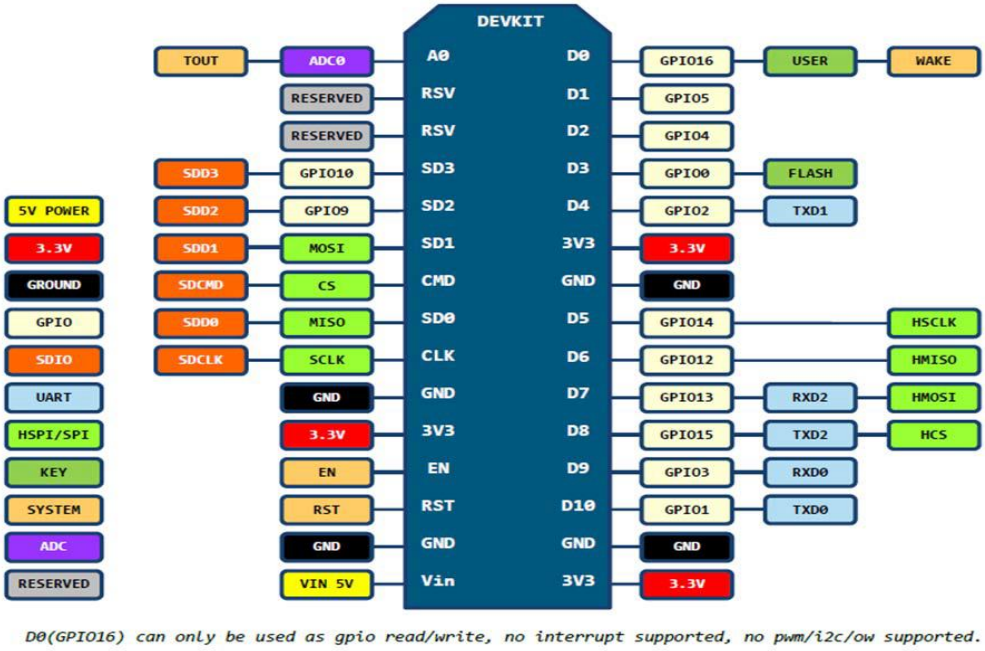


Figure-3. 2 NodeMCU v.1.0 Pin Definition

The NodeMCU can be programmed with the Arduino software (Arduino IDE). Select "ESP 8266 NodeMCU v.1.0" from the Tools > Board menu. The microcontroller provides pre-burned with a boot loader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files). It also can bypass the boot loader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header. The NodeMCU has a resettable poly fuse that protects your computer's USB ports from shorts and over current. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

3.2.2 Gas sensor MQ-6 Module Specifications

MQ-6 is a semiconductor type gas sensor which detects the gas leakage. The sensitive material of MQ-6 is tin dioxide (SnO2). It has very low conductivity in clean air. MQ-6 gas sensor has high sensitivity to LPG, concentration level of it is from 200 – 1000 ppm and it also detects the

following flammable gases: 1. Propane 2. Hydrogen 3. Methane 4. Butane. Figure-3.3 shows a schematic of MQ-6 gas sensor module [37].

Figure-3. 3 MQ-6 Gas sensor

The features of MQ-6 gas sensor are given below:

1. Wide detecting scope

2. High sensitivity to combustible gas in wide range

3. Fast response

4. Stable and long life

5. Simple drive circuit

6. Low cost and compact size.

The gas sensor senses the analog value according to the concentration of the gas level in the environment. The concentration range of MQ-6 gas sensor is 200-1000ppm for LPG and use value of Load resistance (R_L) about 20 k.Ohm (10 k.Ohm to 47 k.Ohm). When accurately measuring, the proper alarm point for the gas detector should be determined after considering the temperature and humidity influence. The voltage that the sensor outputs changes accordingly to the smoke/gas level that exists in the atmosphere. The sensor outputs a voltage that is proportional to the concentration of smoke/gas. The resistance of the sensor is different depending on the type of the gas. Figure-3.4 shows a schematic of inside circuitry of MQ-6 gas sensor.

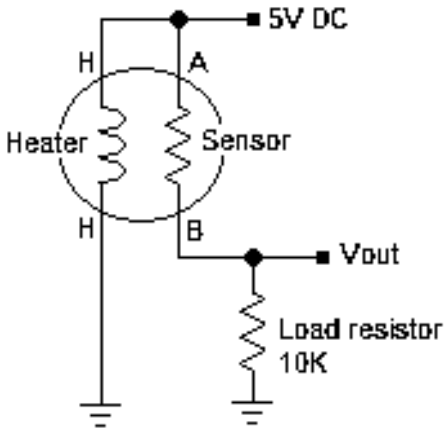


Figure-3. 4 MQ-6 circuit.

MQ-6 sensor senses the flammable gases by the increase in temperature when they are oxidized by the heating element. Consider the figure given above. If there is any flammable gas present in the sample, the oxidization of the same gas results in increased temperature and the resistance of the sensor resistor will drop. That means more current will flow through the load resistor and so the voltage across it will shoot up. MQ-6 gas sensor pin description: 1. D_0 (Digital Pin), 2. A_0 (Analog Pin), 3. V_{CC} (+5V), 4. GND [Graphical illustration in Figure-2.5]

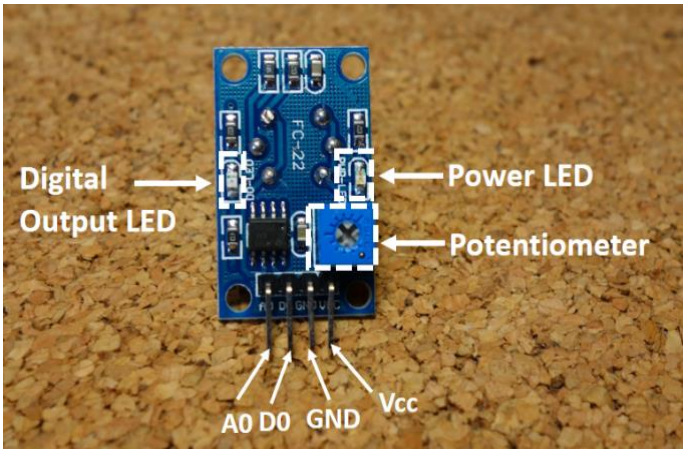


Figure-3. 5 MQ-6 pinout.

The working condition is very important for the sensory performance. Table-2 shows the standard working conditions and optimal operations of MQ-6 gas sensor. Sensitivity is also an important factor for the sensory signal calibration. Figure-2.6 and Figure-3.7 shows the sensitive relations of MQ-6 with respect to ppm (Parts per Millions) and Relative Humidity respectively.

Table-3: 1 Standard Working Condition:

Symbol	Parameter Name	Technical Condition	Remarks
V_C	Circuit voltage	5V±0.1	AC or DC
V_H	Heating voltage	5V±0.1	AC or DC
R_L	Load resistance	Adjustable	
R_H	Heater resistance	33k.Ohm±5%	Room Temperature
P_H	Heating consumption	Less than 800mW	

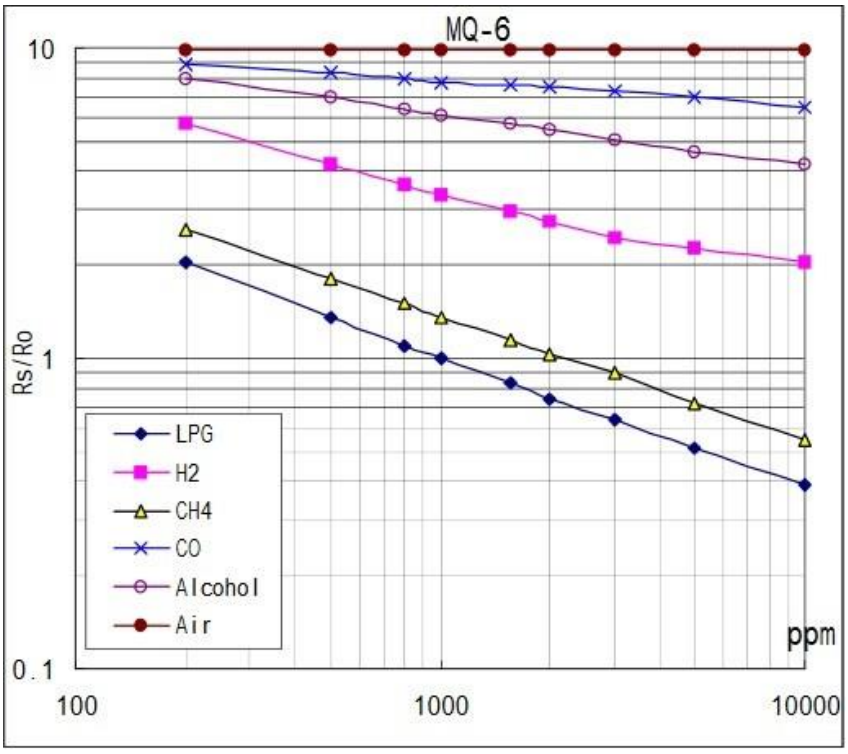


Figure-3. 6 MQ-6 gas sensor sensitivity at optimum conditions [38].

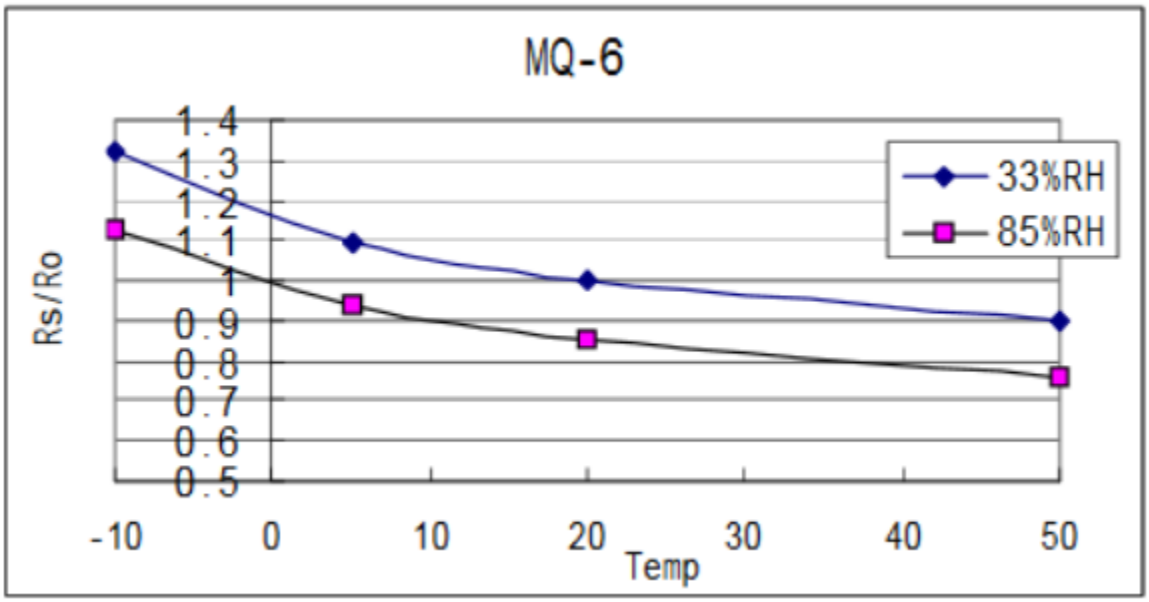


Figure-3. 7 MQ-6 sensor sensitivity with respect to Temperature [38].

Resistance value of MQ-6 is difference to various kinds and various concentration gases. So, when using these components, sensitivity adjustment is very necessary. Recommended that you calibrate the detects for 1000ppm liquefied petroleum gas (LPG), or 1000ppm iso-butane concentration in air and use value of load resistance that about 20 Kilo-Ohms. When accurately measuring the proper alarm point for the gas detector should be determined after considering the temperature and humidity influence. Applications of MQ-6 gas sensor is given below:

- Domestic gas leakage detector.
- Industrial combustible gas leakage detector.
- Portable gas leakage detector.
- Concentration level for LPG is 400-1000ppm.
- Circuit voltage is 5V.
- Heating consumption less than 800mw.
- Load resistance adjustable.
- Wide detecting scope.

- High sensitivity to combustible gas in wide range.
- Fast response.
- Stable and long life.

3.2.3. Solenoid Valve Specifications

- APL-3/2’’-12VDC
- Position: Normally Closed
- Port Size: 1/2" Male NPT
- Voltage: 12V DC
- Body Material: POM Plastic
- Components: Stainless Steel
- Seal Material: EPDM Diaphragm
- Orifice Size: 8.5 mm
- Temp Range: 32 to 125° F / 0 to 50°C
- Pressure Range: 3 - 115 PSI (Minimum Required)
- Flow Rate: Cv 0.6 (Appx 4.5 GPM @ 60 PSI)
- Power: 6 Watts

Below Figure-3.8 Shows a graphical illustration of 12 V (DC) Solenoid Valve.

Figure-3. 8 12V DC solenoid valve [39].

3.2.4 DC Pump Specification

- Dimension: 8 x 6 cm
- Brushless pump power: 12V 960mA
- Max water height: 6m
- Max flow: 460LPH
- Lifespan: not less than 20000 hours
- Submersible or the diving installation
- Cable length: 40 cm
- Color: black
- The lead-out wire is 10 cm
- Net weight: 200g
- Package dimension: 88 x 56 x 70 mm

Below Figure-3.9 Shows a graphical illustration of 12 V DC Pump

Figure-3. 9 DC Pump

3.3. Software Description

Software tools that are used in the project

1) Arduino IDE (v1.0.6)
2) MySQL Database

3.3.1 Arduino IDE (v1.0.6)

The Arduino integrated development environment (IDE) is a cross-platform application written in Java, and derives from the IDE for the Processing programming language and the Wiring projects. It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation, and is also capable of compiling and uploading programs to the board with a single click. A program or code written for Arduino is called a "sketch". Figure-2.9 shows the graphical picture of Arduino-IDE [40].

Figure-3. 10 Arduino-IDE lookout

Arduino programs are written in C or C++. The Arduino IDE comes with a software library called "Wiring" from the original Wiring project, which makes many common input/output operations

much easier. The users need only to define two functions to make an executable cyclic executive program:

- setup(): a function run once at the start of a program that can initialize settings
- loop(): a function called repeatedly until the board powers off

A typical first program for a microcontroller simply blinks an LED on and off. In the Arduino environment, the user might write a program like this:

```
#define LED_PIN 13

void setup() {
    pinMode(LED_PIN, OUTPUT);       // Enable pin 13 for digital output
}


void loop() {
    digitalWrite(LED_PIN, HIGH);    // Turn on the LED
    delay(1000);                    // Wait one second (1000 milliseconds)
    digitalWrite(LED_PIN, LOW);     // Turn off the LED
    delay(1000);                    // Wait one second
}
```

It is a feature of most Arduino boards that they have an LED and load resistor connected between pin 13 and ground; a convenient feature for many simple tests. The previous code would not be seen by a standard C++ compiler as a valid program, so when the user clicks the "Upload to I/O board" button in the IDE, a copy of the code is written to a temporary file with an extra include header at the top and a very simple main() function at the bottom, to make it a valid C++ program. The Arduino IDE uses the GNU tool chain and AVR Libc to compile programs, and uses avrdude to upload programs to the board. As the Arduino platform uses Atmel microcontrollers, Atmel's

development environment, AVR Studio or the newer Atmel Studio, may also be used to develop software for the Arduino.

3.3.2 MySQL database

MySQL is written in C and C++. Its SQL parser is written in yacc, but it uses a home brewed lexicalanalyzer. MySQLworksonmany systemplatforms,including AIX, BSDi, FreeBSD, HPUX, eComStation, i5/OS, IRIX, Linux, macOS, MicrosoftWindows, NetBSD, Novell,NetWare, OpenBSD, OpenSolaris, OS/2 Warp, QNX, Oracle Solaris, Symbian, SunOS, OpenServer, SCO UnixWare, Sanos and Tru64. A port of MySQL to OpenVMS also exists. The MySQL server software itself and the client libraries use dual-licensing distribution. They are offered under GPL version 2, or a proprietary license. Support can be obtained from the official manual. Free support additionally is available in different IRC channels and forums. Oracle offers paid support via its MySQL Enterprise products. They differ in the scope of services and in price. Additionally, a number of third party organizations exist to provide support and services, including MariaDB and Percona.

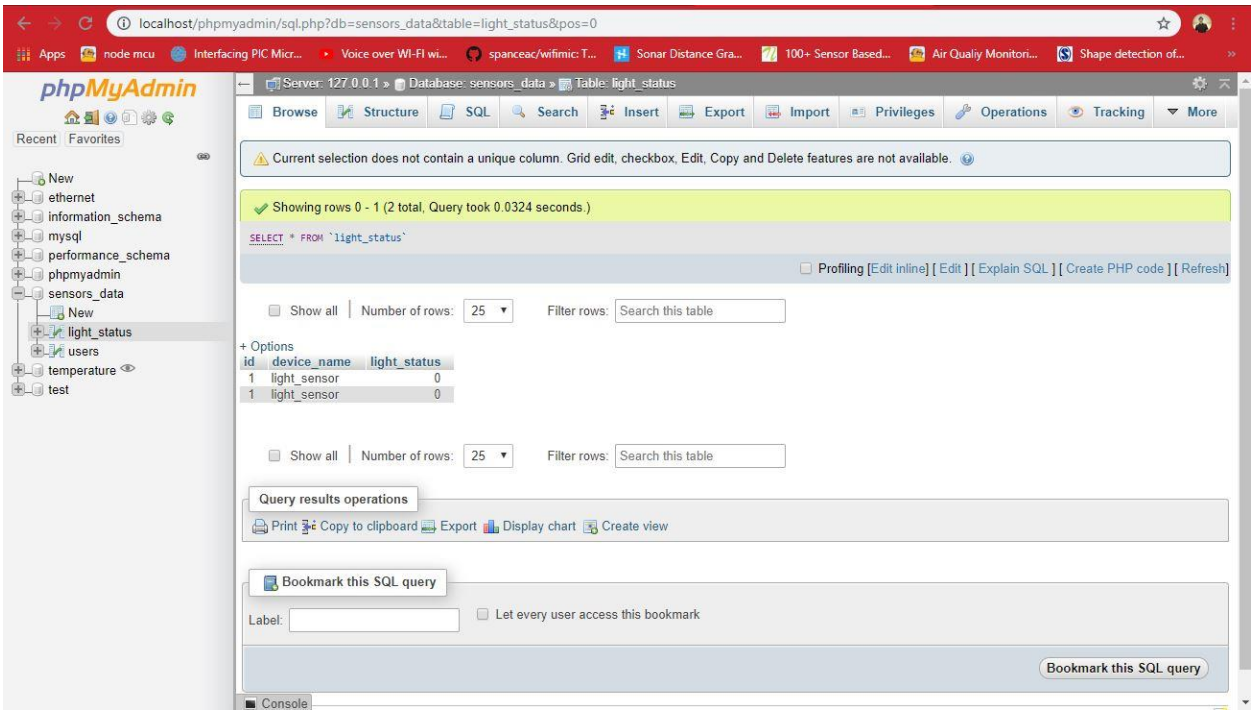


Figure-3. 11 Php My Admin console

Chapter 4
System Methodology

4.1 System Overview

In this thesis, environment monitoring systems is implemented by using sensors and then send the sensors data are sent to a local internet protocol (IP) using Wi-Fi. For enchanting data, DHT-11 (Temperature and Humidity sensor) and MQ-6 (LPG gas sensor) are being used. The basic objective of this work is to monitor and to develop a real-time monitoring of humidity and temperature, as well as the availability of gas using the very available DHT-11 sensor, MQ-6 sensor, and ESP-8266 NodeMCU module and then observe the data from a local IP based webpage. In this, a system has been developed to monitor the real-time condition (Humidity and Temperature to be more precise) through a local IP without the access of internet, because of security reason and so on; in which physical presence is not needed.

Figure-4. 1 System Block Diagram

Node MCU is a kind of component that consists of an ESP8266 Wi-Fi along with a microcontroller with 1 analog and 12 digital pins, for which it is more efficient than an Arduino board and the data rate is also effective for this module. Combining the results with MATLAB environment, the upcoming data can also be forecasted through the basic knowledge of MATLAB or other engineering and scientific software. However, using an Arduino board is not necessary for just monitoring services, because of cost management and time consumption. This work has been

directed using very modest methodology and appliances which are available and require minimal technical knowledge to operate. It discusses one of the implementations of a system using Wireless Sensor Networking (WSN). MQ-2 gas sensor and IR photodiode are used to determine smoke/gas concentration and fire presence respectively. As the gas sensor works fine within temperature ranging 2535 0C & RH ranging 55-70%, so DHT-11 sensor is used to check the temperature and relative humidity level of the experimental plant. Further mathematical modeling and analysis are done for the steady-state design of the experimental system. After that acquisition of those factors which combines a threshold, value has to be set for controlling the total system modeling. From the datasheet of DHT-11, MQ-2 and IR photodiode, parameters and the operations are obtained. The DHT-11 sensor is a digital sensor with digital output data, which combines Integrated Circuits, is the reason behind the direct values of relative humidity and temperature can be obtained.

4.2 Why ESP8266 is Used

In this project work we have used ESP8266 as MCU because of the following reasons:

- **Ideal Stuff for IoT and WSN:** ESP8266 is ideal for Internet of Things (IoT) My current project involves home automation and IoT stuff. It would send the status of a PIR Sensor over WIFI to an MQTT broker every 2.5 seconds. With built-in WIFI, these boards are ideal for Internet activities.
- **Programmable PWM Frequency:** The ESP8266 core of the Arduino library, however, has a built-in PWM frequency call. The ESP8266 core has 1024 (0-1023) levels of pulse-width instead of Arduino's 256 (0-255). One option is to change the PWM frequency, but that would mean digging in the Arduino libraries.
- **Any pins can be selected for I2C:** A simple call before initializing the wire library allows for I2C communication on any of the module's pins.
- **Works with Arduino IDE:** The reason for choosing this is that the ESP8266 works well with the Arduino IDE
- **Inexpensive:** The Adafruit ESP8266 boards have incredible build quality, nice features (like all the necessary pull-ups), and support a company that supports the maker community in countless ways. Moreover, it is cheap in price.

4.3 System Algorithm

Firstly, when the Node MCU is turned, it first runs the program compiled in it and starts calibrating the sensors. On concluding calibration, it first goes to connect the local server *via* triggering its Tx and Rx. Now, two types of conditions are relevant; at calibration, the controller detects any problem in sensor value then it sends a signal to the control unit and commands it to make an action. Next, it starts recalibration and if no defect is found then it tries to connect the hotspot which is specified in the program and it checks for the preprogrammed SSID and if found nearby then the connection is created and an IP address is obtained. After connecting to the internet, it tries to connect with the prespecified Local IP. Then the controller creates a data transmission network between the sensors and the IP. Data from the sensor are transferred to the server. The system flowchart is shown in Fig. (**3**)

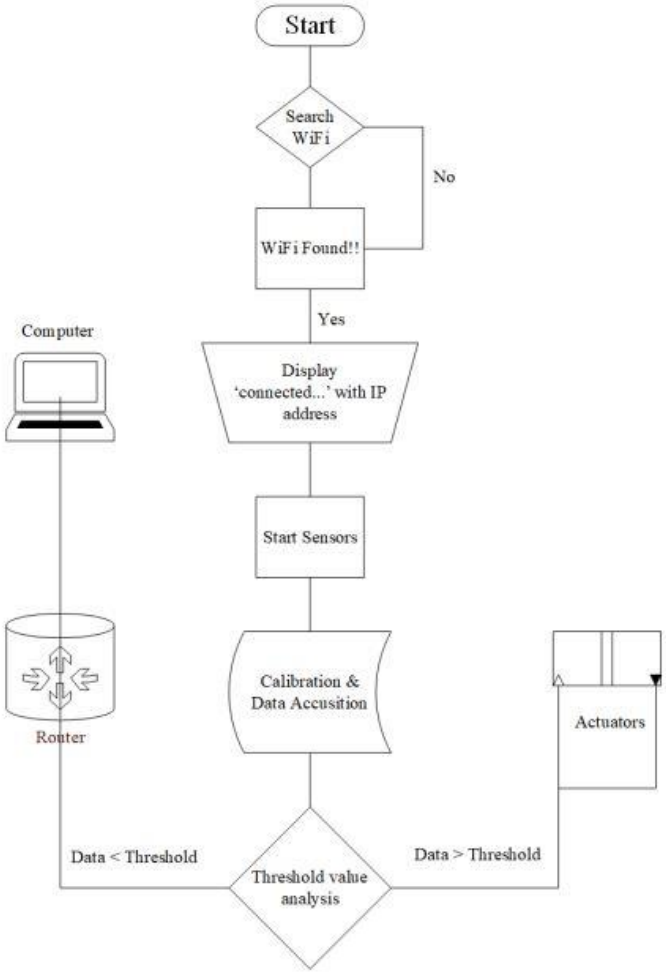


Figure-4. 2 Flowchart of the monitoring system

Chapter 5
Mathematical Modeling

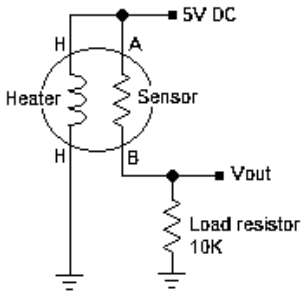
As the system focuses on gas concentration-based alerting and control of appliances, thus only MQ-6 gas sensor calibration is considered for the mathematical analysis. DHT-11 sensor is to for checking the conditional Temperature and Relative Humidity.

Let, V_C = Supply Voltage = +5V

R_S = Sensor Resistance

R_L = Load Resistance (Variable)

V_{out} = Sensor output Voltage



From, Current flow and Voltage relationship,

$$V = I \times R \tag{5.1}$$

$$V_{out} = \frac{\text{Analog value} \times 5}{1023}$$

Resistance of the sensor () is defined in datasheet [38] of MQ-6 as:

$$R_s = \left(\frac{V_{cc}}{V_{out}} - 1\right) \times R_L$$

$$= \left(\frac{1023}{\text{Analog value}} - 1\right) \times R_L \tag{5.2}$$

Fresh air resistance ration for gas sensors: [38]
Data analyzing from R_S/R_0 vs ppm graph: From the equations of a straight line,

$$y = mx + b \tag{5.3}$$

Here, y = value on Y-axis; x = value on X-axis; m = Slope of line; b = Intercept from Y-axis

For log-log plot equation-(5) becomes:

$$\log(y) = m \times \log(x) + b \tag{5.4}$$

Note: the log is base 10.

Slope (m) Value formula: If (x_0, y_0) and (x, y) are any two points of a line from a log-log plot then the formula for determining m is below:

$$m = \frac{\log_{10}(\frac{y}{y_0})}{\log_{10}(\frac{x}{x_0})} \quad (5.5)$$

Intercept from Y-axis (b) Value formula:

From, equation-(5.6);

$$\log_{10}(y) = m \times \log_{10}(x) + b$$

Or, $$b = \log_{10}(y) - m \times \log_{10}(x) \quad (5.6)$$

Using equations-(1) to (8) the gas concentration can be determined directly in ppm (Parts per millions)

$$\log_{10}(ppm) = \frac{(\log_{10}\left(\frac{R_s}{R_0}\right) - b)}{m} \quad (5.7)$$

From the MQ-6 datasheet we have obtained the value of m and b from equation (5.5) and (5.6)

Slope, $m = -0.423$
And $b = 1.276$
And sensor resistance R_0=5.62 kΩ

Table- 5. 1 Raw voltage values received through the MQ-6 sensor and corresponding temperature and humidity values:

ID	Time	Temperature (in °C)	Humidity (in %)	Analog value
1	13:19:41	30	61	533.15
2	13:19:36	30	61	533.48
3	13:19:31	30	61	532.48
4	13:19:25	30	61	528.04
5	13:19:20	30	61	532.48
6	13:19:15	30	61	532.48
7	13:19:10	30	61	528.04
8	13:19:05	30	61	528.04
9	13:18:59	30	61	528.04
10	13:18:54	30	61	528.04

Table- 5. 2Raw voltage converted to resistance value and then the corresponding PPM, R_0 was previously calibrated

ID	Time	Resistance (R_s)	PPM
1	13:19:41	9187.764	325
2	13:19:36	9175.828	316
3	13:19:31	9211.781	313
4	13:19:25	9373.252	310
5	13:19:20	9211.781	313
6	13:19:15	9211.781	313
7	13:19:10	9373.252	310
8	13:19:05	9373.252	310
9	13:18:59	9373.252	310
10	13:18:54	9373.252	310

Table- 5. 3 The Temperature and Humidity correction factor is used and the Corrected Resistance and subsequently the Corrected PPM is evaluated: [69]

$$\mathrm{R_s\_Scaling\ factor} = 1.6979 - 0.012\mathrm{t} - 0.00612\mathrm{h} \quad (5.8)$$

$$\mathrm{Corrected\ R_s} = \frac{\mathrm{R_s}}{\mathrm{R_s\_Scaling\ factor}}.$$

ID	Time	Resistance Corrected R_s	Corrected PPM	Correction Factor
1	13:19:41	7177.940	582.55	1.28
2	13:19:36	7168.615	584.34	1.28
3	13:19:31	7196.703	578.96	1.28
4	13:19:25	7338.478	552.87	1.28
5	13:19:20	7196.703	578.96	1.28
6	13:19:15	7196.703	578.96	1.28
7	13:19:10	7338.478	552.87	1.28
8	13:19:05	7338.478	552.87	1.28
9	13:18:59	7338.478	552.87	1.28
10	13:18:54	7338.478	552.87	1.28

5.1 Algorithm of the process

```
setup( )
{ setAnalogPinforMQ135(analog0);
setAnalogP
inforDHT11(analog1);
Serial.begin(9600);
}
loop( )
{  AnalogValue = receiveMQdata( );
T = TempfromDHT11( );
H = HumfromDHT11( );
Vout= convertanalogtovoltage(AnalogValue);
while(calibrationisnotcomplete)
{ RO= calculateRoValue(Vout, DefaultPPM);}
RS= calculateRsValue(CaliberatedRO, Vout);
CorrectedRS= TempHumCalibRs(RS, T, H);
ppm = PPM(RS, RO, MQ_Scaling_Factor, MQ_Exponent_Factor);
Correctedppm = PPM(CorrectedRS, RO, MQ_Scaling_Factor, MQ_Exponent_Factor);
PlotDatainMatlab( );
SendDatatoSerialMonitor( );
SendDatatoExcel( );
}
```

5.2 Calculation of the solar PV energy output of a photovoltaic cell [70]

We construct a solar panel which is used for powering the controller. A dimension about 11.5cm × 6 cm rectangular solar plate is used to make the solar panel.

Global formula: $E = A \times r \times H \times PR$ (5.9)

E = Energy (kWh)	1 kWh/an
A = Total solar panel Area (m²)	0.0069 m²
r = solar panel yield (%)	15%
H = Annual average irradiation on tilted panel (shadings not included) *	1250 kWh/m².an
PR = Performance ratio, coefficient for losses (range between 0.9 and 0.5, default value =0.75)	0.75

5.3 Relative Humidity Calculation using DHT11 Sensor

If you know the temperature and the dewpoint, and want to obtain relative humidity, the formulas are as follows:

First, to convert the temperature and the dewpoint from Fahrenheit to Celsius, use the following formulas.

$$T_c = \frac{5\times(T_f - 32.0)}{9} \quad (5.10)$$

$$T_{dc} = \frac{5\times(T_{df} - 32.0)}{9} \quad (5.11)$$

T_c=air temperature in degrees Celsius, T_f=air temperature in degrees Fahrenheit

T_{dc}=dewpoint temperature in degrees Celsius

T_{df}=dewpoint temperature in degrees Fahrenheit

The next set of formulas assumes a standard atmospheric pressure. These formulas will calculate saturation vapor pressure (Es) and actual vapor pressure(E) in millibars.

$$E_s = 6.11 \times 10.0 \times \left(\frac{7.5\times T_c}{237.7+T_c}\right) \quad (5.12)$$

$$E = 6.11 \times 10.0 \times \left(\frac{7.5\times T_{dc}}{237.7+T_{dc}}\right) \quad (5.13)$$

Once the saturation vapor pressure and the actual vapor pressure, relative humidity can be computed by dividing the actual vapor pressure by the saturation vapor pressure and then multiplying by 100 to convert the quantity to a percent.

$$\text{Relative Humidity (RH) in percent} = \frac{E}{E_s} \times 100\% \quad (5.14)$$

For example, a station report that included an air temperature of 85 degrees Fahrenheit and a dewpoint of 65 degrees Fahrenheit and you wanted to compute the relative humidity, you would proceed as follows.

First, convert the Fahrenheit values to Celsius using formulas. The values should be T_c=29.4 and T_{dc}=18.3

Next, calculate the saturation vapor pressure and the actual vapor pressure using formulas (5.12) and (5.13) respectively. The values should be E_s =40.9 and E =21.0

Finally, calculate relative humidity using formula (5.14). The final answer should be RH=51.3 %.

Chapter 6
Experimental setup

The WSN has one or more sensor node. These sensor nodes are used to extract data from the environment of the home. A Router is used to develop local IP based web server. It will provide a

muscular networking mechanism over comfortable range of Wireless sensor areas over local IP. The wireless sensor node is consisting of ESP8266 Wi-Fi module, sensors and actuators. A webserver is designed for user interface to monitor the sensor data and control the node operation and actuation state.

In this chapter the detailed implementation of the proposed system is aggregated. The main structures that are needed to be developed for home automation are the Implementation and design of

(1) WSN
(2) Server gateway
(3) Local server
(4) Web interface

The WSN has one or more sensor node. These sensor nodes are used to extract data from the environment of the home.

6.1 Implementation of sensor-based node

The ESP8266 humidity-temperature sensor node uses the Digital Humidity and Temperature (DHT11) sensor, which is connected to the ESP8266 modules' GPIO pin 0. DHT11 sensor is used to collect the raw humidity and temperature data. It is a basic temperature and capacitive humidity sensor which works on 3-5 V power. The sensor node has a wireless connection with a router where the router works as a gateway in the star topology network.

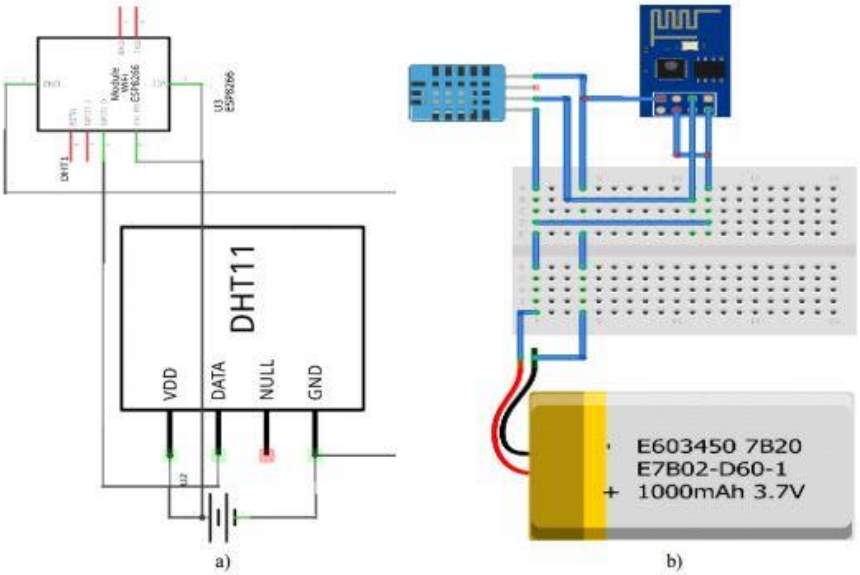


Figure-6. 1 Temperature and Humidity sensor node, a) schematic diagram and b) wring diagram

The gas sensor node is used for the monitoring and enhancing the kitchen safety of the home. The schematic and wiring diagram are presented in Fig. 6.2. The DIO pin of MQ-2 is connected to the ESP8266 DIO pin 0. The other connections are same as temperature and humidity sensor node.

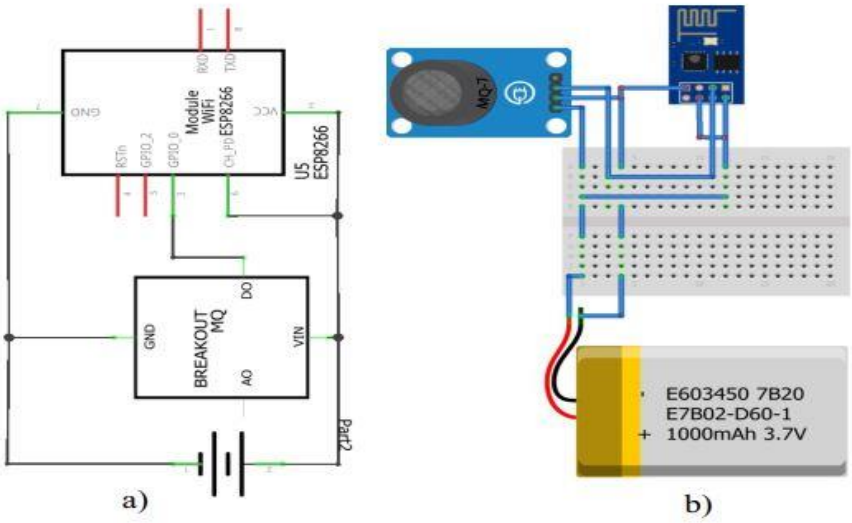


Figure-6. 2 Gas sensor node, a) schematic diagram and b) wring diagram

6.2 Implementation of solar panel

The ESP8266 is powered by the 3.7V battery that is connected to the Solar Lipo Charger in the battery input port. The solar cells are connected in the PWR In ports. The Vin and GND ports of the ESP8266 are connected to Vout ports of the Solar Lipo Charger.

The BME280 power is supplied by the 3.3V port in the ESP8266. The communication is done through the I2C lines (SDA / SCL). For the sleep mode, D0 need to be connected to reset pin (RST). To fix all components in the box, a preboard is used.

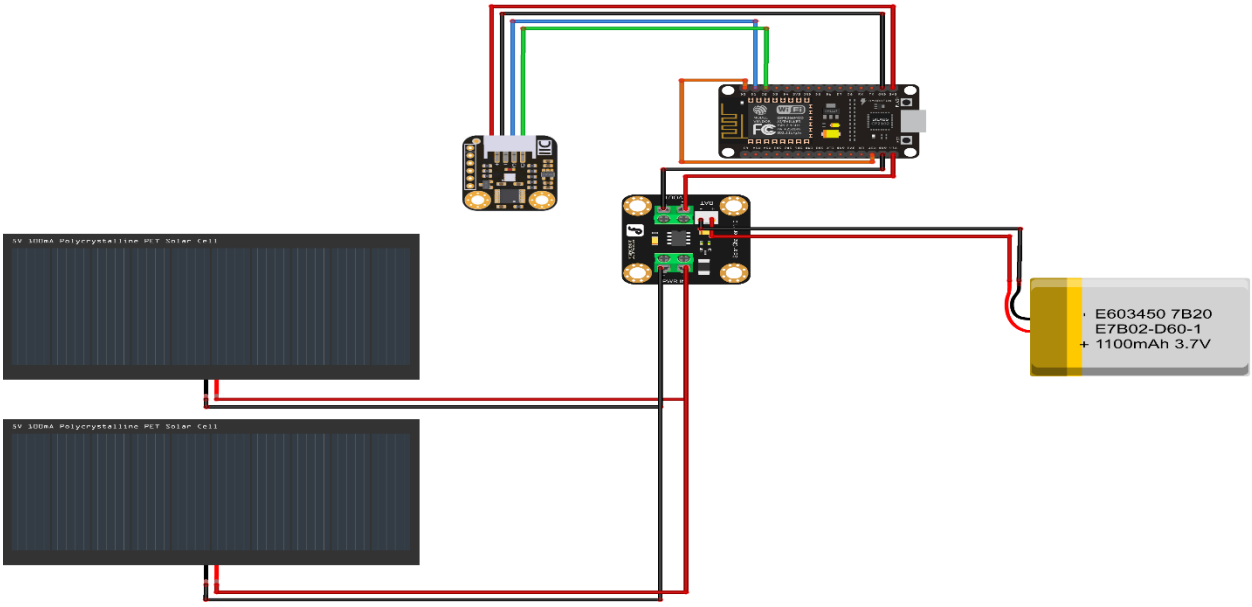


Figure-6. 3 Schematic diagram of a solar panel

6.3 Data Acquisition and Storage

The software service running on the server performs the following two tasks

(1) acquiring the data, and

(2) storing the data.

A TCP socket receives sensor data created by the gateway in the form of TCP packets. Once a UDP packet is received, the sample data is extracted from the payload, and the source address of the ESP8266 Module that sent the data is extracted from the IPv6 source address of the TCP packet. The second task is to create a database for sensor sample extracted from the TCP packet using the information extracted from a TCP packet.

Go Back to Previous Page

Data From Wireless Sensor Network

Search By Date 07/30/2019 Search

ID	Temperature(in °C)	Humidity(in %)	LPG(in ppm)	Time	Date
679	32	62	391	21:46:19	2019-07-30
678	32	62	394	21:46:14	2019-07-30
677	32	62	394	21:46:09	2019-07-30
676	32	62	384	21:46:04	2019-07-30
675	32	62	437	21:45:59	2019-07-30
674	32	62	421	21:45:53	2019-07-30
673	32	62	394	21:45:48	2019-07-30
672	32	62	391	21:45:43	2019-07-30
671	33	61	380	21:45:37	2019-07-30
670	33	61	384	21:45:32	2019-07-30
669	33	62	350	21:45:27	2019-07-30
668	30	63	363	21:45:21	2019-07-30
667	30	64	350	21:45:16	2019-07-30

Figure-6. 4 Data stored in database.

6.3.1 Receiving Data Samples

The basic network communication between two programs is defined as socket. It is the endpoint of a two-way communication link. In an OS, the socket has two operating mode, client and server. When the OS acts as a client it is connected to the server to exchange data through the socket. In server mode the socket listens for incoming connections from clients. A socket is bound to a port number so that the TCP layer can identify the application that data is destined to be sent to. An endpoint is a combination of an IP address and a port number.

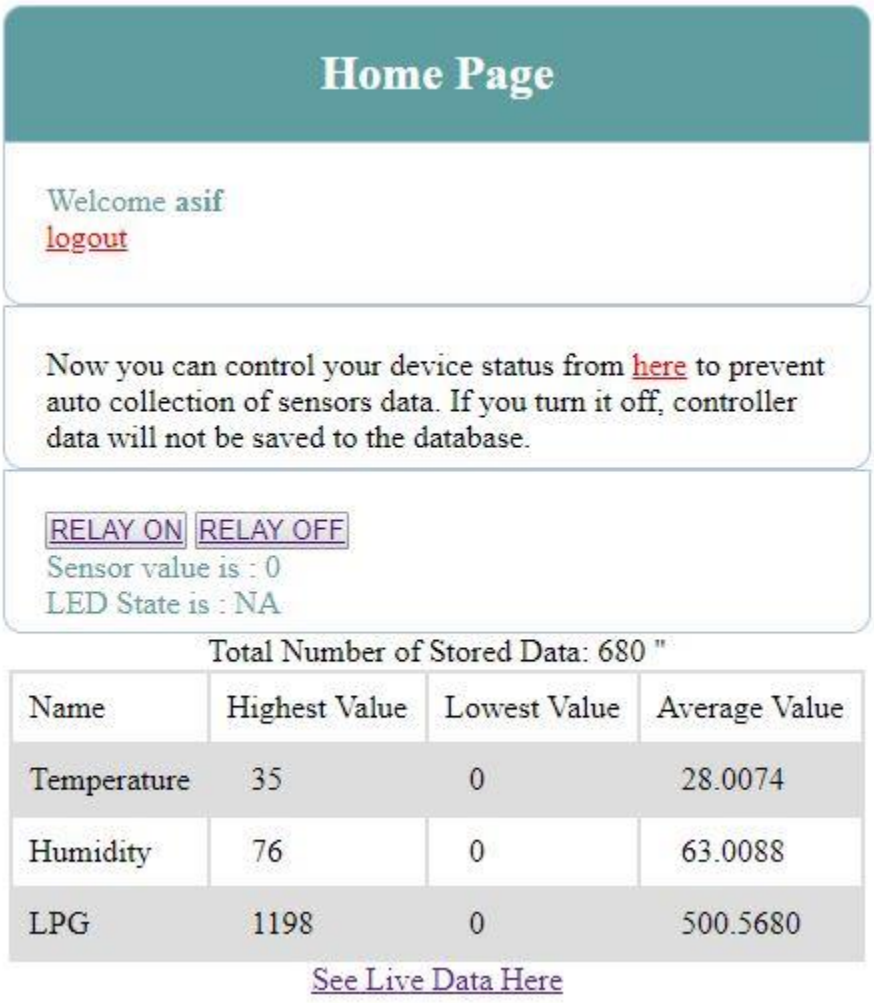


Figure-6. 5 Highest and lowest data samples.

6.3.2 Database Server

For hosting a website in the local machine XAMPP software package is used. Raspberry pi works as a local server machine in which XAMPP package is installed. It facilitates in hosting website and works as a local database server. The software package also contains Apache MySQL server. The MySQL server is used to create database for the proposed system which is shown in Fig. 4.7. In MySQL server, data acquired by sensors are stored and retrieved when needed. Also, information's of the end user are stored in the database. For the administration of the created database from remote panel PhpMyAdmin is also configured. When user sends a get request via the web interface PHP converts it into MySQL queries. PHP establishes the connection to the MySQL server and then fetch or post data according to the user command. The fetched data is then sent back to the html webpage for showing it to the user. HTML works in the client side for getting and PHP works in the server side for handling request for client. When the raspberry pi is powered and connected to the internet, the webpage hosted in it can be accessed from anywhere of the world. The sensor data from the TCP packet is stored in a MySQL database in an organized way. These sensors data are stored with a designated sensor ID, sensor channel and date and time in MySQL database so that the designed webpage can access those data.

(1) Sensor ID

(2) Sensor Channel

(3) Date and time

(4) Sample Data

The data stored in the database can be viewed in the webpage.

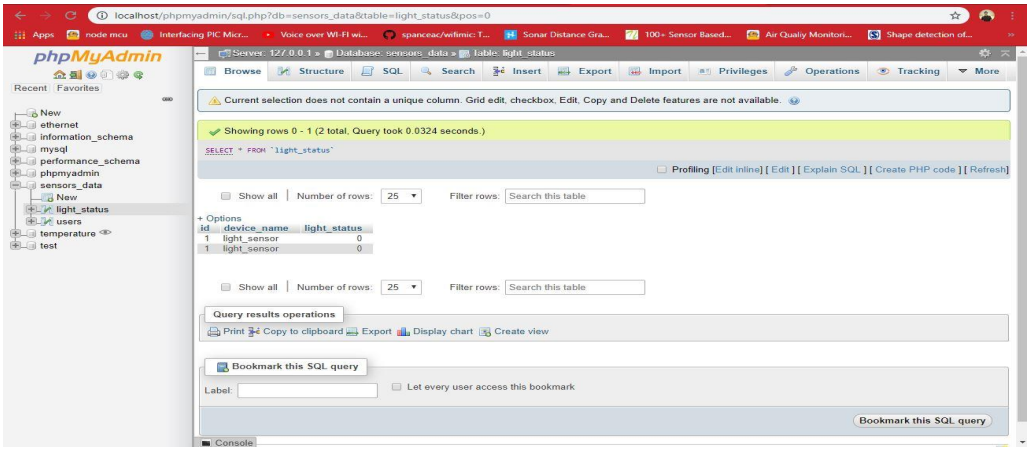


Figure-6. 6 Database Server (PHP my admin).

6.4 Web Interface

The website in the server which is used to control the digital output pins is a group of pages on World Wide Web containing buttons, sliders, sensors data and User Interface (UI) allowing the user to control the appliances all over the world through HTTP protocols. The webpage interface is used for controlling appliances, equipment like light, fan, cooler, air-conditioner etc. The status of different equipment is also visualized in the webpage. The developed webpage is presented in Fig. 4.8. Html files, .txt file and PHP files comprise the website to store data and to control the things by toggling the current state of the digital output pins. The web interface for the client side is implemented using HTML, CSS, JavaScript, Ajax, jQuery, and PHP script. The static web page is designed using HTML and CSS. CSS is a language that describes the style and elements of an HTML document. This static page is not interactive to user. To make it dynamic and interactive to the end user JavaScript is used for client-side scripting. jQuery is used to simplify the programing of JavaScript. It is a widely-used JavaScript library. Ajax is used as an interface language between client-side web applications and a server. It is a for user group of interrelated Web development techniques which is used on the client side to create asynchronous web applications. With Ajax, client-side web application can exchange data and status of the GPIO pin with a server asynchronously in the background without interfering with the display and behavior of the existing page. To continuously extract data from the MySQL data base A PHP script is written.

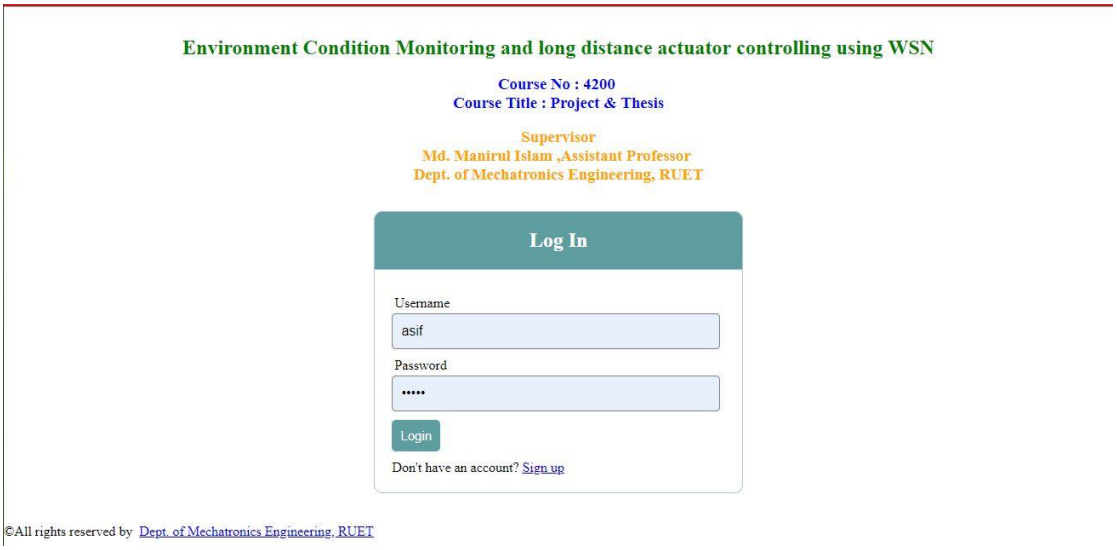


Figure-6. 7 Web interface

6.4.1 Retrieving Data from Database

To retrieve data from database the sensor id, sensor channel, and data and time range are required. These parameters are used to create a SQL query to the database to retrieve the required sensor samples. A PHP script is written which contains those database parameters to generate a SQL query for the purpose of retrieving data. These data are placed on JSON array. The output of the JSON array are then transferred to the web browser. The web browser calls the PHP scripts to retrieve data by running a java script. For the control of GPIO pins another PHP script which toggles the current state of the GPIO pins is also called by the web page to generate a control message when graphical interface buttons, sliders are operated

6.5 Controlling actuator from the web server

To take you one step ahead towards WSN development, today we will make our own web server to host a webpage and control any appliance remotely from anywhere in the world. Here we will use ESP12E NodeMCU as webserver, although any ESP module can be used here.

Figure-6. 8 Controlling Actuator from server

In simple terms, Web server is a place where we can store the web pages, process them and deliver them to the web clients. A protocol is used to establish and transfer information between web client and server. This protocol is known as Hypertext Transfer Protocol (HTTP**)**. In this protocol, communication is initiated by making a request for a particular web page using HTTP GET request and the server responds with the content of that web page. If server does not respond it will through an error message i.e. 404 Error. Webpages delivered by the server are mostly in HTML coding.

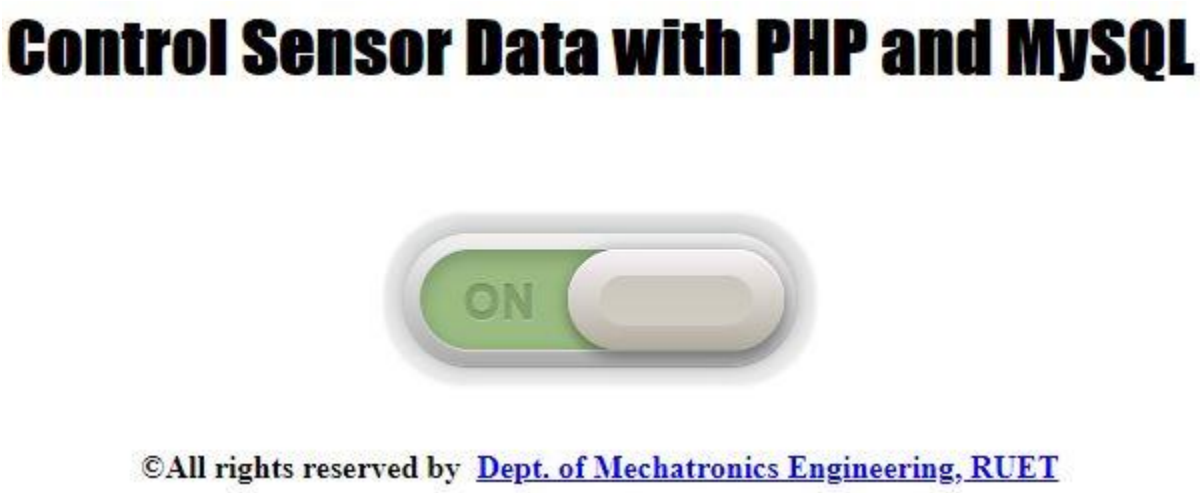


Figure-6. 9 Data storing state control

All the websites are hosted on some webserver, mostly Linux based operating system is used on webservers. Any computer can be converted into a webserver, provided that it is connected to the network. We have previously done many webserver projects with different microcontrollers. Raspberry pi already has inbuilt Wi-Fi module so it doesn't need any other hardware to turn it into a webserver, whereas other microcontroller needs some network connecter (ESP module) for a webserver.

Chapter 7
Test Results & Discussions

7.1 Sensor Signal Analysis & Processing

From experimental data, various calculations were performed using 'MATLAB' environment. The Digital Signal Analysis- Toolbar is a great tool for signal processing and smoothing. As the data were some arbitrary thus a processing is needed to overcome the problem. Mainly Fast Fourier Transformations were performed for better signal processing. The below figures are the results:

(i) Coherence Estimation via Welch Method: This algorithm is based on standard MATLAB's tools. The standard derivation of the mean is computed as [41-44];

$$\sigma(f) = \sqrt{\frac{2}{N|\Upsilon(f)^2|}(1 - |\Upsilon(f)|^2)^2} \tag{7.1}$$

Where,

$$|\Upsilon(f)|^2 = \frac{|P_{xy}(f)|^2}{P_{xx}(f).P_{yy}(f)} \tag{7.2}$$

Is the coherence function. By using equations-(1) & (2), Figure-7.1, can be plotted. Here, Figure-7.1 depicts the coherence estimation via Welch method. This figure shows that no data has been overlapped into one another.

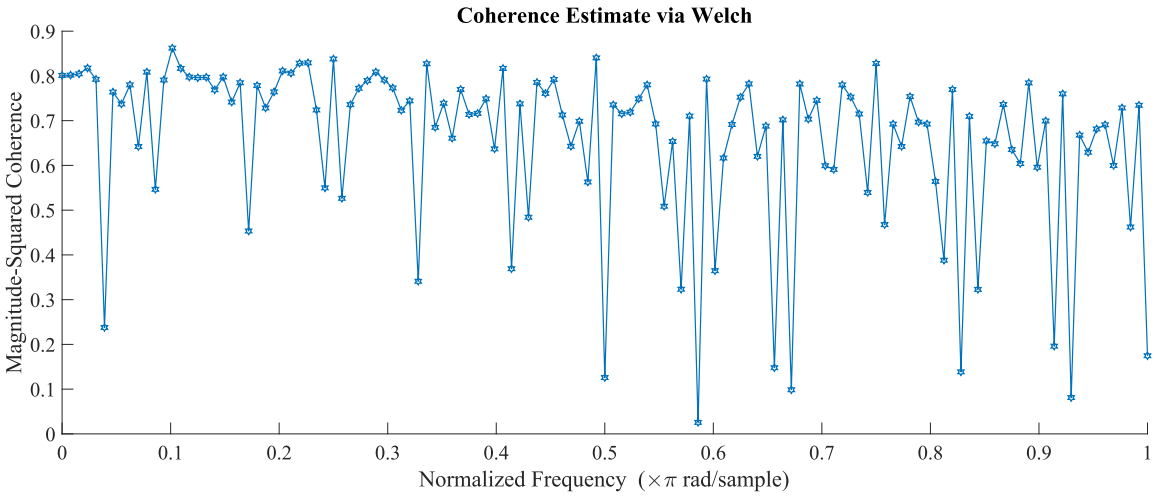


Figure-7. 1 Coherence estimation via Welch method

The Figure-7.2, denotes the magnitude response curve. The rms signal-to-noise ratio for an ideal N-bit converter is:

$$\text{Signal to Noise Ratio (SNR)} = 20\log_{10}\frac{\text{r.m.s value of } F_s \text{ input}}{\text{r.m.s value of quantization noise}} \tag{7.3}$$

$$\text{or, SNR} = 20\log_{10}\frac{q^{2^N}/2\sqrt{2}}{q/\sqrt{2}} = 20\log_{10}2^N + 20\log_{10}\sqrt{\frac{3}{2}} \tag{7.4}$$

SNR = 6.02N + 1.76 dB (13), over the Nyquist bandwidth of interest.

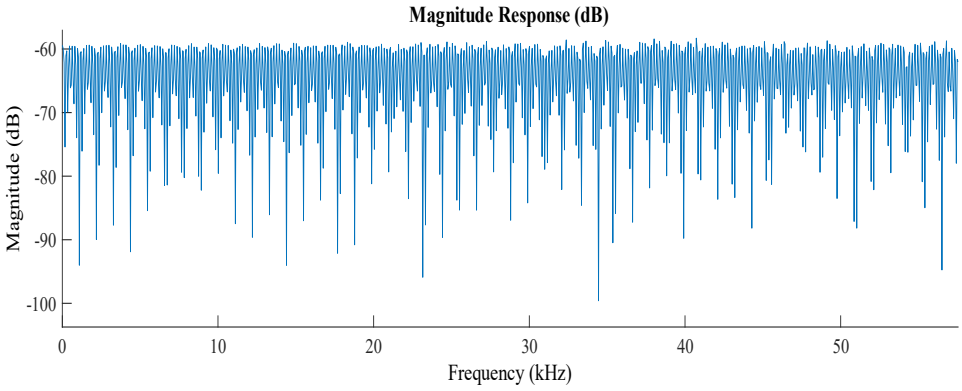


Figure-7. 2 Magnitude Response Graph (Gas Sensor)

Figure-7.3, represents the combination of magnitude and phase response. The plot shows limited errors.

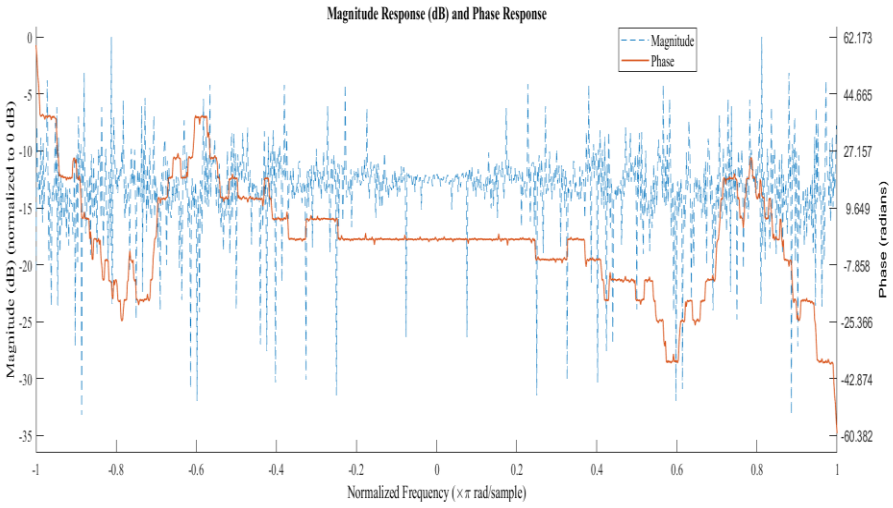


Figure-7. 3 Magnitude and Phase Response (Gas Sensor)

(ii) Time Domain signal processing: Now, using the gas sensor data Time-domain graph was plotted in Figure-7.4 to Figure-7.7, along with a 3rd and 8th order Fast Fourier Transformation (FFT) of the signal for better analysis. The Fourier series is a sum of sine and cosine functions that describes a periodic signal. It is represented in either the trigonometric form or the exponential form. The toolbox provides this trigonometric Fourier series form [45]:

$$y = a_0 + \sum_{i=1}^{n} \mathrm{a}_i \cos(iwx) + b_i sin(iwx) \tag{7.5}$$

where a_0 models a constant (intercept) term in the data and is associated with the $i = 0$ cosine term, w is the fundamental frequency of the signal, n is the number of terms (harmonics) in the series, and $1 \leq n \leq 8$. A Savitzky–Golay filter is a digital filter that can be applied to a set of digital data points for the purpose of smoothing the data, that is, to increase the signal-to-noise ratio without greatly distorting the signal. This is achieved, in a process known as convolution, by fitting successive sub-sets of adjacent data points with a low-degree polynomial by the method of linear least squares. Figure-7.4 depicts the plots of Raw analog, Savitzky Golay Filtered value, moving avg. filtered value and the median plot.

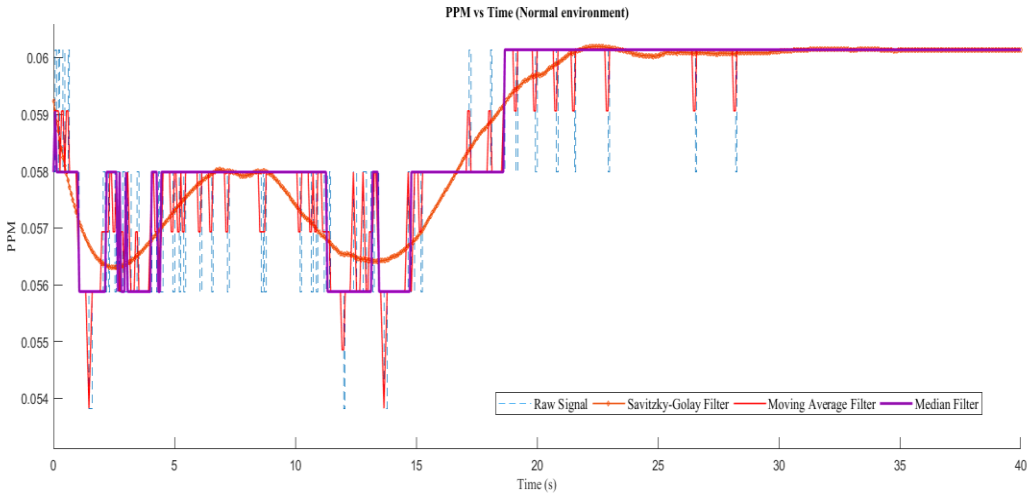


Figure-7. 4 Processed Signal (PPM vs Time)

Figure-7.5 shows the Furrier fitted curve for gas sensor value.

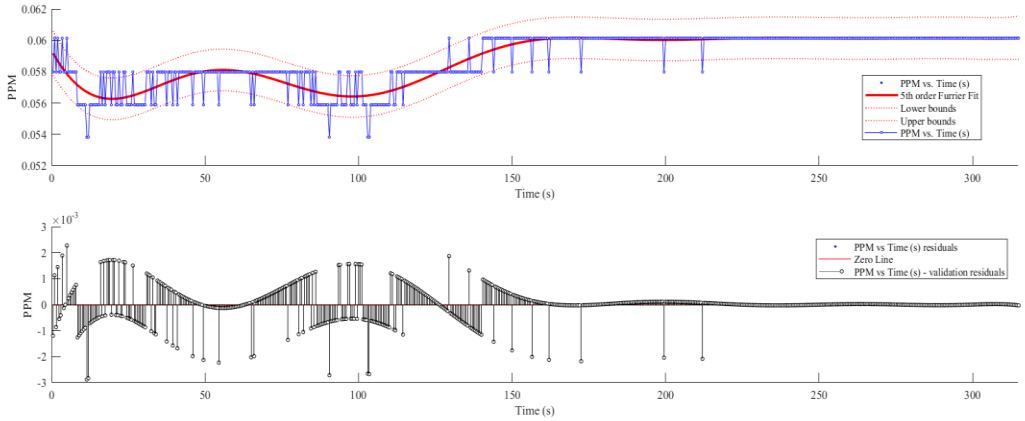


Figure-7. 5 PPM vs Time Domain. (5th order furrier transforms with 95% confidence limit)

The below Figure-7.6 & 7.7 are the same plots as Figure-7.4 & 7.5, but those value represents the Temperature and Humidity.

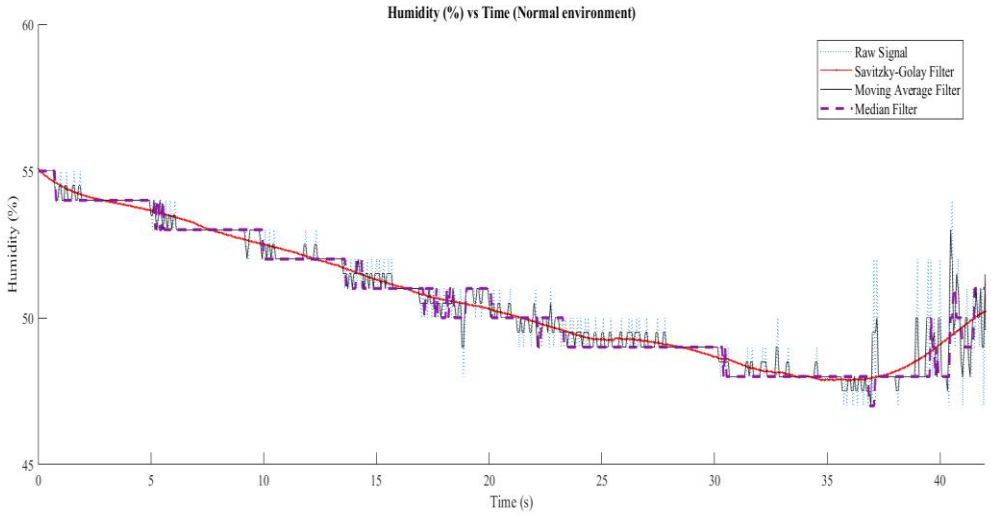


Figure-7. 6 Processed Signal (Humidity vs Time)

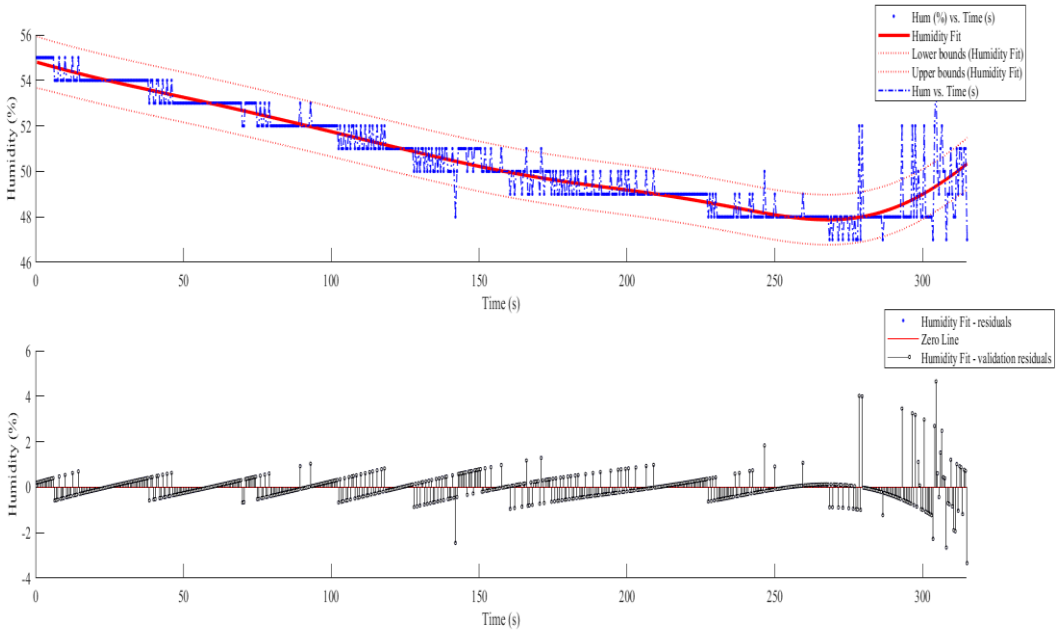


Figure-7. 7 Humidity vs Time domain graph with 8th order furrier transform using 95% of confidence limit.

It shows that an 8th order Fourier Transformation will result in a better signal conditioning.

(iii) Group delay response analysis: In signal processing, group delay is the time delay of the amplitude envelopes of the various sinusoidal components of a signal through a device under test, and is a function of frequency for each component. Phase delay, in contrast, is the time delay of the phase as opposed to the time delay of the amplitude envelope. Now from convolution theorem [46-50],

$$\text{If, } y(t) = (h * x)(t) \stackrel{\text{def}}{=} \int_{-\text{inf.}}^{+\text{inf.}} x(u)h(t-u)du \tag{7.6}$$

In frequency domain; Y(s) = H(s).X(s). Then, on simplification and using some computational techniques it can be showed that;

$$x(t) = a(t)\cos(\omega t + \theta) \tag{7.7}$$

& By assuming, $\left|\frac{d\log(a(t))}{dt}\right| \ll \omega$, the output of such an LTI (Linear time-invariant) system can be approximated as,

$$y(t) = |H(i\omega)|a(t-\tau_g)\cos(\omega(t-\tau_\phi) + \theta) \tag{7.8}$$

& they can be computed from the phase shift ϕ by, $\tau_g(\omega) = -\frac{d\phi(\omega)}{d\omega}$ and $\tau_\phi(\omega) = -\frac{\phi(\omega)}{\omega}$. From equation-(16), it can be shown that the group delay, τ_g, and phase delay, τ_ϕ, are frequency-dependent. To understand the convolution process of the LTI system, let the notation $\{x(u-\tau); u\}$ represent the function $x(u-\tau)$ with variable u *and* constant τ. Then a continuous-time system transforms an input function (x) into an output function (y) and in general, every value of the output can depend on every value of the input. This concept is represented by;

$$y(t) \stackrel{\text{def}}{=} O_t(x) \tag{7.9}$$

Where, O_t is the transformation operator for time (t). In a typical system, y(t) depends most heavily on the values of x that occurred near the time t. So, for a linear system;

$$O_t\left\{\int_{-\text{inf.}}^{+\text{inf.}} c_\tau . x_\tau(u)d\tau;\ u\right\} = \int_{-\text{inf.}}^{+\text{inf.}} c_\tau . y_\tau(t)d\tau;\ u \tag{7.10}$$

& the time-invariance requirement is;

$$O_t\{x(u-\tau); u\} = y(t-\tau) \stackrel{\text{def}}{=} O_{t-\tau}\{x\} \tag{7.11}$$

From, eqn-10, the impulse response can be noted as;

$$h(t) \stackrel{\text{def}}{=} O_t\{\delta u;\ u\} \tag{7.12}$$

From, equations-(10) to (12), using Amplitude vs samples; Figure-7.8 to Figure-7.10, can be drawn. Figure-7.8 is the group delay response plot for gas sensor which shows that between 11.3-12.2 kHz of sampling frequency the delay remains constant.

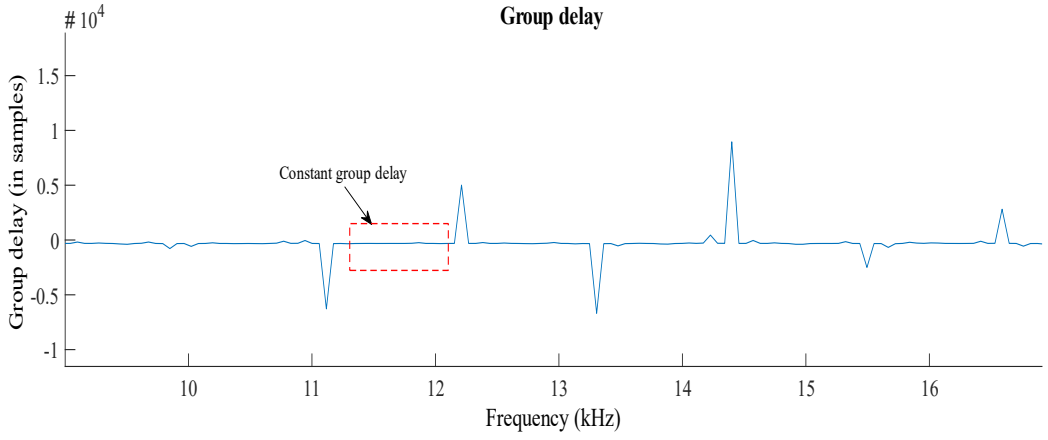


Figure-7. 8 Group delay response

Figure-7.9 represents the phase delay (with respect to samples), it shows also that 11.3-12.2 kHz sampling is the most effective one for the controller.

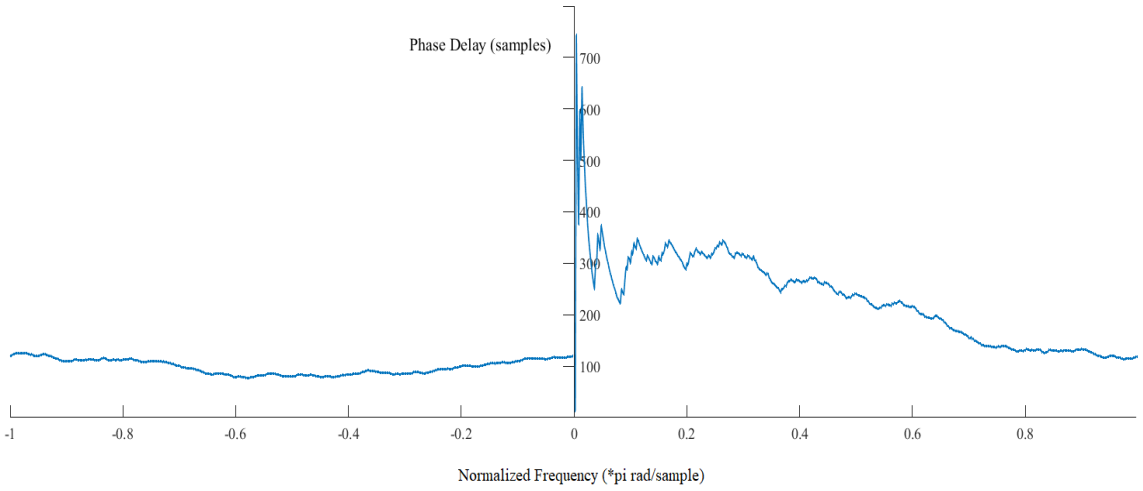


Figure-7. 9 Phase Delay Samples

Figure-7.10, denotes the pole zero plot for the system, which shows that this system has some noise but ultimately it is stable. Figure-7.11 is the round-off noise power spectrum plot for the system. It shows some unnatural noise in the system.

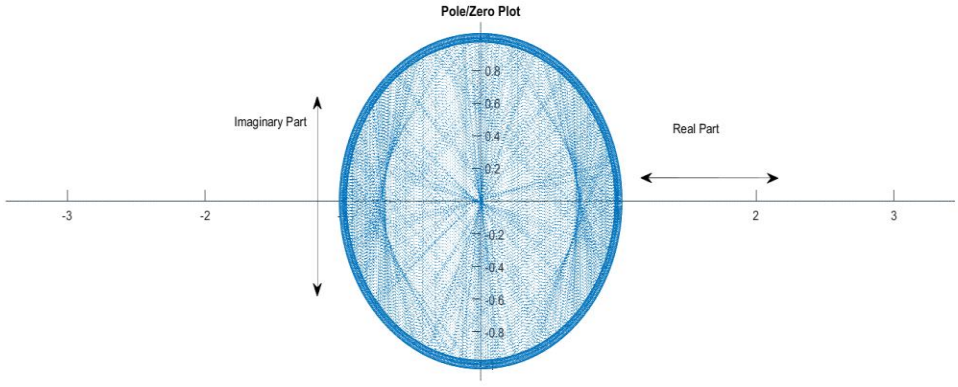


Figure-7. 10 Pole Zero Plot.

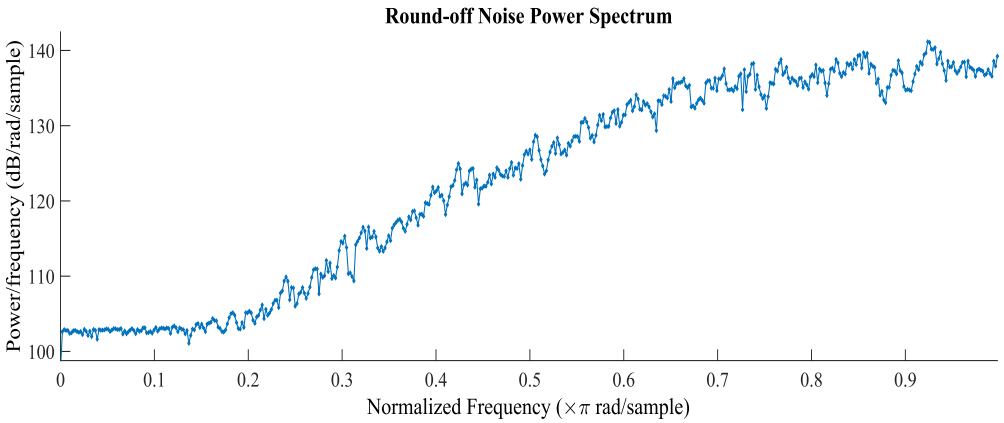


Figure-7. 11 Round-off noise Power Spectrum.

Figure-7.12 shows the actual & experimental Sensitivity vs Concentration plot, from where it can be said that the calibration was 90-95% accurate, which is good.

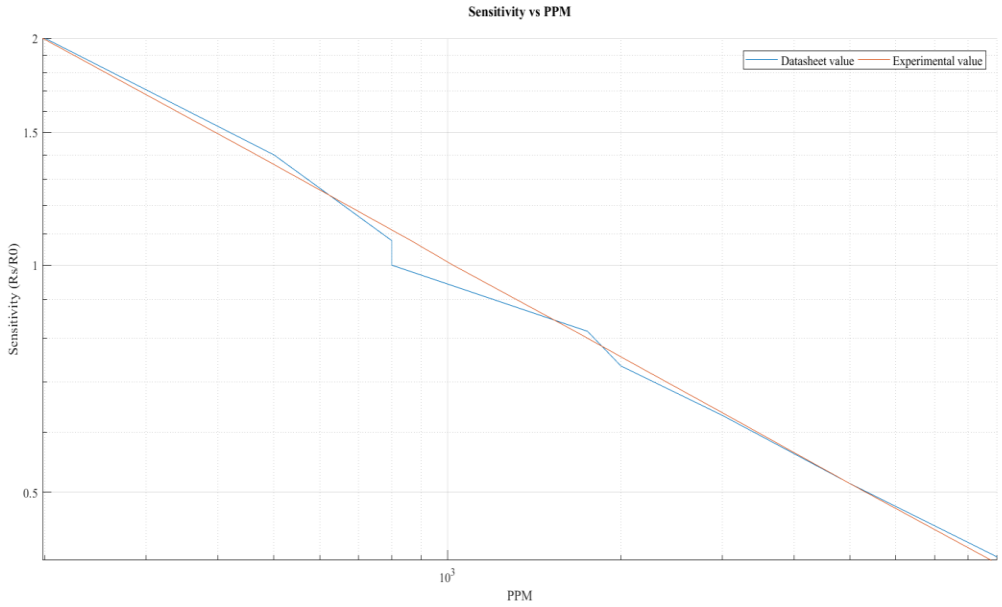


Figure-7. 12 Sensitivity vs PPM graph. (Log-Log plot)

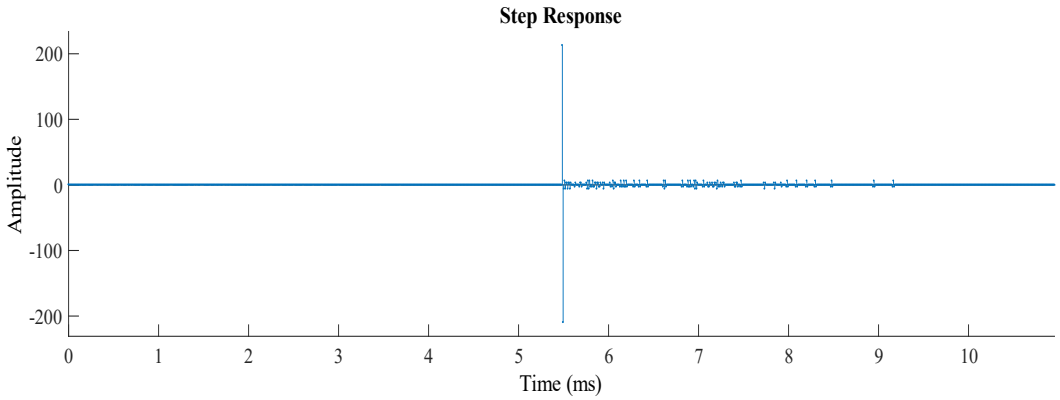


Figure-7. 13 Step response.

From, Figure-7.13, the step response can be shown where it is seen that initially the system runs fine but after a certain period it shows some abrasions but the used filter automatically fixes the errors. The below, Figure-7.14 shows the time domain and frequency domain analysis plot for the system.

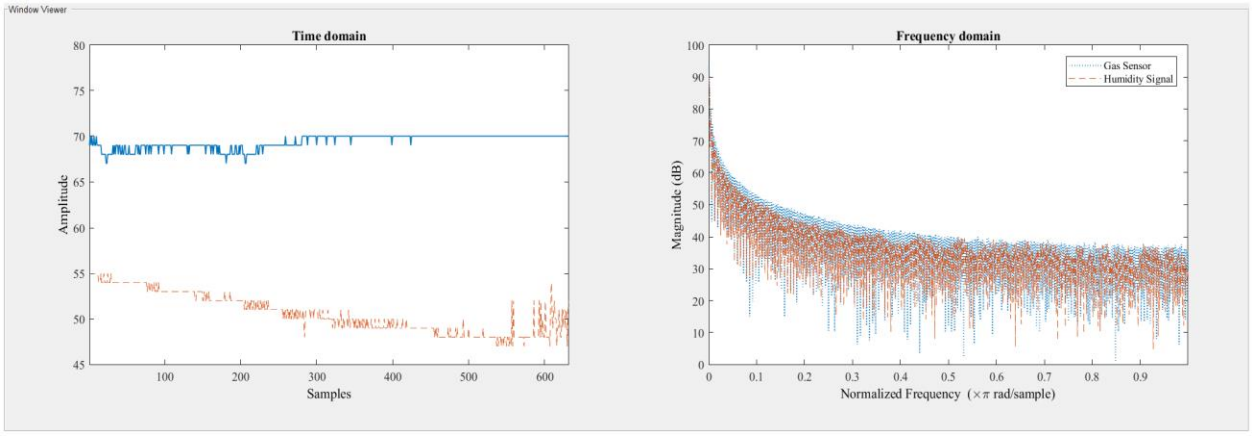


Figure-7. 14 Time domain and Frequency domain graph for the system.

The transfer function of the system can be estimated by using Welch Transfer function distribution theory, which is plotted on Figure-7.15. The Figure-7.16 shows the experimental relation between supplied voltage and gas concentration.

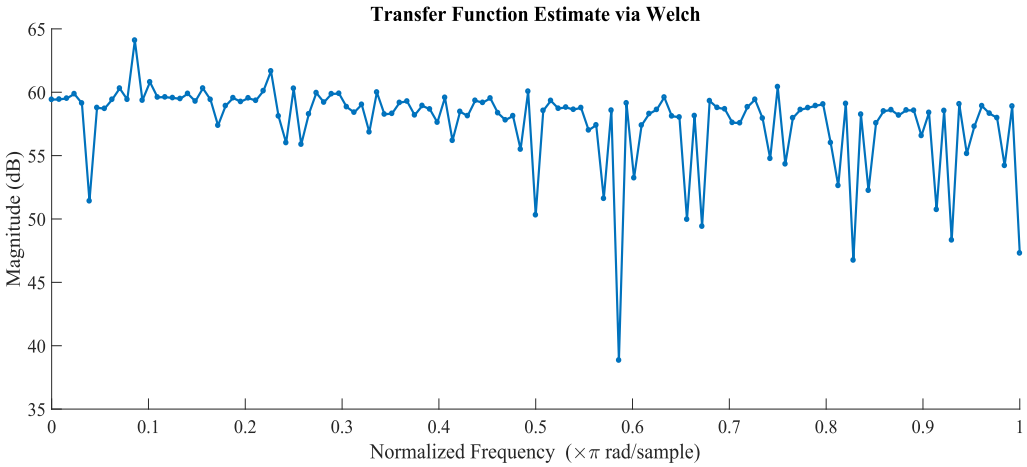


Figure-7. 15 Transfer function estimation via Welch distribution.

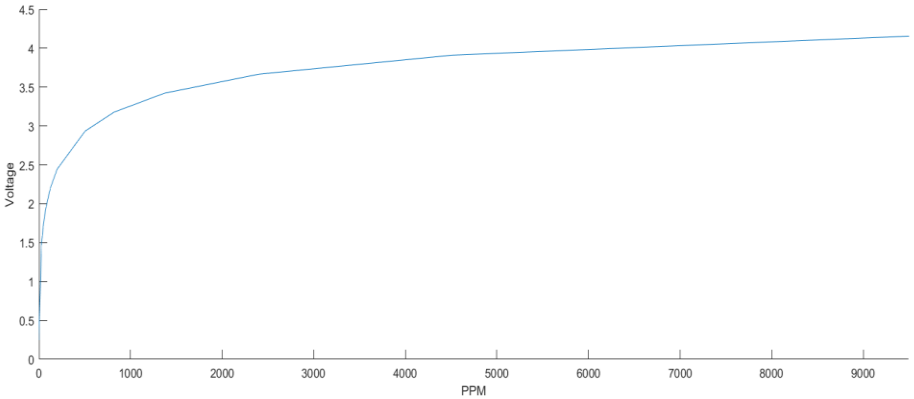


Figure-7. 16 Voltage vs PPM graph.

(iv) Power Spectrum Density Analysis (PSE or PSD): PSE is most important application area in Digital Signal Processing. There are mainly two types of power spectrum estimation (PSE) method: Parametric and nonparametric. Parametric or non-classical methods an analyzed process is place by an appropriate model with known spectrum. Non-parametric do not make any assumption on the data generating process. It is start by estimating autocorrelation sequence from a given data. The power spectrum then is estimated via FT of an estimated autocorrelation sequence. Window function is a mathematical function that is zero valued outside of some chosen period. When another unction or waveform or data sequence is multiplied by a window function, the product is also zero-valued outside the period; all that is left is the part where they overlap the

observation through the window. It is simple to apply and understand. In this method the frequency of a filter, HD(w) and the corresponding impulse response, hD(n), are related by the inverse Fourier transform [44, 50-53]:

$$h_D(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_D(w)e^{iwn}dw \tag{7.13}$$

Where HD(w) is frequency response of a filter and hD(n) is corresponding impulse response. The subscript D is used to make a difference between the ideal and practical responses. Here, HD(w) can be obtained from hD(n) by evaluating the inverse Fourier transform. The truncation of hD(n) to a length M-1 is equivalent to multiplying hD(n)by a rectangular window [2] defined as:

$$w(n) = \begin{cases} 1, & n = 0,1,2,\ldots..(M-1) \\ 0, & \text{otherwise} \end{cases} \tag{7.14}$$

And unit impulse response will be:

$$h(n) = h_D(n).w(n) \tag{7.15}$$

$$h(n) = \begin{cases} h_D(n), & n = 0,1,2,\ldots..(M-1) \\ 0, & \text{otherwise} \end{cases} \tag{7.16}$$

Frequency domain function in representation of window function is,

$$W(w) = \sum_{n=0}^{M-1} w(n).e^{-jwn} \tag{7.17}$$

The individuality of it play a significant role in establishment of the resulting frequency response of the finite impulse response filter obtained by truncation hD(n) to length M. The undesirable effects are best alleviated by the use of window that do not contain abrupt discontinuities in their time domain characteristics and have likewise low side lobes in their frequency domain characteristics. For the same value of M for both Rectangular and Hamming window or other windows, the width of the main lobe is also wider for these windows compared to the rectangular window. The Fourier transform of rectangular window [44-50, 54]:

$$W(w) = \sum_{n=0}^{M-1} e^{-jwn} \tag{7.18}$$

On simplification equation-(26) reaches,

$$W(w) = \sum_{n=0}^{M-1} e^{-jwn} = e^{-jw.\frac{(M-1)}{2}} . \frac{\sin(wM/2)}{\sin(w/2)} \tag{7.19}$$

The Hamming window function in time domain decrease more gently towards zero on either side and in frequency domain, the amplitude of the main lobes is wider approximately double than that of rectangular window, but side lobes are lesser relative to the main lobe about 40 dB down the main lobes, compared with 14 dB for the rectangular window. Hamming window lead to a filter with wider transition width but higher stopband attenuation.

$$Y(n) = x(n).w(n) \tag{7.20}$$

Where Y(n) is output signal x(n) is input signal, and w(n) [1] is window function.

$$w(n) = 0.54 + 0.46\cos\left(\frac{2\pi}{N}\right), \quad \text{where} \begin{cases} -\frac{N}{2} \le n \le \frac{N}{2} \ (N = \text{Even number}) \\ -(N-2) \le n \le \frac{(N-1)}{2} (N = \text{odd number}) \end{cases} \tag{7.21}$$

Transition width, $\Delta f = \frac{3.3}{N}$. Where N is filter length, and Δf is normalized transition width Window Length, L = N+1. In Power Spectrum Estimation is most important application area in Digital Signal Welch method have two basic modification to the normal analytic method. These are allowed the data length to overlap. The data segment can be represented as,

$$x(n) = x(n + iD) \begin{cases} n = 0,1,2, \dots \dots . (M-1) \\ i = 0,1,2, \dots \dots \dots . (L-1) \end{cases} \tag{7.22}$$

Where iD is the starting point for the ith sequence. If D = M, the segment does not overlap and the L of data sequence is identical to the data segment of Bartlett method. The second change in Welch method is to window the data segments prior to computing the periodogram.

$$\breve{P}_{xx}^{i}(f) = \frac{1}{MU} \left| \sum_{n=0}^{M-1} x(n).w(n).e^{-j.2\pi fn} \right|, \text{where } i = 1,2, \dots \dots \dots . . (L-1) \tag{7.23}$$

Where U is a normalization factor for the power.

$$U = \frac{1}{M} \sum_{n=0}^{M-1} w^2(n) \tag{7.24}$$

The Welch power spectrum estimate is the average of modified periodogram, is

$$\breve{P}_{xx}^{w} = \frac{1}{L}\sum_{i=0}^{L-1} \breve{P}_{xx}^{i}(f) \tag{7.25}$$

Mean value of Welch estimate [55],

$$E\left[\breve{P}_{xx}^{w}(f)\right] = \frac{1}{L}\sum_{i=0}^{L-1} E.\,\breve{P}_{xx}^{i}(f) \tag{7.26}$$

The resolution of estimated power estimation is determined by the spectral resolution of each segment which is of length L, it is window dependent. From equations-(21) to (34), Figure-7.17 & Figure-7.18 can be drawn.

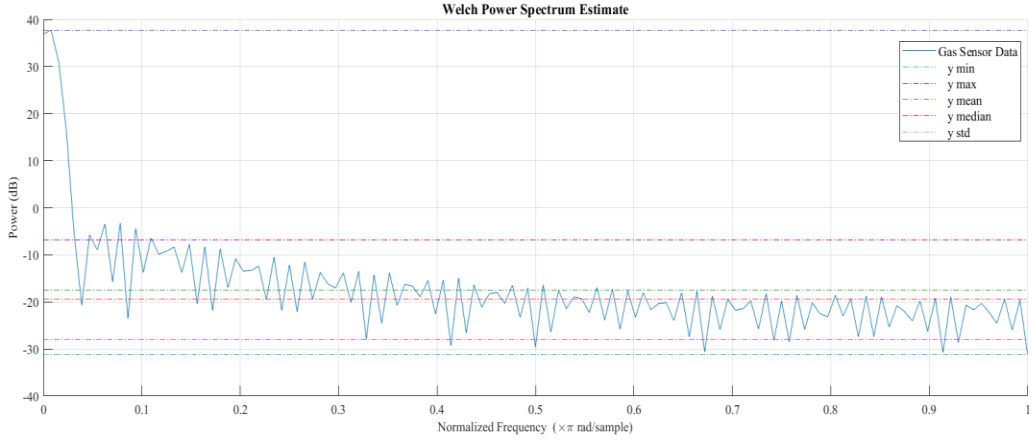


Figure-7. 17 Welch power spectrum estimate for Gas sensor.

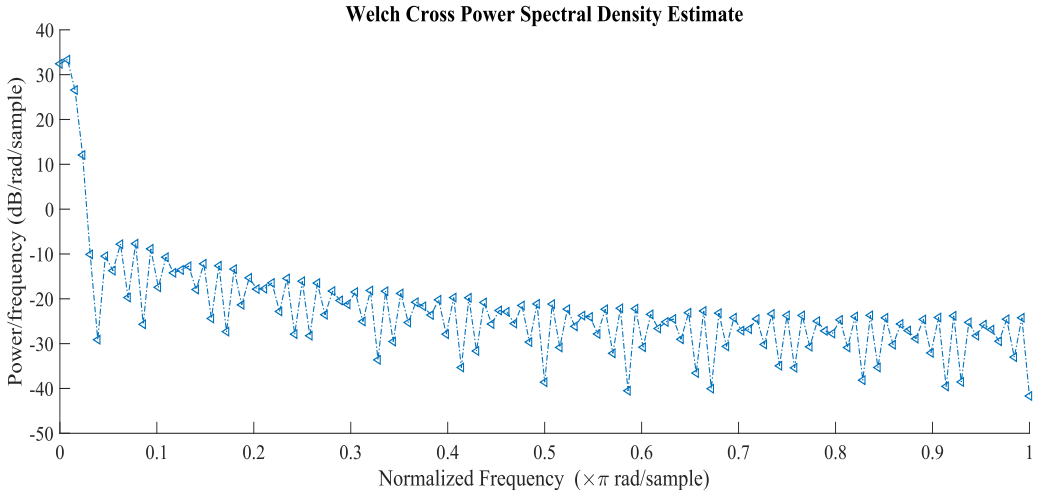


Figure-7. 18 Welch Cross power density estimate.

7.2 Feedback Control for the Controller

In this system, the reference value is set at 2000 ppm of smoke concentrations. Whenever the controller gets a reading higher than 2000 ppm from the gas sensor then it automatically activates the actuators and the rise in concentration gets mitigated. Thus, the error in the system is eliminated very soon after it takes place in the plant. Fig. shows the schematic of the used feedback method of control for the proposed system.

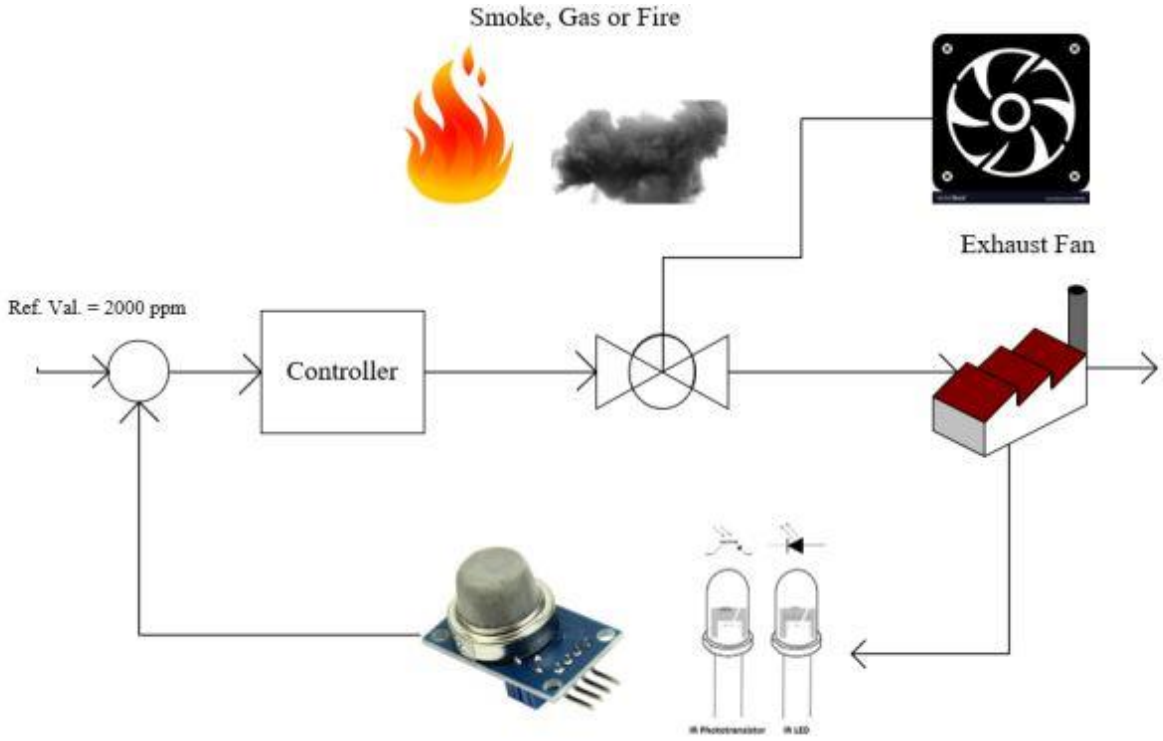


Figure-7. 19 Schematic for feedback control system

7.3 Feedforward Control for the Controller

For the feedforward control, when any abruptions take place in any sensors, then it is very necessary to mitigate the error before it takes place in the plant. Thus, manual wireless switching is introduced here to control the abruptions. From the computer, the actuation signal buttons are pressed from the webpage which in regards sends a signal to the controller via a routing media (for this experiment a 300 mbps router is used). When the signal is received by the controller, a swift actuator switching takes place to mitigate the abruptions. Fig. shows the graphical representation of the proposed feedforward control system for the experimental setup.

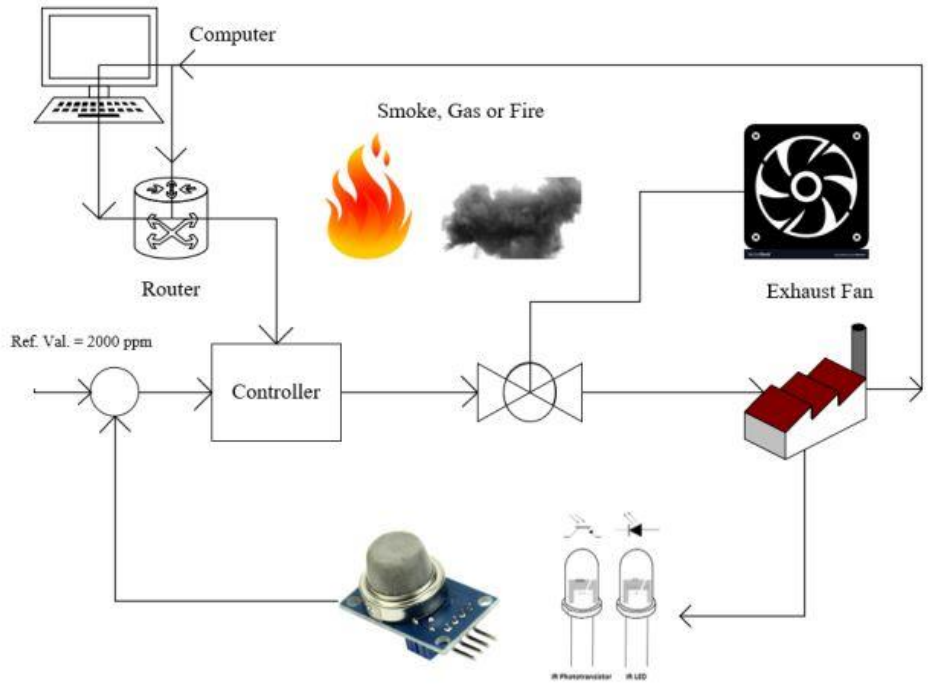


Figure-7. 20 Schematic for feedforward control system

7.4 Data Security

7.4.1 Cryptographic nonce

A nonce is a random or semi-random number that is generated for a specific use, typically related to cryptographic communication or information technology. The term itself stands for "number used once" or "number once" and is most commonly referred to specifically as a cryptographic nonce. Data was sent by adding an arbitrary value and when data was received by a controller it subtracted the arbitrary value by itself. The value was randomly selected by the user. If any attack will occur during data communication the attacker will get the data with the added arbitrary value.

7.5 Conclusions

The fabricated system run successfully and 20 trials were performed to measure its performance. It showed no errors in those 20 trials which took almost 3 hours to execute. The control system took almost 1 seconds (approximately) to perform necessary commands when signals came from the host server. From analyzed signals the impulse period and the group delay period seemed quite perfect if the Baud Rate (Sampling Frequency) tuning ranges from 11 to 12.6 kHz. In this dissertation, 11.52 kHz Sampling Frequency was used and the power density spectrum of gas

sensor was quite fine though the Signal to Noise ratio was a little high. This can be overcome by using a 20-pF ceramic disk capacitor as it can work as a low pass filter. The overall system efficiency was about 95%, which is quite good for a robust controlling operation.

7.6 Future Work

This framework can be extended for the development of building and city automation and Industry 4.0. Also, the WSN based control scheme can be extended for controlling and monitoring of remote devices such as robots, different types of rovers etc. Though there are numerous future scopes of the project, some of this are listed below

(1) The web page further can be developed to provide real time chart for better visualization of sensor node data in an android APK app for the Android users.

(2) Voice commands technology can be implemented for controlling Actuators.
(3) Adaptive fuzzy logic can be implemented for automatic controlling of AC, water pump and other equipment's. It can also be used to pre-detect the probability of the fire.

References

[1] Amin, M. Miftakul, M. Azel Aji Nugratama, Andino Maseleno, Miftachul Huda, and Kamarul Azmi Jasmi. "Design of cigarette disposal blower and automatic freshner using mq-5 sensor based on atmega 8535 microcontroller." *International Journal of Engineering & Technology* 7, no. 3 (2018): 1108-1113.

[2] Sinha, Nitin, Korrapati Eswari Pujitha, and John Sahaya Rani Alex. "Xively based sensing and monitoring system for IoT." In *Computer Communication and Informatics (ICCCI), 2015 International Conference on*, pp. 1-6. IEEE, 2015.

[3] Keshamoni, Kumar, and Sabbani Hemanth. "Smart Gas Level Monitoring, Booking & Gas Leakage Detector over IoT." In *Advance Computing Conference (IACC), 2017 IEEE 7th International*, pp. 330-332. IEEE, 2017.

[4] Kim, Jung-Yoon, Chao-Hsien Chu, and Sang-Moon Shin. "ISSAQ: An integrated sensing systems for real-time indoor air quality monitoring." *IEEE Sensors Journal* 14, no. 12 (2014): 4230-4244.

[5] Dutta, Joy, and Sarbani Roy. "IoT-fog-cloud based architecture for smart city: Prototype of a smart building." In *Cloud Computing, Data Science & Engineering-Confluence, 2017 7th International Conference on*, pp. 237-242. IEEE, 2017.

[6] Yadav, Vasudev, Akhilesh Shukla, Sofiya Bandra, Vipin Kumar, Ubais Ansari, and Suraj Khanna. "A Review On Iot Based Hazardous Gas Leakage Detection & Controlling System Using Microcontroller & Gsm Module." *Journal of VLSI Design and Signal Processing* 3, no. 1 (2017).

[7] Sharma, Manaswi, Diksha Tripathi, Narendra Pratap Yadav, and Parth Rastogi. "Gas Leakage Detection and Prevention Kit Provision with IoT." *Gas* 5, no. 02 (2018).

[8] Keshamoni, Kumar, and Sabbani Hemanth. "Smart Gas Level Monitoring, Booking & Gas Leakage Detector over IoT." In *Advance Computing Conference (IACC), 2017 IEEE 7th International*, pp. 330-332. IEEE, 2017.

[9] Koduru, Suresh, Prasad Reddy PVGD, and Preethi Padala. "Integrated disaster management and smart insurance using cloud and internet of things." *International Journal of Engineering & Technology* 7, no. 2.6 (2018): 241-246.

[10] Mahalingam, A., R. T. Naayagi, and N. E. Mastorakis. "Design and implementation of an economic gas leakage detector." *Recent Researches in Applications of Electrical and Computer Engineering* (2012): 20-24.

[11] Alekseev, Vladimir V., Vera S. Konovalova, and Ekaterina N. Sedunova. "Information-measurement and control system “smart house” as object of practice-oriented training of master's degree “instrumentation technology”." In *" Quality Management, Transport and Information Security, Information Technologies"(IT&QM&IS), 2017 International Conference*, pp. 612-615. IEEE, 2017.

[12] Sabilla, Shoffi Izza, Riyanarto Sarno, and Joko Siswantoro. "Estimating gas concentration using artificial neural network for electronic nose." *Procedia Computer Science* 124 (2017): 181-188.
[13] Lee, Duk-Dong, and Dae-Sik Lee. "Environmental gas sensors." *IEEE Sensors Journal* 1, no. 3 (2001): 214-224.
[14] Moshayedi, Ata Jahangir, M. V. Kukade, and Damayanti Gharpure. "Electronic-nose (E-nose) for recognition of Cardamom, Nutmeg and Clove oil odor." (2014).
[15] Kwok, Jessica, and Yu Sun. "A Smart IoT-Based Irrigation System with Automated Plant Recognition using Deep Learning." In *Proceedings of the 10th International Conference on Computer Modeling and Simulation*, pp. 87-91. ACM, 2018.
[16] Karim, A. B., A. Z. Hassan, and M. M. Akanda. "Monitoring food storage humidity and temperature data using IoT." *MOJ Food Process Technol* 6, no. 4 (2018): 400-404.
[17] Amuzuvi, Christian Kwaku. "Design of an Electronic Wireless Communication-Based Real Time Alert Liquefied Petroleum Gas Leakage Detector." *Ghana Journal of Technology* 2, no. 1 (2017): 7-19.
[18] Sethi, Pallavi, and Smruti R. Sarangi. "Internet of things: architectures, protocols, and applications." *Journal of Electrical and Computer Engineering* 2017 (2017).
[19] Tsang, Y. P., K. L. Choy, C. H. Wu, G. T. S. Ho, H. Y. Lam, and P. S. Koo. "An IoT-based cargo monitoring system for enhancing operational effectiveness under a cold chain environment." *International Journal of Engineering Business Management* 9 (2017): 1847979017749063.
[20] Joe Mari J. Maja, James Robbins. Controlling irrigation in a container nursery using IoT. AIMS Agriculture and Food, 2018, 3(3): 205-215. doi: 10.3934/agrfood.2018.3.205
[21] Dewi, L., and Y. Somantri. "Wireless Sensor Network on LPG Gas Leak Detection and Automatic Gas Regulator System Using Arduino." In *IOP Conference Series: Materials Science and Engineering*, vol. 384, no. 1, p. 012064. IOP Publishing, 2018.
[22] Gubbi, Jayavardhana, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. "Internet of Things (IoT): A vision, architectural elements, and future directions." *Future generation computer systems* 29, no. 7 (2013): 1645-1660.
[23] Lee, In, and Kyoochun Lee. "The Internet of Things (IoT): Applications, investments, and challenges for enterprises." *Business Horizons* 58, no. 4 (2015): 431-440.
[24] Zhu, Qian, Ruicong Wang, Qi Chen, Yan Liu, and Weijun Qin. "Iot gateway: Bridgingwireless sensor networks into internet of things." In *Embedded and Ubiquitous Computing (EUC), 2010 IEEE/IFIP 8th International Conference on*, pp. 347-352. Ieee, 2010.
[25] Chiang, Mung, and Tao Zhang. "Fog and IoT: An overview of research opportunities." *IEEE Internet of Things Journal* 3, no. 6 (2016): 854-864.
[26] Tan, L. T., W. Y. Lim, M. Farihin, M. Jefry, M. Thway, and F. Lin. "Design of Smart Cooling System for Solar Panel on Performance Efficiency with Internet of Things (IoT)."

In *Hyperspectral Imaging and Sounding of the Environment*, pp. JT2A-7. Optical Society of America, 2018.

[27] Jokar, Paria, and Victor CM Leung. "Intrusion detection and prevention for zigbee-based home area networks in smart grids." *IEEE Transactions on Smart Grid* 9, no. 3 (2018): 1800-1811.

[28] Collotta, Mario, Giovanni Pau, Timothy Talty, and Ozan K. Tonguz. "Bluetooth 5: A Concrete Step Forward toward the IoT." *IEEE Communications Magazine* 56, no. 7 (2018): 125-131.

[29] Sindhuri, V. Chaitanya, L. Suresh Babu, and N. Lakshminarayana. "TO FUNCTIONING OF ECG OBSERVING DEVICE BY USING IOT TECHNOLOGY." *IJITR* 6, no. 2 (2018): 7873-7875.

[30] Ercan, Ali Ö., M. Oğuz Sunay, and Ian F. Akyildiz. "Rf energy harvesting and transfer for spectrum sharing cellular iot communications in 5g systems." *IEEE Transactions on Mobile Computing* 17, no. 7 (2018): 1680-1694.

[31] Chatterjee, Anjan. "Artificial Intelligence based IoT Automation: Controlling devices with Google and Facebook." *Artificial Intelligence* 5, no. 04 (2018).

[32] Liu, Yimin, Wenyan Liu, and Xiangyang Luo. "Survey on the Indoor Localization Technique of Wi-Fi Access Points." *International Journal of Digital Crime and Forensics (IJDCF)*10, no. 3 (2018): 27-42.

[33] Cheikhrouhou, Omar, Ghulam M Bhatti, and Roobaea Alroobaea. "A Hybrid DV-Hop Algorithm Using RSSI for Localization in Large-Scale Wireless Sensor Networks." *Sensors* 18, no. 5 (2018): 1469.

[34] Khelifi, Fekher, Abbas Bradai, Kamal Singh, and Mohamed Atri. "Localization and Energy-Efficient Data Routing for Unmanned Aerial Vehicles: Fuzzy-Logic-Based Approach." *IEEE Communications Magazine* 56, no. 4 (2018): 129-133.

[35] Choosaksakunwiboon, Shanatip, Chawin Terawong, Suppakorn Suttisirikul, Isara Anantavrasilp, Surapa Thiemjarus, Sodsai Wisadsud, and Kamol Kaemarungsi. "A Pre-processing Technique for BLE-based Indoor Localization." In *Proceedings of the 12th International Convention on Rehabilitation Engineering and Assistive Technology*, pp. 241-244. Singapore Therapeutic, Assistive & Rehabilitative Technologies (START) Centre, 2018.

[36] NodeMCU Documentation, (Web); Retrieved from: http://modtronix.com.au/product/nodemcu-v1/ (Last Accessed: 10/11/2018)

[37] MQ-6 Product Figure, (Modern, Electronics Ltd.); Retrieved from: https://modernelectronics.com.pk/product/mq-6-lpg-gas-sensor/ (Last Accessed: 10/11/2018)

[38] MQ-6 Sensor Datasheet, (Sparkfun Inc., USA); Retrieved from: https://www.sparkfun.com/datasheets/Sensors/Biometric/MQ-6.pdf (Last Accessed: 10/11/2018)

[39] Solenoid Valve Figure, (MGA Controls Ltd.); Retrieved from: https://www.mgacontrols.com/product/elektrogas-auto-reset-gas-solenoid-valve-vmr932/ (Last Accessed: 10/11/2018)

[40] Arduino Official Software- Open Source, (Arduino Corp., Italy); Retrieved from: https://arduino.cc (Last Accessed: 10/11/2018)

[41] Brandt, Anders. "A signal processing framework for operational modal analysis in time and frequency domain." *Mechanical Systems and Signal Processing* 115 (2019): 380-393.

[42] Young, Eric D., Kyle Strom, Ashley F. Tsue, Joseph L. Usset, Seth MacPherson, John T. McGuire, and Danny R. Welch. "Automated quantitative image analysis for ex vivo metastasis assays reveals differing lung composition requirements for metastasis suppression by KISS1." *Clinical & experimental metastasis* (2018): 1-10.

[43] Mariani, Sara, Leila Tarokh, Ina Djonlagic, Brian E. Cade, Michael G. Morrical, Kristine Yaffe, Katie L. Stone et al. "Evaluation of an automated pipeline for large-scale EEG spectral analysis: the National Sleep Research Resource." *Sleep medicine* 47 (2018): 126-136.

[44] Zawawi, TNS Tengku, A. R. Abdullah, W. T. Jin, R. Sudirman, and N. M. Saad. "Electromyography Signal Analysis Using Time and Frequency Domain for Health Screening System Task." *International Journal of Human and Technology Interaction (IJHaTI)* 2, no. 1 (2018): 35-44.

[45] Gres, Szymon, Palle Andersen, C. Hoen, and Lars Damkilde. "Orthogonal projection-based harmonic signal removal for operational modal analysis." In *Structural Health Monitoring, Photogrammetry & DIC, Volume 6*, pp. 9-21. Springer, Cham, 2019.

[46] Regalia, Phillip. *Adaptive IIR filtering in signal processing and control*. Routledge, 2018.

[47] Boashash, Boualem, Abdeldjalil Aïssa-El-Bey, and Mohammad F. Al-Sa'd. "Multisensor Time–Frequency Signal Processing MATLAB package: An analysis tool for multichannel non-stationary data." *SoftwareX* (2018).

[48] Cohen, Aaron E. "Automated HDL signal processing deployment performance from high level MATLAB specification for an unmanned aerial vehicle (UAV)." In *Computing and Communication Workshop and Conference (CCWC), 2018 IEEE 8th Annual*, pp. 900-905. IEEE, 2018.

[49] Van Drongelen, Wim. *Signal processing for neuroscientists*. Academic press, 2018.

[50] Anchal, A., A. Jain, S. Ahmad, and Pradeep Kumar Krishnamurthy. "Nonlinearity Mitigation in Coherent Optical Communication Systems: All-Optical and Digital Signal Processing Approaches." In *Selected Topics in Photonics*, pp. 41-51. Springer, Singapore, 2018.

[51] Seichter, Felicia, Erhan Tuetuencue, Tamina Hagemann, Josef Vogt, Ulrich Wachter, Michael Groeger, Sandra Kress, Peter Radermacher, and Boris Mizaikoff. "Online monitoring of carbon dioxide and oxygen in exhaled mouse breath via substrate-integrated hollow waveguide-Fourier transform infrared-luminescence spectroscopy." *Journal of breath research* (2018).

[52] Sauer, Uta, H. Borsdorf, P. Dietrich, A. Liebscher, I. Möller, S. Martens, F. Möller, S. Schlömer, and C. Schütze. "Application of open-path Fourier transform infrared spectroscopy for atmospheric monitoring of a CO 2 back-production experiment at the Ketzin pilot site (Germany)." *Environmental monitoring and assessment* 190, no. 3 (2018): 114.

[53] Hijazi, Mohamad, Mathilde Rieu, Valérie Stambouli, Guy Tournier, Jean-Paul Viricelle, and Christophe Pijolat. "Ambient temperature selective ammonia gas sensor based on SnO2-APTES modifications." *Sensors and Actuators B: Chemical*256 (2018): 440-447.

[54] Movlaee, K., P. Periasamy, T. Krishnakumar, M. R. Ganjali, S. G. Leonardi, G. Neri, Murthy Chavali, Prem Felix Siril, and V. P. Devarajan. "Microwave-assisted synthesis and characterization of WOx nanostructures for gas sensor application." *Journal of Alloys and Compounds* 762 (2018): 745-753.

[55] Goutham, Solleti, Kishor Kumar Sadasivuni, Devarai Santhosh Kumar, and Kalagadda Venkateswara Rao. "Flexible ultra-sensitive and resistive NO 2 gas sensor based on nanostructured Zn (x) Fe (1− x) 2 O 4." *RSC Advances* 8, no. 6 (2018): 3243-3249.

[56] Arif, Abul, Peng Yao, Fulvia Terenzi, Jie Jia, Partho Sarothi Ray, and Paul L. Fox. "The GAIT translational control system." *Wiley Interdisciplinary Reviews: RNA* 9, no. 2 (2018): e1441.

[57] Tischler, Mark B. "System identification methods for aircraft flight control development and validation." In *Advances In Aircraft Flight Control*, pp. 35-69. Routledge, 2018.

[58] Sloo, David, Nicholas Unger Webb, Evan Jarman Fisher, Yoky Matsuoka, Anthony Fadell, and Matthew Rogers. "Smart-home control system providing HVAC system dependent responses to hazard detection events." U.S. Patent 9,905,122, issued February 27, 2018.

[59] Losey, Dylan P., Craig G. McDonald, Edoardo Battaglia, and Marcia K. O'Malley. "A Review of Intent Detection, Arbitration, and Communication Aspects of Shared Control for Physical Human–Robot Interaction." *Applied Mechanics Reviews* 70, no. 1 (2018): 010804.

[60] Yilmaz, Unal, Ali Kircay, and Selim Borekci. "PV system fuzzy logic MPPT method and PI control as a charge controller." *Renewable and Sustainable Energy Reviews* 81 (2018): 994-1001.

[61] He, Wei, Tingting Meng, Deqing Huang, and Xuefang Li. "Adaptive boundary iterative learning control for an Euler–Bernoulli beam system with input constraint." *IEEE transactions on neural networks and learning systems* 29, no. 5 (2018): 1539-1549.

[62] Walczak, Steven. "Artificial neural networks." In *Advanced Methodologies and Technologies in Artificial Intelligence, Computer Simulation, and Human-Computer Interaction*, pp. 40-53. IGI Global, 2019.

[63] Comer, Douglas E. *The Internet book: everything you need to know about computer networking and how the Internet works*. Chapman and Hall/CRC, 2018.

[64] Tang, Fengxiao, Bomin Mao, Zubair Md Fadlullah, Nei Kato, Osamu Akashi, Takeru Inoue, and Kimihiro Mizutani. "On removing routing protocol from future wireless networks: A real-time deep learning approach for intelligent traffic control." *IEEE Wireless Communications* 25, no. 1 (2018): 154-160.

[65] Kral, Jan, Tomas Gothans, Roman Marsalek, and Michal Harvanek. "Digital Predistorter with Real-Valued Feedback Employing Forward Model Estimation." In *2018 25th International Conference on Telecommunications (ICT)*, pp. 471-475. IEEE, 2018.

[66] Bekraoui, Nabyl, Lamia Boussaidi, Georges Cazorla, and Luc Léger. "Oxygen Uptake, Heart Rate, and Lactate Responses for Continuous Forward Running and Stop-and-Go Running With and Without Directional Changes." *Journal of strength and conditioning research* (2018).

[67] Kawato, Mitsuo. "Feedback-error-learning neural network for supervised motor learning." In *Advanced neural computers*, pp. 365-372. 1990.

[68] Selemon, Lynn D., and Patricia S. Goldman-Rakic. "Common cortical and subcortical targets of the dorsolateral prefrontal and posterior parietal cortices in the rhesus monkey: evidence for a distributed neural network subserving spatially guided behavior." *Journal of Neuroscience* 8, no. 11 (1988): 4049-4068.

[69] Vandana, K., & Baweja, C. (2016). Simmarpreet, and S. Chopra,"Influence of Temperature and Humidity on the Output Resistance Ratio of the MQ-135 Sensor,". *Int. J. Adv. Res. Comput. Sci. Softw. Eng*, *6*(4), 423-429.

[70] https://photovoltaic-software.com/principle-ressources/solar-radiation-databases

Appendix

Login Code:

```
<?php include('server.php') ?>
<!DOCTYPE html>
<html>
<head>

	<title>Log In</title>
	<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>

	<div class="header">
	<h2>Log In</h2>
	</div>

	<form method="post" action="login.php">

		<?php include('errors.php'); ?>

		<div class="input-group">
			<label>Username</label>
			<input type="text" name="username" >
		</div>
		<div class="input-group">
			<label>Password</label>
			<input type="password" name="password">
		</div>

		<div class="input-group">
			<button type="submit" class="btn btn-success"
name="login_user">Login</button>
		</div>
		<p>
			Don&#39;t have an account? <a href="register.php">Sign up</a>
		</p>
	</form>
```

```
</body>
</html>
```

Get Status:

```
<?php
   include_once('server.php');

   // Check connection
   if ($conn->connect_error) {
      die("Connection failed: " . $conn->connect_error);
   }
   date_default_timezone_set("Asia/Karachi");
   $device_name    =   $_GET['device_name'];

   $sql          =   "SELECT device_status FROM devices_status WHERE
device_name='$device_name'";
   $result       =   $conn->query($sql);
   if ($result->num_rows > 0) {
      while($row = $result->fetch_assoc())
      {
         echo $row['device_status'];
      }
   } else {
      echo "Error:" . $sql . "<br>" . $conn->error;
   }

   $conn->close();
?>
```

Live data:

```
<!DOCTYPE html>
<html>
<head>
<style type="text/css">
.table_titles {
  border: 1px solid #dddddd;
  text-align: left;
  padding: 8px;
}
.table_titles {
  border: 1px solid #dddddd;
  text-align: left;
```

```
  padding: 8px;
}
.form-inline{
        width: 47%;
        height: auto;
        color: black;
        background-color: whitesmoke;
        text-align: center;
        padding: 10px;
    }
.submit{
    padding: 5px;
            font-size: 13px;
            color: white;
            background: #ADD8E6;
            border: none;
            border-radius: 5px;
    }
.header {
            width: 45%;
    height: 15px;
            margin: 10px auto 0px;
            color: white;
            background: #5F9EA0;
            text-align: center;
            border: 1px solid #B0C4DE;
            border-bottom: none;
            border-radius: 10px 10px 0px 0px;
            padding: 20px;
}
        .table_cells_odd{
          border: 1px solid #dddddd;
  text-align: left;
  padding: 8px;
        }
        .table_cells_even{
          border: 1px solid #dddddd;
            text-align: left;
            padding: 8px;
            background-color: #dddddd;
        }
table {
border: 1px solid #dddddd;
```

```
}
body { font-family: "Trebuchet MS", Courier; }
</style>
<body>
<?php
$url=$_SERVER['REQUEST_URI'];
header("Refresh: 5; URL=$url"); // Refresh the webpage every 5 seconds
?>

<center>
   <a href="index.php">Go Back to Previous Page</a>
   <div class="header">
                  <font size="5px">Data From Wireless Sensor Network</font>
         </div>
<form method="post" action="search.php" class="form-inline">
  <div>
    <label>Search By Date</label>
    <input type="date" name="date_from" value="<?php echo date('Y-m-d'); ?>"  />
  <button type="submit" class="submit" name="search">Search</button>
     </div>
</form>
<table border="0" cellspacing="0" cellpadding="4">
<tr>
<td class="table_titles">ID</td>
<td class="table_titles">Temperature(in °C)</td>
<td class="table_titles">Humidity(in %)</td>
<td class="table_titles">LPG(in ppm)</td>
<td class="table_titles">Time</td>
<td class="table_titles">Date</td>
</tr>
<?php
include('connection.php');
   $s = "select * from temps";
$result = mysqli_query($con,$s);
$count = mysqli_num_rows($result);
$p=$count/15;
$pn=ceil($p);
$page="";
$page=$_GET['page'];
   if($page=="" || $page=="1")
   {
      $page1=0;
      $page1=0;
```

```
    }
    else{
        $page1=($page*15)-15;
    }
$rec=mysqli_query($con,"SELECT * FROM temps ORDER BY id DESC limit $page1,15");
while($row = mysqli_fetch_array($rec))
{
$oddrow = true;
while($row = mysqli_fetch_array($rec))
{
if ($oddrow)
{
$css_class=' class="table_cells_odd"';
}
else
{
$css_class=' class="table_cells_even"';
}
$oddrow = !$oddrow;


echo "<tr>";
echo "<td '.$css_class.'>"  .$row['id'] . "</td>";
echo "<td '.$css_class.'>"  .$row['temp'] . "</td>";
echo "<td '.$css_class.'>"  .$row['humidity'] . "</td>";
echo "<td '.$css_class.'>"  .$row['lpg'] . "</td>";
echo "<td '.$css_class.'>"  .$row['time'] . "</td>";
echo "<td '.$css_class.'>"  .$row['dated'] . "</td>";
echo "</tr>";
}
}

?>
</table></center>
 <?php   for($b=1; $b<=$pn; $b++)
{
   ?> <a href="livedata.php?page=<?php echo $b; ?>" style="text-decoration:none" ><?php echo
$b.""; ?></a><?php
}
    // Close the connection
mysqli_close($con);
    ?>
</body>
```

```
</html>
```

Status Page:

```
<!DOCTYPE html>
<html lang="en">
   <head>
                  <meta charset="UTF-8" />
      <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
                  <meta name="viewport" content="width=device-width, initial-scale=1.0">
      <title>Controlling Page</title>

      <link rel="stylesheet" type="text/css" href="css/style.css" />

   </head>
   <body>

   <?php
   include_once('server.php');




   if(isset($_GET['device_status']))
   {

      $status  =  $_GET['device_status'];
      if($status==1)
      {
       $update_status  =  0;
      }
      elseif($status==0)
      {
       $update_status  =  1;
      }
      $query  =  "UPDATE devices_status SET device_status='$update_status' WHERE
device_name='home_sensor'";
      $conn->query($query);
   }
```

```
   $get_status_sql          =   "SELECT device_status FROM devices_status WHERE
device_name='home_sensor'";
   $result        =   $conn->query($get_status_sql);

   if ($result->num_rows > 0) {
   while($row = $result->fetch_assoc())
   {
      $current_status = $row['device_status'];

      if($current_status == 1)
      {
         $checked    =   'checked';
      }
      else
      {
         $checked    =   '';
      }
   }
   }

   ?>



      <div class="container">


                        <section class="main">
                              <h1 style="text-align: center; padding: 30px; font-size: 35px; font-
family: fantasy;">Control Sensor Data with PHP and MySQL</h1>

            <div class="switch demo3">

               <input onClick="location.href='<?php echo $_SERVER['PHP_SELF'];
?>?device_status=<?php echo $current_status; ?>'" type="checkbox" value="<?php echo
$current_status;?>" <?php echo $checked;?> />


                                    <label><i></i></label>
```

```
                                    </div>
<h4 style="text-align: center;">&copyAll rights reserved by  <a
href="http://mte.ruet.ac.bd/" target="_blank">Dept. of Mechatronics Engineering,
RUET</a></h4>

                            </section>

        </div>

    </body>
</html>
```

Server :

```
<?php
    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "temperature";
    // Create connection
     $conn      =      mysqli_connect($servername, $username,$password);
    $connection = mysqli_select_db($conn,$dbname);
?>
```

Index :

```
<?php
        session_start();

        if (!isset($_SESSION['username'])) {
                $_SESSION['msg'] = "You must log in first";
                header('location:home.php');
        }
        if (isset($_GET['logout'])) {
                session_destroy();
                unset($_SESSION['username']);
                header("location: home.php");
        }

?>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
```

```
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Home</title>
        <link rel="stylesheet" type="text/css" href="style.css">
   <style type="text/css">
      .table_cells_odd{
       border: 1px solid #dddddd;
  text-align: left;
  padding: 8px;
      }
      .table_cells_even{
       border: 1px solid #dddddd;
         text-align: left;
         padding: 8px;
         background-color: #dddddd;
      }
   </style>

</head>
<body>
        <div class="header">
                <h2>Home Page</h2>
        </div>

   <!-- notification message -->
                <?php if (isset($_SESSION['success'])) : ?>
                <?php endif ?>

                <!-- logged in user information -->
                <?php  if (isset($_SESSION['username'])) : ?>
                        <div class="content">
            <p>Welcome <strong><?php echo $_SESSION['username']; ?></strong></p>
                        <p> <a href="index.php?logout='1'" style="color: red;">logout</a> </p>
                <?php endif ?>
   </div>
 <div class="content">
    <p style="color: black;">Now you can control your device status from <a
href="http://localhost/device_controll/status_page.php" target="_blank" style="color:
red;">here</a> to prevent auto collection of sensors data. If you turn it off, controller data will
not be saved to the database.</p>
</div>
<?php
   include_once('server2.php');
   //Light_Code
```

```
    if(isset($_GET['light_status']))
   {

     $status  =  $_GET['light_status'];
     if($status==1)
     {
      $update_status  =  1;
     }
     elseif($status==0)
     {
      $update_status  =  0;
     }
     $query  =  "UPDATE light_status SET light_status='$update_status' WHERE
name='light_sensor'";
     $conn->query($query);
   }


 ?>
 <div class="content">
<button type="button" ><a href="index.php?light_status=1"> RELAY ON</a></button>
<button type="button" ><a href="index.php?light_status=0"> RELAY OFF</a></button><br>

  Sensor value is : <span id="ADCValue">0</span><br>
   LED State is : <span id="LEDState">NA</span>

    </div>

<center>  <table border="0" cellspacing="0" cellpadding="4">
<tr>
<td class="table_titles">Name</td>
<td class="table_titles">Highest Value</td>
<td class="table_titles">Lowest Value</td>
<td class="table_titles">Average Value</td>
</tr>
    <?php

require_once('connection.php');
$s = "select * from temps";
$result = mysqli_query($con,$s);
$count = mysqli_num_rows($result);
echo "Total Number of Stored Data: " .$count. " " ;
```

```
 $query="SELECT max(temp) as 'hightemp' FROM temps";
$res=mysqli_query($con, $query);
$data= mysqli_fetch_array($res);
$field1a= $data["hightemp"];
   $query="SELECT min(temp) as 'lowtemp' FROM temps";
$res=mysqli_query($con, $query);
$data= mysqli_fetch_array($res);
$field2a= $data['lowtemp'];
   $query="SELECT avg(temp) as 'avgtemp' FROM temps";
$res=mysqli_query($con, $query);
$data= mysqli_fetch_array($res);
$field3a= $data['avgtemp'];

         ?><tr>
            <td class="table_cells_even">  Temperature </td>
            <td class="table_cells_even"><?php echo "    $field1a" ?></td>
            <td class="table_cells_even"> <?php echo "   $field2a" ?></td>
            <td class="table_cells_even"><?php echo "   $field3a" ?></td>
            </tr>
<?php

   $query="SELECT max(humidity) as 'highhum' FROM temps";
$res=mysqli_query($con, $query);
$data= mysqli_fetch_array($res);
$field1b= $data["highhum"];
   $query="SELECT min(humidity) as 'lowhum' FROM temps";
$res=mysqli_query($con, $query);
$data= mysqli_fetch_array($res);
$field2b= $data["lowhum"];
   $query="SELECT avg(humidity) as 'avghum' FROM temps";
$res=mysqli_query($con, $query);
$data= mysqli_fetch_array($res);
$field3b= $data["avghum"];
       ?>  <tr>
              <td class="table_cells_odd">  Humidity </td>
              <td class="table_cells_odd"><?php echo "   $field1b" ?></td>
              <td class="table_cells_odd"><?php echo "   $field2b" ?> </td>
              <td class="table_cells_odd"> <?php echo "   $field3b" ?></td>
              </tr>"
<?php

   $query="SELECT max(lpg) as 'highlpg' FROM temps";
```

```
   $res=mysqli_query($con, $query);
   $data= mysqli_fetch_array($res);
   $field1c= $data["highlpg"];
      $query="SELECT min(lpg) as 'lowlpg' FROM temps";
   $res=mysqli_query($con, $query);
   $data= mysqli_fetch_array($res);
   $field2c= $data["lowlpg"];
      $query="SELECT avg(lpg) as 'avglpg' FROM temps";
   $res=mysqli_query($con, $query);
   $data= mysqli_fetch_array($res);
   $field3c= $data["avglpg"];

            ?><tr>
               <td class="table_cells_even">  LPG </td>
               <td class="table_cells_even"><?php echo "   $field1c" ?></td>
               <td class="table_cells_even"><?php echo "   $field2c" ?></td>
               <td class="table_cells_even"><?php echo "   $field3c" ?></td>
               </tr>

   </table>
</center>
<center><a href="livedata.php?page=1" style"color: blue;"> See Live Data Here</a></center>

<div id="header-content"><br><br><br><br> &copyAll rights reserved by  <a
href="http://mte.ruet.ac.bd/" target="_blank">Dept. of Mechatronics Engineering, RUET</a>
    </div>
</body>
</html>
```

CSS Codes:

```
html {
   background: #e6e9e9;
   background-image: linear-gradient(270deg, rgb(230, 233, 233) 0%, rgb(216, 221, 221) 100%);
   -webkit-font-smoothing: antialiased;
}

body {
   background: #fff;
   box-shadow: 0 0 2px rgba(0, 0, 0, 0.06);
   color: #545454;
   font-family: "Helvetica Neue", Helvetica, Arial, sans-serif;
   font-size: 16px;
   line-height: 1.5;
```

```
    margin: 0 auto;
    max-width: 800px;
    padding: 2em 2em 4em;
}

h1, h2, h3, h4, h5, h6 {
    color: #222;
    font-weight: 600;
    line-height: 1.3;
}

h2 {
    margin-top: 1.3em;
}

a {
    color: #0083e8;
}

b, strong {
    font-weight: 600;
}

samp {
    display: none;
}

img {
    animation: colorize 2s cubic-bezier(0, 0, .78, .36) 1;
    background: transparent;
    border: 10px solid rgba(0, 0, 0, 0.12);
    border-radius: 4px;
    display: block;
    margin: 1.3em auto;
    max-width: 95%;
}

@keyframes colorize {
    0% {
        -webkit-filter: grayscale(100%);
        filter: grayscale(100%);
    }
    100% {
```

```
    -webkit-filter: grayscale(0%);
    filter: grayscale(0%);
  }
}
```

Index HTML:

```
<!DOCTYPE html>
<html>

  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>GETTING STARTED WITH BRACKETS</title>
    <meta name="description" content="An interactive getting started guide for Brackets.">
    <link rel="stylesheet" href="main.css">
  </head>
  <body>

    <h1>GETTING STARTED WITH BRACKETS</h1>
    <h2>This is your guide!</h2>

    <!--
      MADE WITH <3 AND JAVASCRIPT
    -->

    <p>
      Welcome to Brackets, a modern open-source code editor that understands web design. It's
a lightweight,
      yet powerful, code editor that blends visual tools into the editor so you get the right
amount of help
      when you want it.
    </p>

    <!--
      WHAT IS BRACKETS?
    -->
    <p>
      <em>Brackets is a different type of editor.</em>
      Brackets has some unique features like Quick Edit, Live Preview and others that you may
not find in other
      editors. Brackets is written in JavaScript, HTML and CSS. That means that most of you
using Brackets
```

have the skills necessary to modify and extend the editor. In fact, we use Brackets every day to build
Brackets. To learn more about how to use the key features, read on.

<!--
GET STARTED WITH YOUR OWN FILES
-->

<h3>Projects in Brackets</h3>

In order to edit your own code using Brackets, you can just open the folder containing your files.
Brackets treats the currently open folder as a "project"; features like Code Hints, Live Preview and
Quick Edit only use files within the currently open folder.

<samp>
Once you're ready to get out of this sample project and edit your own code, you can use the dropdown
in the left sidebar to switch folders. Right now, the dropdown says "Getting Started" - that's the
folder containing the file you're looking at right now. Click on the dropdown and choose "Open Folder…"
to open your own folder.
You can also use the dropdown later to switch back to folders you've opened previously, including this
sample project.
</samp>

<!--
THE RELATIONSHIP BETWEEN HTML, CSS AND JAVASCRIPT
-->
<h3>Quick Edit for CSS and JavaScript</h3>

No more switching between documents and losing your context. When editing HTML, use the
<kbd>Cmd/Ctrl + E</kbd> shortcut to open a quick inline editor that displays all the related CSS.
Make a tweak to your CSS, hit <kbd>ESC</kbd> and you're back to editing HTML, or just leave the

CSS rules open and they'll become part of your HTML editor. If you hit <kbd>ESC</kbd> outside of
a quick inline editor, they'll all collapse. Quick Edit will also find rules defined in LESS and
SCSS files, including nested rules.

<samp>
Want to see it in action? Place your cursor on the <!-- <samp> --> tag above and press
<kbd>Cmd/Ctrl + E</kbd>. You should see a CSS quick editor appear above, showing the CSS rule that
applies to it. Quick Edit works in class and id attributes as well. You can use it with your
LESS and SCSS files also.

You can create new rules the same way. Click in one of the <!-- <p> --> tags above and press
<kbd>Cmd/Ctrl + E</kbd>. There are no rules for it right now, but you can click the New Rule
button to add a new rule for <!-- <p> -->.
</samp>

You can use the same shortcut to edit other things as well - like functions in JavaScript,
colors, and animation timing functions - and we're adding more and more all the time.

For now inline editors cannot be nested, so you can only use Quick Edit while the cursor
is in a "full size" editor.

<!--
LIVE PREVIEW
-->
<h3>Preview HTML and CSS changes live in the browser</h3>

You know that "save/reload dance" we've been doing for years? The one where you make changes in
your editor, hit save, switch to the browser and then refresh to finally see the result?
With Brackets, you don't have to do that dance.

```
    </p>
    <p>
      Brackets will open a <em>live connection</em> to your local browser and push HTML
and CSS updates as you
      type! You might already be doing something like this today with browser-based tools, but
with Brackets
      there is no need to copy and paste the final code back into the editor. Your code runs in
the
      browser, but lives in your editor!
    </p>

    <h3>Live Highlight HTML elements and CSS rules</h3>
    <p>
      Brackets makes it easy to see how your changes in HTML and CSS will affect the page.
When your cursor
      is on a CSS rule, Brackets will highlight all affected elements in the browser. Similarly,
when editing
      an HTML file, Brackets will highlight the corresponding HTML elements in the browser.
    </p>

    <samp>
      If you have Google Chrome installed, you can try this out yourself. Click on the lightning
bolt
      icon in the top right corner of your Brackets window or hit <kbd>Cmd/Ctrl + Alt +
P</kbd>. When
      Live Preview is enabled on an HTML document, all linked CSS documents can be edited
in real-time.
      The icon will change from gray to gold when Brackets establishes a connection to your
browser.

      Now, place your cursor on the <!-- <img> --> tag above. Notice the blue highlight that
appears
      around the image in Chrome. Next, use <kbd>Cmd/Ctrl + E</kbd> to open up the
defined CSS rules.
      Try changing the size of the border from 10px to 20px or change the background
      color from "transparent" to "hotpink". If you have Brackets and your browser running
side-by-side, you
      will see your changes instantly reflected in your browser. Cool, right?
    </samp>

    <p class="note">
      Today, Brackets only supports Live Preview for HTML and CSS. However, in the
current version, changes to
```

JavaScript files are automatically reloaded when you save. We are currently working on Live Preview
support for JavaScript. Live previews are also only possible with Google Chrome, but we hope
to bring this functionality to all major browsers in the future.

<h3>Quick View</h3>

For those of us who haven't yet memorized the color equivalents for HEX or RGB values, Brackets makes
it quick and easy to see exactly what color is being used. In either CSS or HTML, simply hover over any
color value or gradient and Brackets will display a preview of that color/gradient automatically. The
same goes for images: simply hover over the image link in the Brackets editor and it will display a
thumbnail preview of that image.

<samp>
To try out Quick View for yourself, place your cursor on the <!-- <body> --> tag at the top of this
document and press <kbd>Cmd/Ctrl + E</kbd> to open a CSS quick editor. Now simply hover over any of the
color values within the CSS. You can also see it in action on gradients by opening a CSS quick editor
on the <!-- <html> --> tag and hovering over any of the background image values. To try out the image
preview, place your cursor over the screenshot image included earlier in this document.
</samp>

<h3>Need something else? Try an extension!</h3>

In addition to all the goodness that's built into Brackets, our large and growing community of
extension developers has built hundreds of extensions that add useful functionality. If there's
something you need that Brackets doesn't offer, more than likely someone has built an extension for
it. To browse or search the list of available extensions, choose File > Extension
Manager… and click on the "Available" tab. When you find an extension you want, just click

```
        the "Install" button next to it.
    </p>

    <!--
        LET US KNOW WHAT YOU THINK
    -->
    <h2>Get involved</h2>
    <p>
        Brackets is an open-source project. Web developers from around the world are
contributing to build
        a better code editor. Many more are building extensions that expand the capabilities of
Brackets.
        Let us know what you think, share your ideas or contribute directly to the project.
    </p>
    <ul>
        <li><a href="http://brackets.io">Brackets.io</a></li>
        <li><a href="http://blog.brackets.io">Brackets Team Blog</a></li>
        <li><a href="https://github.com/adobe/brackets">Brackets on GitHub</a></li>
        <li><a href="https://brackets-registry.aboutweb.com">Brackets Extension
Registry</a></li>
        <li><a href="https://github.com/adobe/brackets/wiki">Brackets Wiki</a></li>
        <li><a href="https://groups.google.com/forum/#!forum/brackets-dev">Brackets
Developer Mailing List</a></li>
        <li><a href="https://twitter.com/brackets">@brackets on Twitter</a></li>
        <li>Chat with Brackets developers on IRC in <a
href="http://webchat.freenode.net/?channels=brackets&uio=d4">#brackets on
Freenode</a></li>
    </ul>

  </body>
</html>
<!--
```

Arduino Code:

For Fetching data into server:

```
#include <ESP8266HTTPClient.h>
#include <ESP8266WiFi.h>
#include <dht.h>
#define LED D4
#define MQ_PIN A0
// Setting up variables
int delay_time  = 3000; // Its mean 3 seconds wait to enter first record
float get_lpg;

//Access point credentials
const char* ssid        = "Asif";
const char* pwd         = "18273645";
String get_status_url   = "http://192.168.43.207/device_controll/get_status.php";
String add_data_url     = "http://192.168.43.207/device_controll/add_data.php";
String light_data_url   = "http://192.168.43.207/main/get_state.php";
WiFiServer server(80);  // open port 80 for server connection

void setup()
{
  Serial.begin(115200); //initialise the serial communication
  delay(20);
  WiFi.begin(ssid, pwd);


    //starting the server
    server.begin();

}
void handleLED() {
  String LEDState = "OFF";
   String t_state = get_device_status("light_sensor");
   Serial.println(t_state);
 if(t_state == "1")
 {
  digitalWrite(LED,HIGH); //LED ON
  LEDState = "ON"; //Feedback parameter
 }
 else
 {
```

```
  digitalWrite(LED,LOW); //LED OFF
  LEDState = "OFF"; //Feedback parameter
 }}

void loop(){

    String device_status  = get_device_status("home_sensor");
    delay(2000);
    WiFiClient client = server.available();
    DHT.read11(dht_apin);
    get_temperature =   DHT.temperature;
    get_humidity    =   DHT.humidity;
  float sensor_volt; //Define variable for sensor voltage
  float RS_gas; //Define variable for sensor resistance
  float ratio; //Define variable for ratio
  float sensorValue = analogRead(MQ_PIN); //Read analog values of sensor
  float m = -0.423; //Slope
  float b = 1.276; //Y-Intercept
  float R0 = 5.62; //Sensor Resistance in fresh air from previous code

  sensor_volt = sensorValue*(5.0/1023.0); //Convert analog values to voltage
  RS_gas = ((5.0*10.0)/sensor_volt)-10.0; //Get value of RS in a gas
  ratio = RS_gas/R0;  // Get ratio RS_gas/RS_air

  double ppm_log = (log10(ratio)-b)/m; //Get ppm value in linear scale according to the the ratio
value
  double ppm = pow(10, ppm_log); //Convert ppm value to log scale
  double percentage = ppm/10000; //Convert to percentage

  double get_lpg = ppm;


    if(device_status=="1")
    {
     add_device_data(get_temperature, get_humidity, get_lpg); //add sensor
    }
    else
    {
      Serial.println("Device Status is going OFF");
      }

    delay(delay_time);
```

```
}


///This is the function that will get status of device that is On or Off
void add_device_data(float get_temperature, float get_humidity,float get_lpg) // add sensor
  {

      WiFiClient client = server.available();

      HTTPClient http;
      String url =
add_data_url+"?temp="+get_temperature+"&humidity="+get_humidity+"&lpg="+get_lpg; //add
sensors
      http.begin(url);
      //GET method
      int httpCode = http.GET();
      String payload = http.getString();
      Serial.println(url);
      Serial.println(payload);
      http.end();


  }


String get_device_status(String device_name)
  {

      WiFiClient client = server.available();

      HTTPClient http;
      String url = get_status_url+"?device_name="+device_name;
      http.begin(url);
      int httpCode = http.GET();
      String payload = http.getString();
      Serial.println(url);
      Serial.println(payload);
      return payload;


  }
```

For controlling Actuator:

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>

#include "Asif.h" //Our HTML webpage contents with javascripts
#define MQ A0
int relay_1;
int relay_2;

#define LED D4  //On board LED
//SSID and Password of your WiFi router
const char* ssid = "Asif";
const char* password = "18273645";

ESP8266WebServer server(80); //Server on port 80

void handleRoot() {
 String s = MAIN_page; //Read HTML contents
 server.send(200, "text/html", s); //Send web page
}

void handleADC() {
  float MQ = analogRead(A0);
  String ADCValue = String(MQ);


 server.send(200, "text/plane", ADCValue); //Send ADC value only to client ajax request
}

void handleLED() {
 String LEDState = "OFF";
 String t_state = server.arg("LEDstate"); //Refer  xhttp.open("GET", "setLED?LEDstate="+led,
true);
 Serial.println(t_state);
 if(t_state == "1")
 {
  digitalWrite(LED,LOW); //LED ON
  LEDState = "ON"; //Feedback parameter
 }
 else
 {
  digitalWrite(LED,HIGH); //LED OFF
```

```
  LEDState = "OFF"; //Feedback parameter
 }

 server.send(200, "text/plane", LEDState); //Send web page
}

void setup(void){
  Serial.begin(115200);

  WiFi.begin(ssid, password);     //Connect to your WiFi router
  Serial.println("");

  //Onboard LED port Direction output
  pinMode(LED,OUTPUT);
  //pinMode(trigPin, OUTPUT);
  //pinMode(echoPin, INPUT);
  // Wait for connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  //If connection successful show IP address in serial monitor
  Serial.println("");
  Serial.print("Connected to ");
  Serial.println(ssid);
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());  //IP address assigned to your ESP

  server.on("/", handleRoot);      //Which routine to handle at root location. This is display page
  server.on("/setLED", handleLED);
  server.on("/readADC", handleADC);

  server.begin();                  //Start server
  Serial.println("HTTP server started");
}

void loop(void){
  server.handleClient();          //Handle client requests
}
```