Interfacing Arduino with Iot: A Water Pressure Monitoring System

A project report presented in MEng 125n – Basic Electronics

Submitted by:

Balbarino, Vic Marie L.

Basan, Jojules T.

Cerna, Risbby Arne P.

Larena, Sean Clej E.

Tabucanon, Romulo Y.

Submitted to:

Engr. Philip Caesar L. Ebit

June 2023

# Abstract

Continuous monitoring of water pressure is essential to attain maximum system performance in different applications, and with the rise of Internet of Things (IoT) technology, an Arduino-based system that gathers, processes, and transmits data from a pressure sensor to a cloud-based platform presents a real-time, cost-friendly alternative to conventional monitoring systems, enabling remote access, quick issue identification, and integration with other IoT devices for increased efficiency and reliability in water supply systems. The study aims to manufacture a portable water pressure monitoring system with IoT capabilities, which will involve designing a system with a pressure transducer sensor, incorporating Arduino Uno R3 and ESP8266 Wi-Fi Module for data logging and analysis, calibrating the system, conducting system validation, and displaying the finished monitoring device. For constructing the system, the researchers carefully considered the hardware and software development process. The researchers integrated IoT into the system, including data transmission from the system to the database, with the help of a database and an MIT application creator. By comparing the measurement from the pressure monitoring system to an analog pressure gauge, the researchers manually calibrated the water pressure monitoring system within time intervals and recorded the readings. The correlation between the information acquired by the pressure monitoring system and the analog pressure gauge was analyzed and modeled using the T-test, Kolmogorov-Smirnov Test of Normality, and Levene’s Test of Homogeneity of Variance. The values of an analog pressure gauge and a pressure transducer were compared using the statistical tools mentioned and showed that there was no discernible difference between them.

Keywords: Pressure, Internet of Things (IoT), Arduino UNO R3, Database, T-test, Linear Regression, Calibration, ESP8266 Wi-Fi Module

Table of Contents

[Abstract 2](#_heading=h.gjdgxs)

[Introduction 7](#_heading=h.30j0zll)

[Nature and Importance of the Study 7](#_heading=h.1fob9te)

[Statement of the Problem 7](#_heading=h.3znysh7)

[Significance of the Study 8](#_heading=h.2et92p0)

[Objectives of the Study 9](#_heading=h.tyjcwt)

[Scope and Limitations of the Study 9](#_heading=h.3dy6vkm)

[Definition of Terms 9](#_heading=h.1t3h5sf)

[Review of Related Literature 11](#_heading=h.4d34og8)

[Methodology 20](#_heading=h.2s8eyo1)

[Results and Discussion 27](#_heading=h.17dp8vu)

[Summary and Conclusion 36](#_heading=h.35nkun2)

[Recommendations 37](#_heading=h.1ksv4uv)

[References 38](#_heading=h.44sinio)

[Appendices 42](#_heading=h.2jxsxqh)

List of Figures

Figure 1. System Hardware Design 20

Figure 2. Prototype Breadboard Wiring Diagram 21

Figure 3. Analog Pressure Gauge and Transducer Set-up 25

Figure 4. Completed Hardware System 27

Figure 5. Real-time Database 34

Figure 6. Application Interface 34

List of Tables

*Table 1. Significant difference between the Analog Pressure Gauge and the Pressure Transducer* 29

Table 2. Kolmogorov-Smirnov Test of Normality for Analog Pressure Gauge and Pressure Transducer 31

Table 3. The significant difference of variance between the Analog Pressure Gauge and Pressure Transducer 33

List of Appendices

Appendix A. Bill of Materials 43

Appendix B. Arduino Uno R3 Code 44

Appendix C. ESP8266 Wi-Fi Module Code 46

Appendix D. System Block Diagram 48

Appendix E. MIT App Inventor Code Blocks 49

Appendix F.Trial No.1 Data Recordings from Analog Pressure Gauge and Pressure Transducer 50

Appendix G.Trial No.2 Data Recordings from Analog Pressure Gauge and Pressure Transducer 53

Appendix H. Trial No.3 Data Recordings from Analog Pressure Gauge and Pressure Transducer 56

# Introduction

## Nature and Importance of the Study

Water pressure is critical in various applications, such as water supply systems, irrigation systems, and industrial processes. In these applications, water pressure must be maintained within a specific range for optimal system performance. One of the significant factors influencing leakage is the pressure in a water distribution system [1] [2]. Water supply systems with low pressure can lead to insufficient water flow, while high water pressure can cause damage to the pipes and other components of the system. Therefore, it is essential to monitor water pressure continuously to ensure the proper functioning and efficiency of the system.

Traditionally, water pressure monitoring systems have been implemented using wired sensors and meters that require manual reading and recording of data. These systems are often expensive, time-consuming, and require frequent maintenance. With the advent of IoT technology, it is now possible to implement water pressure monitoring systems that can be remotely monitored and controlled in real-time.

Arduino is an open-source microcontroller that can easily be programmed, erased, and reprogrammed at any instant in time [3]. It is widely used in IoT-based projects due to its low cost, ease of use, and availability of sensors and modules. In the proposed system, an Arduino board collects and processes data from a pressure sensor that measures water pressure. The microcontroller then communicates with the ESP8266 Wi-Fi module, where the data is transmitted to a cloud-based platform, where its users can store and view it.

The IoT-based Arduino water pressure monitoring system has several advantages and offers an attractive alternative to traditional monitoring systems. It is cost-effective, easy to install, and provides real-time data that can quickly identify issues. The system can be accessed remotely using a smartphone or a web interface, allowing users to monitor the water pressure in real-time and receive alerts in case of abnormalities. Moreover, such systems can be integrated with other IoT devices, such as smart valves and pumps, to create a complete IoT-based water supply system.

The proposed IoT-based Arduino water pressure monitoring system can improve the efficiency and reliability of water supply systems by providing real-time monitoring and control. It can contribute to developing sustainable water management systems and pave the way for further research in IoT-based water supply systems.

## Statement of the Problem

Water pressure plays a crucial role in assuring that the water is distributed effectively and efficiently, helps in early problem detection and maintenance, assesses system performance and water quality, and ensures that the system complies with regulations. It allows for the effective management of water resources, the reduction of water loss, and the provision of a reliable and safe water supply for residential, commercial, and industrial uses.

The manual use of a water pressure measuring device like the pressure gauge can be time-consuming and labor-intensive, providing limited monitoring capabilities and insufficient data logging and analysis. The project is essential to develop an improved, easy-access water pressure monitoring system.

## Significance of the Study

This research aims to provide a portable system for monitoring water pressure using the combined capabilities of Arduino and the Internet of Things. By integrating the IoT technology, the system created will be capable of sending real-time data which can be accessed remotely, making it easier for the administrators to monitor the system’s performance. Additionally, the integration will enhance the monitoring system’s efficiency and reliability, creating more opportunities for interconnected IoT ecosystems in water supply systems. Overall, this study is significant because of its contribution to the advancement of monitoring technology by offering a cost-effective solution with an easy-access monitoring system for quick issue identification and ensuring that the water supply operates effectively.

## Objectives of the Study

Generally, this study aims to develop a portable water pressure monitoring system with the feature of Internet of Things (IoT).

Specifically, this study aims to:

1. Fabricate a water pressure monitoring system that employs a pressure transducer for measuring water pressure in pipes;
2. Interface Arduino with IoT capabilities for data logging and analysis; and
3. Test the accuracy and reliability of the monitoring system by calibration and statistical analysis;

## Scope and Limitations of the Study

This study is focused only on the design and fabrication of an IoT-based Arduino Water Pressure Monitoring System, which involves building a prototype system that consists of an Arduino board, a pressure transducer, and a Wi-Fi module and developing a server and an app that can display real-time data. This study is limited to monitoring water pressure on small-scale water distribution networks and will not address other factors affecting water quality and supply.

## Definition of Terms

* Arduino R3 – serves as the brain of the system where the code was transmitted and where the whole operation happened.
* ESP8266 – serves as the gateway of the system, its primary role was to become a bridge between the Arduino R3 and the FireBase Cloud Database.
* Transducer – a digital measuring device that measures pressures in the system.
* SD Card Module – serves as data logging for the system.
* FireBase – is the system’s Real-Time Cloud Database where the data was transmitted through ESP8266.
* API – used for authentication on the FireBase.
* Analog Signal – the signal that the transducer transmits which will be converted to a digital signal.
* IoT – the system itself where sensors, software, and other technologies connect the system to the internet.

# Review of Related Literature

In the era of technological advancement, Industry 4.0 emerged, one of which is IoT. The Internet of Things (IoT) describes the network of physical objects— things embedded with sensors, software, and other technologies to connect and exchange data with other devices and systems over the internet. These devices range from ordinary household objects to sophisticated industrial tools. This chapter provides an overview of relevant literature and studies on the IoT-Arduino Water Pressure Monitoring System. This supported recent and past reviews, articles, books, and published works to use as a foundation to support their claims further.

**Water Pressure Monitoring**

Water pressure is an essential factor in the distribution and consumption of water in households, buildings, and communities. It refers to the force or energy that pushes water through pipes and into various outlets, such as faucets, showers, toilets, and appliances. Water pressure can vary depending on several factors, such as the location, elevation, time of day, and demand. However, excessive or insufficient water pressure can cause several issues that can affect water usage quality, safety, and efficiency.

According to the American Water Works Association (AWWA), high water pressure is one of the leading causes of water main breaks and service line failures. With this, pressure monitoring is an effective strategy for reducing water loss in a water distribution system. By monitoring the pressure in the system, the amount of water that is lost through leaks, bursts, and other water distribution problems can be minimized.

Today's water infrastructure faces a slew of operational challenges. To address these issues while expanding infrastructure to accommodate new customers, utilities must invest significant time and resources in new solutions. While water utilities seek funding to upgrade, repair, and build systems for the future, research and development have progressed, resulting in the development of new technological solutions. One example is the incorporation of IoT into new technologies.

A municipal water provider supplies water to most residential areas. Although surface water—reservoirs, lakes, and rivers—makes up most of the municipal supply, many places use groundwater sources. Water is often pumped from its source to treatment facilities, where it is then stored in pressure tanks that are positioned high up along the distribution network (in certain towns, large water towers are utilized instead). Pressure is produced by the weight of the water and the height of these tanks in relation to the distribution area. The pressure increases with tank height. From the tank, the pressured water travels to the water mains that supply the neighborhood. Booster stations, which employ pumps to maintain pressure in the distribution system, may be dispersed around the area according on the local conditions. Pressure lowering stations move high-pressure water from high-pressure to low-pressure locations in areas where the pressure gets too high, keeping the system's pressure at reasonable levels.

The typical range of residential water pressure is 45 to 80 psi (pounds per square inch). The minimum pressure required by many codes is 20 psi. Anything less than that is seen as too low, and anything less than 40 psi is regarded as low. Pressures greater than 80 psi are too high. [4]

**Pressure Sensing**

Water supply networks, is the one of the most important urban or rural infrastructures, need to be monitored for the perspective of safety and optimized operation. Pressure sensors have been created that can be read optically [5]. Potential uses for optical pressure sensors in severe environments exist. They can be employed, for instance, in situations where electrical signals must be avoided, such as when gauging the pressure of flammable gases or liquids, or in environments with strong magnetic fields. Because aluminum, which is frequently used as the connecting metal, can be avoided, they can also withstand higher temperatures.

More systems and gadgets are being developed using the Micro Electronic Mechanical System (MEMS) technology as its industry has experienced remarkable growth in recent years. Some of the typical MEMS’S devices produced are pressure sensors, accelerometers, gyroscopes, micromirrors ﬂuid pumps, radio frequency devices [6], etc.

MEMS sensors can be used to measure physical parameters such as acceleration, temperature, and pressure. Electronic components can be constructed on the same chip to measure the output of the sensors, perform signal processing, and provide wireless communication. In non-invasive applications like managing the air pressure in breathing equipment and checking blood pressure, pressure sensors have been utilized in medicine for a long time. More recently, the miniaturization offered by MEMS devices has allowed usage in invasive applications like catheter tip sensors as well as for implanted devices monitoring characteristics like blood pressure and heart rate.

Making the typical package, which is constructed of solid material and has sharp edges, suitable with the biological environment is difficult for medical purposes. This can be accomplished by covering the device in wire or biocompatible plastic. The small size, low power consumption and long-term stability of MEMS devices also makes them well suited to markets such as aerospace where long life and reliability are important. They are used in a variety of applications including cabin pressure monitoring, engine control, and instruments such as altimeters and barometers.

With this as the study of Thomson, P. entitled “Remote monitoring of rural water systems: A pathway to improved performance and sustainability?” states that remote automatic monitoring of rural water systems may be emerging from its infancy but has yet to be taken up at significant scale. There is now a need for empirical evidence that this failure prediction can reduce costs by improving operational efficiencies and deliver better health outcomes by eliminating pump downtimes. Remote monitoring can validate the performance of professional service providers. With data on usage level, speed of repairs and system uptimes made public, sources of finance that may have previously been reluctant to be involved—on account of the lack of performance transparency—may now be enticed into the rural water sector. With these data readily available, performance-related contracts that incentivize sustainable service delivery over short-term infrastructure investment can become the norm. [7]

**Arduino on Water Pressure Monitoring System**

Arduino is an open-source physical computing platform for creating interactive objects that stand alone or collaborate with software on your computer. Arduino was designed for artists, designers, and others who want to incorporate physical computing into their designs without having to first become electrical engineers. The Arduino hardware and software is open source. The open-source philosophy fosters a community that shares its knowledge generously. This is great for beginners as help is often available geographically nearby and always online, at many different skill levels, and on a bewildering array of topics. Example projects are presented not just as pictures of the finished project but include instructions for making your own or as a starting point for incorporation into your derivative or related projects. The Arduino software, known as the Integrated Development Environment (IDE), is free. You can download it from www.arduino.cc. The Arduino IDE is based on the Processing language, which was developed to help artists create computer art without having to first become software engineers. The Arduino IDE can run on Windows, Macintosh, and Linux [3]

According to a study of Bruno, F., De Marchis, M., Milici, B., Saccone, D., & Traina, F. entitled “A Pressure Monitoring System for Water Distribution Networks Based on Arduino Microcontroller” states that the proposed research is mainly focused on building a new pressure acquisition system based on open-source hardware and software through Arduino microcontrollers. Hydrostatic as well as hydrodynamic experiments have been performed to calibrate and validate the pressure measurement system. The calibration procedure pointed out the ability of the proposed method to achieve a correct pressure estimation simply by adding a translation coefficient in the conversion between the supply voltage into water pressure [8].

**Application of Industry 4.0**

At the dawn of the 21st century, the world is witnessing the fourth industrial revolution and the digital transformation of the business world, which is commonly referred to as Industry 4.0. The fourth industrial revolution is a hit rather than hype [9]. The publicization of the term “Industrie 4.0” in 2011, the digital transformation necessitated by Industry 4.0 immediately captured the attention of industrialists and governments worldwide [10].

Industry technologies 4.0, smart manufacturing, smart products, big data (BD), and the Internet of Things (IoT), among others, are some of the topics of digital and automated manufacturing [11]. Industry 4.0 approaches combine the interface of knowledge areas of electrical engineering, business administration, computer science, business systems and information engineering and mechanical engineering, as well as areas of complementary knowledge [12].

Industry 4.0 is increasingly gaining attention for the reason of its sustainability impact to the world and the way it can contribute to the sustainable economic, environmental, and social development. Since the birth of the First Industrial Revolution, around 18th century, the world is dealing with the challenge of producing more goods from natural resources which are depleting and limited, just to meet with the ever-growing consumption demand while limiting negative environmental and social impact [13] [14].

In the Industry 4.0 environment, the interconnected computers, smart materials, and intelligent machines communicate with one another, interact with the environment, and eventually make decisions with minimal human involvement. Digitizing manufacturing and business processes and deploying smarter machines and devices may offer numerous advantages such as manufacturing productivity, resource efficiency, and waste reduction [15]. But in contrast with that, an increased rate of production thanks to industrial automation would be associated with higher resource and energy consumption as well as elevated pollution concerns. [13] [16]

**Integrating IoT on Arduino**

The Internet of Things (IoT) is a fast-expanding field that entails attaching physical objects to the internet so they may communicate and share data with other objects and systems. As smart gadgets, inexpensive sensors, and high-speed internet access are all widely available, the idea of IoT has attracted a lot of attention.

Internet of Things is a new term in IT Arena. The term “Internet of Things” which is also known as IoT is coined from the two words, the first word is “Internet”, and the second word is “Things”.  In 1999 British technology pioneer Kevin Ashton, co-founder of the Auto-ID Laboratory at MIT Link, invented the term "The Internet of Things" to describe a system where the Internet is connected to the physical world via ubiquitous sensors, including RFID (Radio-frequency identification) [17]. IoT was generally defined as “dynamic global network infrastructure with self-configuring capabilities based on standards and communication protocols”.

IoT is accompanied by several difficulties, including security worries, data privacy concerns, and interoperability obstacles. Several researchers have proposed solutions to address these challenges, such as the use of blockchain technology to enhance security and privacy [18] and the adoption of standardized protocols to ensure interoperability [19]

In recent years, the use of open-source technologies in sensing projects has increased for two main reasons: low component prices and the easy use of these technologies [20]. Large monitoring and telemetry projects can be developed at a low cost using open-source platforms. Because they are versatile and capable of installing new improvements to update libraries or improve the codes, they can avoid technological obsolescence.

To ensure the success of a monitoring system, it is necessary to properly store the data that is generated [21]. Currently, low-cost cloud databases (IoT platforms) are emerging, and these require low complexity in order to facilitate their manipulation among users [22]. In addition to their main function, the orderly storage of information, they provide other secondary services such as friendly data visualization, the generation of results reports, search for patterns that follow similar behaviors, data analytics, etc. [23]. Hence, it is necessary to use these tools to manage large amounts of data and information.

Alessio Botta, Walter de Donato, Valerio Persico, and Antonio Pescape provided in-depth knowledge of the integration of cloud computing and the Internet of Things. They described a wide set of applications based on IoT and Cloud Computing like healthcare, smart home and smart metering, smart city, video surveillance, and more. They explained the importance of the cloud for the IoT environment. IoT is characterized by a very high compatibility of devices, technologies, and protocols. Therefore, scalability, interoperability, reliability, efficiency, availability, and security can be very difficult to obtain. [24]

The Arduino IDE is a cross-platform tool used to write and upload programs to Arduino-compatible boards, as well as other vendor development boards using third-party cores. The ESP8266 community created an add-on for the Arduino IDE that can be used to program the ESP8266 using the Arduino IDE and its programming language. There are many other platforms suitable for programming ESP8266 NodeMCU. According to (Parihar, 2019), NodeMCU has been equipped with an ESP8266, a highly integrated chip designed to meet the demands of the new linked world. It provides a full and self-contained Wi-Fi networking solution that can either host the application or offload all Wi-Fi traffic another application processor provides networking functions. The ESP8266 NodeMCU features extensive onboard processing and storage capabilities characteristics that enable it to be integrated with sensor-specific devices via its GPIOs with minimum initial development and low loading during runtime.

In this study, researchers utilized ESP8266 as another microcontroller that will communicate from Arduino R3 and the Firebase cloud database. Using the serial library and codes ESP8266 will communicate with Arduino R3 and the data ESP8266 received from the Arduino will be sent to the FireBase cloud database. A database housed in the cloud is Real-time Database. Information is JSON-formatted data that is continuously synced to each customer that is related. Building cross-platform software apps using JavaScript, Android, and IOS SDKs, is the one factor that underlies a larger portion of your client’s demands. Real-time Database example, and as a result, getting the most recent data changes. An example of a systematic gathering of data. Databases can be stored locally on your computer or in the cloud. The purpose of this is to have a visualization of data without going to a website minimizing the complexity of the system’s architecture. In terms of complexity, the researchers did not utilize the use of GPRS as this kind of system has such additional cost for the sim card [25].

According to the study conducted by Peng et al entitled. "Design of a water environment monitoring system based on wireless sensor networks." GPRS is used to send the aggregated data to the monitoring stations and alert the user through the message using a sim card. With this, a drawback was observed as this kind of system has such additional cost for the sim card. [25]

According to the works of Vijayakumar et al entitled "The real-time monitoring of water quality in IoT environment.", online water monitoring technologies have made significant progress for source water surveillance and water plant operation. The use of their technologies has a high cost associated with the installation and calibration of a large, distributed array of monitoring sensors. The algorithm proposed on the new technology must be suitable for the area and for large systems is not suitable. [26]

From the works of Tola et al, From the evaluations which have been conducted, it is concluded that this system functioned perfectly and can be utilized to perform real-time water pressure monitoring. They also added that in order to perfect this system, further research should add some sensors to indicate pH, oxygen content, and water quantity. Therefore, the next monitoring systems developed are expected not only to be able to measure water pressure but also to indicate the quality and quantity of the distributed water. [27]

**Statistical Tool**

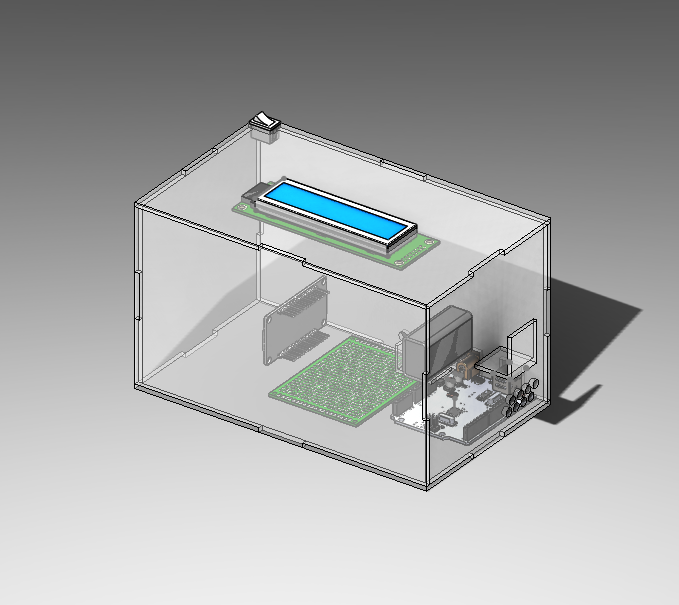
While employed by a brewery in 1908, William Sealy Gosset published a paper under the alias "student" in which he first introduced the t-test. [28] A Student's t-test, put simply, is a ratio that determines the degree to which a difference between two groups' 'means' is significant while accounting for variance or distribution. For assessing samples smaller than 30, the parametric test known as the t-test is helpful. This is because the t-test distribution and the normal distribution cannot be distinguished from one another if the sample size is more than 30.

Regression analysis is a statistical technique used to relate variables [29] Its basic aim is to build a mathematical model to relate dependent variables to independent variables. In general, a regression model will be defined as a single algebraic equation of the form [30].There are three kinds of regression models. They are namely, a) the Variable-based Degree-Day Model (VBDD), b) the Linear Regression Model, and c) the Change-Point models. They all use generalized least squares regression to determine the model coefficients [31].

# Methodology

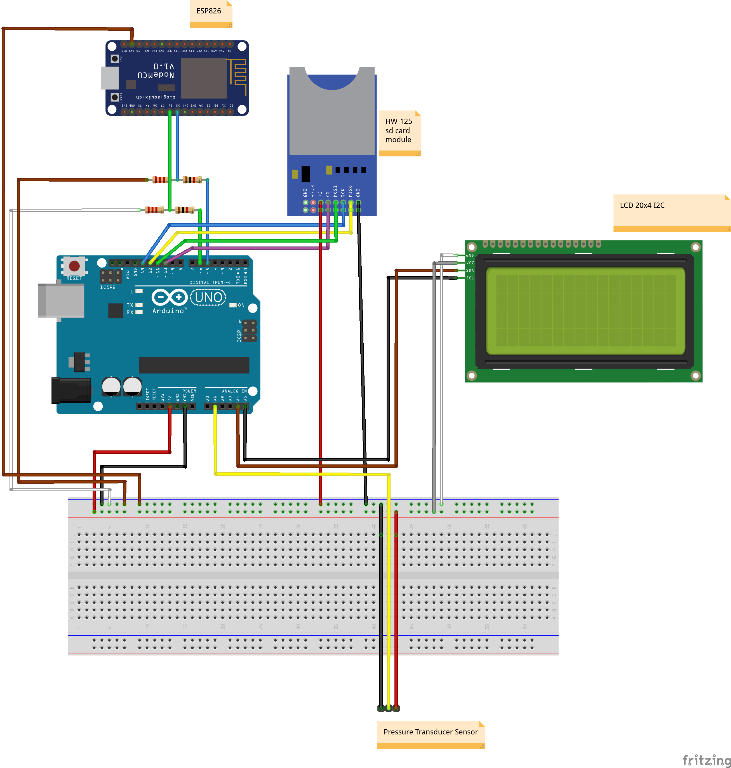
*a. Hardware System Configuration*

The IOT-interfaced Arduino water pressure monitoring system is designed to monitor water pressure in a piping system and send the data to a central server for analysis and visualization. The researchers selected the appropriate hardware components for the system. Refer to Appendix A for the list of materials and their corresponding amounts.





The researchers ensured that the system is designed to be conveniently transported and move around to different locations. After careful design considerations the researchers came up with the proposed hardware design. Figure 1 above shows the system’s hardware design assembled in a CAD Software. The researchers then proceeded to the hardware development.

 The IOT interfaced Arduino water pressure monitoring system development has two stages mainly; Hardware and Software Development. The construction of the system, particularly, the circuit of the microcontroller, is the focus of hardware development. Figure 2 shows the prototype breadboard wiring diagram and the connection between the different hardware components.



Referring to the prototype breadboard wiring diagram, analog pin A1 is used to connect the analog wire of the pressure transducer. Consequently, the ground and voltage pins of the pressure transducer is connected to the GND pin and 5V, respectively. With the regards to the HW-125 SD card module; the GND pin to the common ground wires. Master In, Slave Out (MISO) to the digital pin 12, Serial Clock (SCK) to the digital pin 13, Master Out, Slave In (MOSI) to the digital PWM pin 11, Chip Select (CS) to the digital PWM pin 10. Moving on the LCD 20x4 I2C, VCC, GND, Serial Data (SDA), and Serial Clock (SCL) is connected to the common 5 volts wires, common ground wires, analog pin A4, and analog pin A5, respectively. Shifting the focus to the ESP8266, since the ESP8266 module doesn’t have a built-in voltage regulator, voltage level shifters were created. 1K and 2K-ohms resistors were used to level shift the 5V to a safe and passable value of 3.3V. This is to avoid high voltage values that are detrimental to the ESP8266 if allowed.

*c. Software System*

Following the hardware configuration, the researchers will move on to the software configuration. This method is separated into two parts: system microcontroller configuration and application development and integration with the Internet of Things.

*1) System Microcontroller*

The system uses an Arduino Uno R3 microcontroller which uses a variant of the C++ programming language. The microcontroller software designs are intended to accept, recognize, and process data given by the input device, as well as to transfer data to the application and show it on the output device (LCD).

The first and second section of code is where the libraries are set and the pins are defined which will be used throughout the code. This part also contains the declarations of various data type classes. The third section is the part of the code where the command of initialization of the modules is located. The fourth section of the code is where the command to print the collected data is shown on the output device (LCD). However, the microcontroller has to convert the analog signal (sensor value) read by the pressure transducer to a digital signal. The researchers used the Analog Digital Signal (ADS) conversion formula. The microcontroller must now convert the digital signal to a pressure reading. The linear equation was employed by the researchers to convert the digital pressure to a pressure measurement. Section 5 of the code contains the command to save the pressure data to a software doc, which is then transmitted to the ESP8266 WiFi Module. Simultaneously, the data being sent is being stored on the SD card module. (see Appendix A).

The ESP8266 WiFi module is a critical component of the system. This module serves as a link between the system and the database. This module intends to make the system part of a networked ecosystem, allowing it to exchange data, accept commands, and communicate with other devices or cloud services.

Located in the first section of the code is the command to the compiler to insert the contents of the specified file into your code before it is compiled. In this section, the library that researchers had included are as follows;

1. #include <ESP8266WiFi.h> - library for the WiFi Module.
2. #include <ArduinoJson.h>, #include <ArduinoJson.hpp> - The "ArduinoJson" library provides a set of powerful tools for working with JSON (JavaScript Object Notation) data in Arduino and other embedded systems. JSON is a lightweight data-interchange format that is commonly used for data serialization and communication between different devices or applications.
3. #include <Firebase\_ESP\_Client.h> - library for the database (firebase).
4. #include <SoftwareSerial.h> - This library enables the creation of software-based serial communication ports for Arduino boards lacking built-in hardware support for additional serial ports.

The second section of the code defines the preprocessor directives that define a constant Marco which are as follows; “WIFI\_SSID”, “WIFI\_PASSWORD”, “API KEY”, and “DATABASE”. In addition, a code known as an application programming interface (API) key is used to identify and validate an application or user [32]. The third section of the code is where the configuration command of the Firebase is located. The FirebaseData class is used to manage data and answers from Firebase and is a component of the Firebase Arduino library. The FirebaseAuth class is utilized for user authentication and authorisation with Firebase and is included in the Firebase Arduino library. The configuration parameters needed to connect to Firebase are stored and managed by the FirebaseConfig class. The line of Software serial nodemcu command uses the Arduino pins D6 and D5 for its software serial connection instance. For communication with another device, it configures a software-based serial port.

The void setup of the module is situated in the fourth section of the code. This section also contains the serial communications initialization and the ESP8266 WiFi Module. The instruction to create a static json doc is contained in the fifth part of the code. This is meant for communication between the WiFi module and the Arduino; however, there is a counter instruction that will clear the json document after 10 times of receiving data to avoid unexpected behavior and errors during data transmission (see Appendix B).

*2) Application Development and integration with Internet of Things*

One of the aspects of the Internet of Things is a device application or app. It enables systems to transfer data acquired by systems to mobile phones or other devices. The researchers used an application to combine the water pressure monitoring system with the Internet of things feature in this system.

In this system, the Arduino Uno R3 receives and converts sensor data to a pressure value. It is then printed on the Liquid Crystal Display (LCD). In a form of serial communication, the Arduino will then transmit the pressure value to the ESP8866 WiFi module. After serialization, the ESP module will send the value to the Firebase Realtime Database and writes to the ThingSpeak channel created by the researchers. This will allow the application developed through MIT App Inventor 2 to display the stored data. In coherence, this will allow user to monitor the pressure in real time either remotely or immediately (see Appendix C).

To create an interaction between the application and the system, the researches initiated the idea of having notification alerts whenever the pressure data passes a certain threshold. To give an instance, when the sensor reads 100kpA, the application alerts the user. See Appendix E for the MIT code app blocks.

3) Connecting the Application

To connect the Water Pressure Monitoring Application which was created using the MIT App Inventor with a Firebase database, several processes are needed to attain. First, set up and configure the Firebase database which includes the creation of the Firebase console and enabling the Firebase Realtime Database. Also, obtain important credentials such as the Firebase API key for authentication. At the MIT App inventor, create an app and configure it to connect with the Firebase database which has been set-up. It can only be attained by adding the Firebase component into the application.

Within the app, by using the Firebase component blocks a connection will establish. Provide the credentials required to authenticate the access to the application. To retrieve the data from the created Firebase database, utilize the blocks "Get Value" or "Get Tag List". Pinpoint the location at the database from where the data will be fetched.

For storing or updating the data in Firebase database blocks such as "Set Value" or "Store Value" will be used. Also, specify the data to be written and the location in the database for update and storage purposes. The Firebase database offers real-time data, this database ensures any changes made to the data will immediately be reflected into the application. Handling the real-time updates are the event-driven blocks such as "Data Changes" or "Tag List Changed".

Once the data is updated or retrieved, it is integrated into the app's logic, user interface or other components using the MIT App Inventor''s visual blocks-based coding interface. Define how the data should be presented, also on how the apps behave based on the received data.

Following these steps ensures the connection between the MIT App Inventor made applications for the Water Pressure Monitoring System to its created Firebase database which allows a smooth processing of the data from storage, retrieval and real-time updates.

*d. Calibration*

Calibration is an essential step in using a water pressure sensor because it ensures that the readings it provides are accurate and reliable. The calibration process entails comparing sensor readings to actual pressure at a known point and adjusting sensor readings accordingly. This is accomplished by using a calibration device or by following the calibration instructions provided by the sensor manufacturer. The researchers chose to utilize an analog pressure gauge (actual pressure) as the calibration device in this system.

Typically, the sensor output is recorded at several different known pressure points during the calibration process. The collected data is then used to generate a calibration curve, which maps the sensor's output to actual pressure values, and is then adjusted to the program code.

The researchers calibrated the water pressure monitoring system by manually recording the sensor and analog gauge readings at the simultaneously. To do this, the researchers used a metal T-shaped connection pipe to attach the sensor and analog gauge from a water pipe, as shown below in Figure 3.





After attaching the sensor and analog pressure gauge, the researchers took 80 pressure readings with a 30-second interval for every 2 minutes. The researchers will have 20 mean pressure readings from the pressure transducer. For the analog pressure gauge, pressure will be taken every 2 minutes will be compared to the 20 mean pressure readings from the pressure transducer. Furthermore, the researchers employed the T-Test, Kolmogorov-Smirnov Test of Normality, and Levene’s Test of Homogeneity of Variance assess and compare the calibration data. This will compare the acquired data's relationship, and a graph will be constructed to model the results. The following hypothesis will be used to analyze the data obtained:

Null Hypothesis

Ho:

1. The samples drawn are normally distributed.

2. There is no significant difference between the analog pressure gauge and the pressure transducer variance values.

3. There is no significant difference between the value of the analog pressure gauge and the pressure transducer.

Alternative Hypothesis

Ha:

1. The samples drawn are not normally distributed.

2. There is a significant difference between the analog pressure gauge and the pressure transducer variance values.

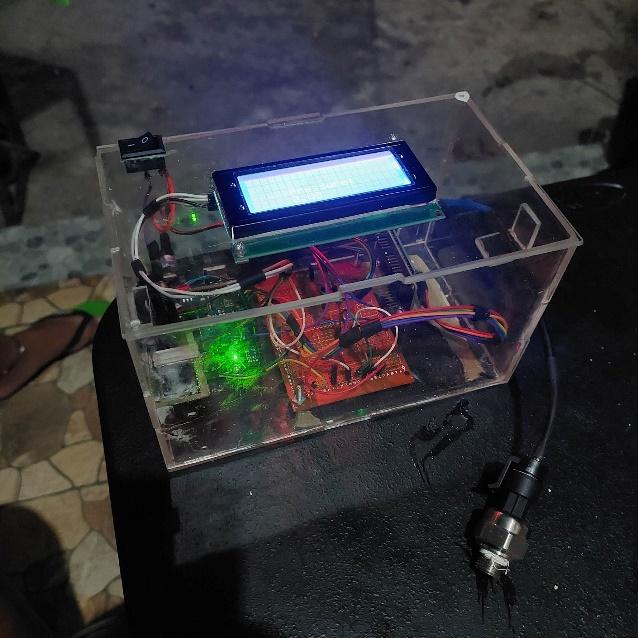
3. There is a significant difference between the value of the analog pressure gauge and the pressure transducer.

# 

# Results and Discussion

Following the proposed hardware design and system, the researchers have mainly integrated the suitable hardware components: the Arduino microcontroller, pressure transducers, and wireless communication modules.

The development of the proposed hardware design for the system involved several considerations. The researchers ensured that the system is portable and be deemed useful in monitoring water pressures in various settings in household and residential areas. By selecting the appropriate hardware components, the researchers created a portable and wireless monitoring system. Figure 4 below shows the completed hardware system.





*System Validation*

System calibration is an essential part in ensuring that the measurements taken by the device is accurate and reliable. By calibrating the system, we aim to reduce inherent mistakes, improve precision, and lay a solid foundation for future data collecting and analysis.

To calibrate the water pressure monitoring system, the researchers took a manual approach by comparing it to an analog pressure gauge. The researchers positioned the two devices side by side to directly observe the differences in their readings as shown in Figure 3. By this method, we were able to align the readings of our device with those of the analog pressure gauge, and were able to achieve the following results.

*Table 1. Significant difference between the Analog Pressure Gauge and the Pressure Transducer*

| Statistical Tool Used  Two-Tailed  T-Test for 2 Independent Samples | T value computed | α | df | p-value obtained | Interpretation | Decision |
| --- | --- | --- | --- | --- | --- | --- |
| Trial 1  (Analog and Pressure Transducer) | -0.07906 | 0.05 | 19 | 0.793876 | There is no significant difference between the value of the analog pressure gauge and the pressure transducer. | Fail to reject the null hypothesis. |
| Trial 2  (Analog and Pressure Transducer) | -0.09124 | 0.05 | 19 | 0.927784 | There is no significant difference between the value of the analog pressure gauge and the pressure transducer. | Fail to reject the null hypothesis |
| Trial 3  (Analog and Pressure Transducer) | -0.05702 | 0.05 | 19 | 0.954831 | There is no significant difference between the value of the analog pressure gauge and the pressure transducer. | Fail to reject the null hypothesis |

Table 1 shows the difference between the value of the analog pressure gauge and the pressure transducer by using a statistical treatment Two Tailed Test for 2 independent samples.

At Trial No.1 for both the Analog Pressure Gauge and Pressure Transducer, the researchers obtained a t-value of -0.07906 with 19 degrees of freedom at a 95 % confidence interval, since the T-value computed is less than the critical value at 19 degrees of freedom at 95% confidence interval. There was enough evidence to accept the null hypothesis. Thus, there is no significant difference between the value of the analog pressure gauge and the pressure transducer.

For Trial No.2, the researchers obtained a t-value of -0.09124 with 19 degrees of freedom at a 95 % confidence interval, since the T-value computed is less than the critical value at 19 degrees of freedom at 95% confidence interval. There was enough evidence to accept the null hypothesis. Thus, there is no significant difference between the value of the analog pressure gauge and the pressure transducer.

For Trial No.3, the researchers obtained a t-value of -0.05702 with 19 degrees of freedom at a 95 % confidence interval, since the T-value computed is less than the critical value at 19 degrees of freedom at 95% confidence interval. There was enough evidence to accept the null hypothesis. Thus, there is no significant difference between the value of the analog pressure gauge and the pressure transducer.

*Table 2. Kolmogorov-Smirnov Test of Normality for Analog Pressure Gauge and Pressure Transducer*

| Statistical Tool Used:  Kolmogorov-Smirnov Test of Normality | K-S test statistic value | Skewness | n | p-value obtained | Interpretation | Decision |
| --- | --- | --- | --- | --- | --- | --- |
| Trial No. 1  (Analog) | 0.24838 | 0.089524 | 20 | 0.14237 | The data does not differ significantly from that which is normally distributed. | The samples are normally distributed. |
| Trial No. 1  (Digital) | 0.24021 | 0.043704 | 20 | 0.16839 | The data does not differ significantly from that which is normally distributed. | The samples are normally distributed. |
| Trial No. 2  (Analog) | 0 .25013 | 0.0200311 | 20 | 0.13724 | The data does not differ significantly from that which is normally distributed. | The samples are normally distributed. |
| Trial No. 2  (Digital) | 0 .2396 | 0.110021 | 20 | 0.17046 | The data does not differ significantly from that which is normally distributed. | The samples are normally distributed. |
| Trial No. 3  (Analog) | 0.24895 | 0.006742 | 20 | 0.1407 | The data does not differ significantly from that which is normally distributed. | The samples are normally distributed. |
| Trial No. 3  (Digital) | 0.23648. | 0.040424 | 20 | 0.18145 | The data does not differ significantly from that which is normally distributed. | The samples are normally distributed. |

Table 2 above shows how normally distributed are the values of the analog pressure gauge and the pressure transducer by using a statistical treatment Kolmogorov-Smirnov Test of Normality.

The Kolmogorov Smirnov test (KS test) is another statistical test employed to evaluate whether a given sample or dataset follows a specific theoretical probability distribution, often the normal distribution.

Unlike parametric tests, it does not make any assumptions about the underlying distribution of the data and is considered non parametric. The KS test compares the cumulative distribution function (CDF) of the sample data with the expected CDF of the theoretical distribution. In this comparison it uses a test statistic called maximum absolute difference (D) between these two CDFs. According to its null hypothesis the sample in question is drawn from the specified theoretical distribution.

From the table the values obtained by the K-S test from the 6 samples of the 3 trials, the values obtained are as follows: In Trial No. 1, the researchers obtained a K-S test statistic value of 0.24838 for analog and 0.24021, in the same manner the researchers obtained a skewness of 0.089524 and 0.16839 respectively. In Trial No. 2, the researchers obtained the values of K-S Test Statistics 0.25013 and 0 .2396 with a skewness value of 0.0200311 and 0.110021. In Trial 3, the K-S Test Statistics values obtained are 0.24895 and 0.23648 and skewness value of 0.006742 and 0.040424.

From the obtained values at the table 2, it shows that the data gathered does not differ significantly from which is normally distributed. Thus, the samples are normally distributed.

*Table 3. The significant difference of variance between the Analog Pressure Gauge and Pressure Transducer*

| Statistical Tool Used:  Levene’s Test | f-ratio value | α | df within treatments | p-value obtained | Interpretation | Decision |
| --- | --- | --- | --- | --- | --- | --- |
| Group No. 1 | 0.00021 | 0.05 | 39 | .988625 | There is no significant difference between the variances between the analog pressure gauge and the pressure transducer. | The required homogeneity is met. |
| Group No. 2 | 0.00115 | 0.05 | 39 | 0.97317 | There is no significant difference between the variances between the analog pressure gauge and the pressure transducer. | The required homogeneity is met. |
| Group No. 3 | 0.00524 | 0.05 | 39 | 0.942697 | There is no significant difference between the variances between the analog pressure gauge and the pressure transducer. | The required homogeneity is met. |

Table 3 above shows the difference between the variances of the analog pressure gauge and the pressure transducer by using a statistical treatment Levene’s Test.

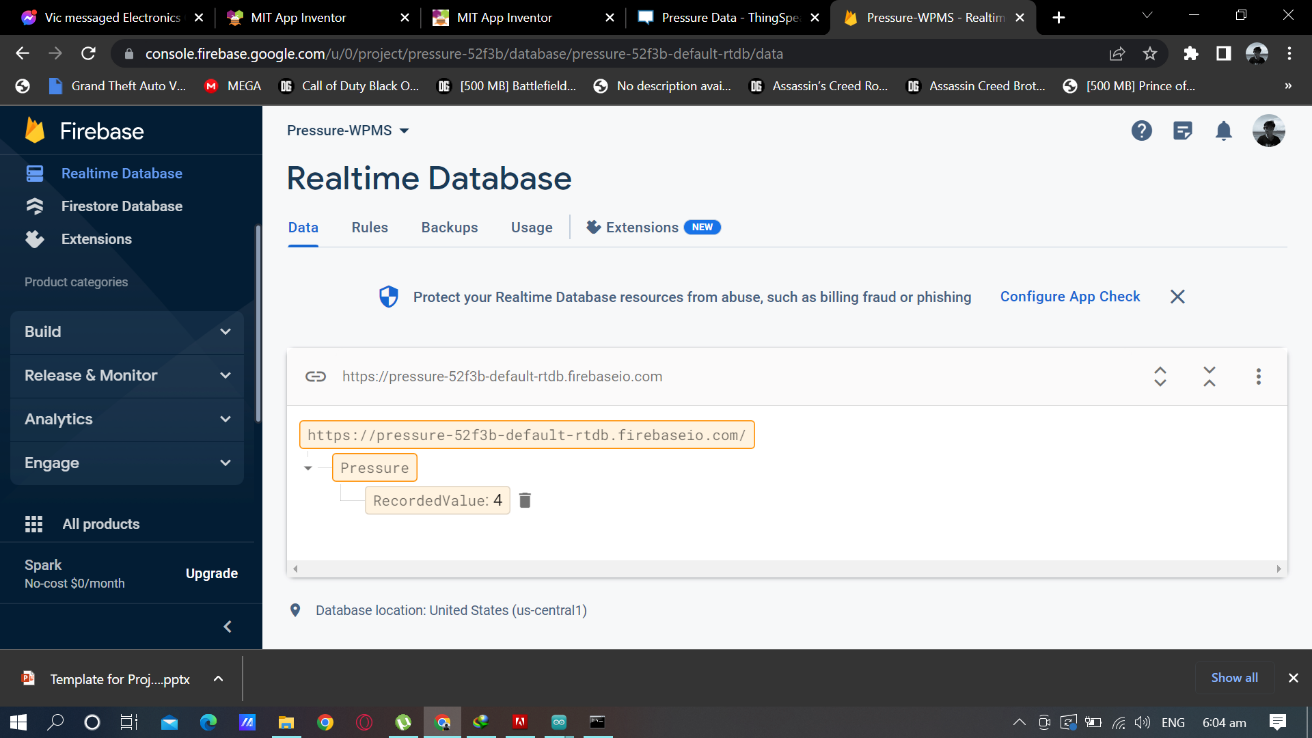
Levene’s test is a statistical test used to determine if the variances across different groups or samples are homogeneous. It is commonly used as a preliminary analysis before conducting parametric tests like the t test or analysis of variance (ANOVA) which assume equal variances between groups. The test aims to assess whether the variability (variance) of a variable is similar across groups or samples. The null hypothesis for Levenes test assumes equal variances across all groups, while the alternative hypothesis suggests that at least one group has a different variance.

At Trial No. 1 the researchers obtained a f-ratio value of 0.00021 with 39 degrees of freedom within treatments at a 95 % confidence interval, the p-value obtained is 0.988625, since the f-ratio value computed is less than the critical value at 39 degrees of freedom within treatments at 95% confidence interval. There was enough evidence to accept the null hypothesis. There is no significant difference between the variances between the analog pressure gauge and the pressure transducer. Thus, the required homogeneity is met.

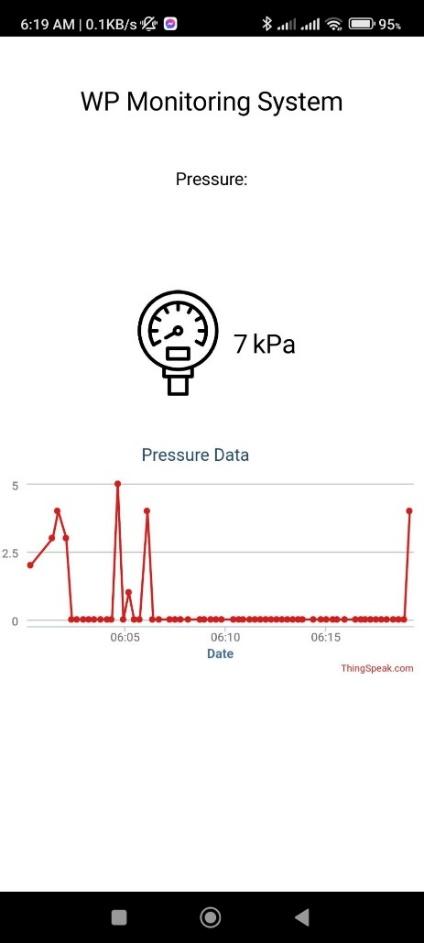
At Trial No. 2 the researchers obtained a f-ratio value of 0.00115 with 39 degrees of freedom within treatments at a 95 % confidence interval, the p-value obtained is 0.97317, since the f-ratio value computed is less than the critical value at 39 degrees of freedom within treatments at 95% confidence interval. There was enough evidence to accept the null hypothesis. There is no significant difference between the variances between the analog pressure gauge and the pressure transducer. Thus, the required homogeneity is met.

At Trial No. 3 the researchers obtained a f-ratio value of 0.00524 with 39 degrees of freedom within treatments at a 95 % confidence interval, the p-value obtained is 0.942697, since the f-ratio value computed is less than the critical value at 39 degrees of freedom within treatments at 95% confidence interval. There was enough evidence to accept the null hypothesis. There is no significant difference between the variances between the analog pressure gauge and the pressure transducer. Thus, the required homogeneity is met.

A software system evaluation was also done. Figure 5 and 6 shows the working Realtime Database and the Water Pressure Monitoring System Application. As shown in the figures below, the application graphs the changes in pressure as the Realtime Database receives data.









The ESP8266 writes the data communicated by the Arduino to the ThingSpeaks channel created by the researchers. The transmitted data is visualized into a line graph by utilizing the website's automatic tabulation. The x-axis is the date and time, while the y-axis is the pressure data.

# Summary and Conclusion

This study intended to develop a water pressure monitoring device that integrates the Internet of Things capabilities to acquire real-time data. By programming the device to capture and transmit the system's readings to a dedicated server and app, data logging and analysis became more efficient and provided a more comprehensive analysis.

The system's ability to obtain real-time data provides significant advantages for monitoring water pressures in various applications. The advantages include immediate detection of pressure change or fluctuations, leading to timely response to anomalies or issues. With the help of the internet and the app, it provides a more user-friendly interface for data logging, visualization, and analysis that enhances the accessibility and usability of the monitoring system.

To test the accuracy and reliability of the monitoring system the researchers used calibration with the help of statistical tool namely T-Test, Kolmogorov-Smirnov Test of Normality, and Levene’s Test of Homogeneity of Variance. It was observed that there is no significant difference between the analog pressure gauge and the pressure transducers. It was also observed that there is a significant relationship between the analog pressure gauge and the pressure transducers.

From the hardware and software evaluation that has been conducted, it is concluded that the water pressure monitoring system functioned with accuracy and can be utilized to perform real-time water pressure monitoring. The hardware system shows complete circuit build-up and the software system shows no malfunction. Real-time pressure readings are received by the Realtime Database and the application.

# Recommendations

The researchers recommend of using Arduino Mega 2560 Rev3 as substitute to the Arduino UNO R3, by changing the microcontroller, the program memory will drastically change from 32 KB to 256 KB of flash memory or program memory. With the additional memory, larger program or include more libraries and functionalities in your code. In lined with this, the Arduino Mega 2560 rev3 should be integrated with the PCB means that the microcontroller must be removed and soldered to the PCB.

The researchers also recommend of using the following for the betterment of the system: RTC Module for tracking the time, date, and year, to provide accurate timekeeping functionality. Turbidity sensor for detect water clarity, pH sensor for measurement of alkalinity and acidity of the water, temperature sensor for measuring the temperature of the water, and integration of other sensors for measuring water quantity and quality. The researchers also recommend to use a more accurate pressure measuring device aside from the pressure gauge, to further test the reliability of the system.

# References

| [1] | T. J. a. L. A., " “Progress in Practical Prediction of Pressure/Leakage, Pressure/Burst Frequency and Pressure/Consumption Relationships”," *Proceedings of IWA Special Conference 'Leakage 2005' , Halifax, Canada, September 2005,* 2005. |
| --- | --- |
| [2] | C. B., "“Experiences in DMA redesign at the Water Board of Lemesos”," *Proceedings of IWA Special Conference 'Leakage 2005' , Halifax, Canada, September 2005,* 2005. |
| [3] | M. (. Banzi, Getting Started with Arduino, CA: USA: Make: Books; 1st edition, 2008. |
| [4] | P. Supply, "Residential water pressure explained," Plumbing Supply, 2017. [Online]. Available: https://www.plumbingsupply.com/residential-water-pressure-explained.html. [Accessed 12 June 2023]. |
| [5] | S. &. G. Y. B. McNamara, "MEMS-Based Sensors," *In P. Bhattacharya, R. Fornari, & H. Kamimura (Eds.), Comprehensive Semiconductor Science and Technology,* pp. 527-559, 2011. |
| [6] | B. Vigna, " “Future of MEMS: an industry point of view. InThermal, mechanical, and multiphysics simulation andexperiments in Micro-Electronics and Micro-Systems,2006”,," *7th International Conference on 2006 Apr 24,EuroSime2006, IEEE,,* pp. 1-8, 2006. |
| [7] | P. Thomson, "Remote monitoring of rural water systems: A pathway to improved performance and sustainability?," *WIREs Water,* vol. 8, no. 2, 2020. |
| [8] | F. D. M. M. M. B. S. D. &. T. F. Bruno, " A Pressure Monitoring System for Water Distribution Networks Based on Arduino Microcontroller," *Water,* vol. 13, no. 17, p. 2321, 2021. |
| [9] | L. A. e. al., "Towards Industry 4.0: Mapping digital technologies for supply chain management-marketing integration," *Business Process Management Journal,* vol. 25, no. 2, pp. 323-346, 2019. |
| [10] | M. &. A. A. &. F. M. Ghobakhloo, "Lean-green manufacturing: the enabling role of information technology resource," *Kybernetes,* vol. 47, no. 1, 2018. |
| [11] | S. S. G. A. &. G. S. A. Kamble, "Sustainable industry 4.0 framework: a systematic literature review identifying the current trends and future perspectives.," *Process Safety and Environmental Protection,* vol. 117, pp. 408-425, 2018. |
| [12] | H. F. P. K. H.-G. F. T. &. H. M. Lasi, "Industry 4.0.," *Business & Information Systems Engineering,* vol. 6, no. 4, pp. 239-242, 2014. |
| [13] | S. N. a. B. X. G. Beier, " “More Sustainability in Industry through Industrial Internet of Things?,”," *Applied Sciences,* vol. 8, no. 2, p. 219, 2018. |
| [14] | J. M. &. B. O. &. V. K.-I. Müller, " "Fortune favors the prepared: How SMEs approach business model innovations in Industry 4.0,"," *Technological Forecasting and Social Change,* vol. 132(c), pp. 2-17, 2018. |
| [15] | G. L. T. &. D. Fettermann, "“Implementation of Industry 4.0 and Lean Production in Brazilian Manufacturing"," *International Journal of Production Research,* vol. 56, no. 8, pp. 2975-2987, 2018. |
| [16] | G. N. S. Z. T. e. a. Beier, "Sustainability aspects of a digitalized industry – A comparative study from China and Germany.," *Int. J. of Precis. Eng. and Manuf.-Green Tech.,* vol. 4, p. 227–234 , 2017. |
| [17] | M. B. I. M. R. &. U. M. N. Amin, "Internet of things (IoT) based health monitoring system: A review," *Journal of medical systems,* vol. 41, no. 7, p. 114, 2017. |
| [18] | L. D. X. a. S. Z. S. Li, "The internet of things: a survey," *Information Systems Frontiers ,* vol. 2, no. 17, pp. 243-259, 2015. |
| [19] | A. J. Z. M. A. &. S. A. F. Jara, "Interoperability, security and privacy in the Internet of Things: The EU approach," *IEEE Internet of Things Journal,* vol. 2, no. 4, pp. 263-275, 2015. |
| [20] | A. Raza, A. Ikram and A. Amin, A Review of Low Cost and Power Efficient Development Boards for IoT Applications., San Francisco, CA, USA: 2016 Future Technologies Conference (FTC), 2016, p. 786–790. |
| [21] | J. Gubbi, R. Buyya, S. Marusic and M. Palaniswami, " Internet of Things (IoT): A vision, architectural elements, and future directions.," *Futur. Gener. Comput. Syst.,* vol. 29, p. 1645–1660, 2013. |
| [22] | M. Zdravkovi, M. Trajanović, J. Sarraipa, R. Jardim-gonçalves and M. Lezoche, " Survey of Internet-of-Things platforms," *In Proceedings of the 6th International Conference on Information Society and Technology, Kopaonik, Serbia,* p. 216–220, 2016. |
| [23] | A. Botta, W. De Donato, V. Persico and A. Pescapé, "Integration of Cloud Computing and Internet of Things: A Survey.," *Futur. Gener. Comput. Syst. ,* vol. 56, p. 684–700, 2015. |
| [24] | S. K. a. R. D. H. G. C. Fox, "Architecture and measured characteristics of a cloud-based internet of things," *In Collaboration Technologies and Systems (CTS) 2012 International Conference,* p. 6–12, 2012. |
| [25] | P. e. Jiang, ""Design of a water environment monitoring sytem based on wireless sensor networks," *Senors,* vol. 9.8, pp. 6411-6434, 2009. |
| [26] | N. a. a. R. R. Vijayakumar, ""The real time monitoring of water quality in IoT environment."," *International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS),* 2015. |
| [27] | T. e. al, "Web Based Real Time Water Pressure Monitoring System," in *International Conference On Electrical Engineering, Computer Science and Informatics (EECSI)*, Yogyakarta, Indonesia, 2014. |
| [28] | T. B. Drummond GB, "Statistics, probability, significance, likelihood: words mean what we define them to mean.," *The Journal of physiology,* 2011. |
| [29] | B. L. a. O. R. T. Bowerman, Linear Statistical Models: An Applied Approach,, vol. II, Belmont, California: Duxbury Press, 1990. |
| [30] | N. R. a. S. H. Draper, Applied Regression Analysis, Second Edition, John Wiley & Sons, 1981. |
| [31] | J. J. P. H. S. D. P. C. E. M. P. &. M. A. Kissock, "Inverse Modeling Toolkit:," *ASHRAE ANNUAL MEETING,* vol. 109, no. 2, 2003. |
| [32] | Fortinet, Inc., "What Is an API Key?," Fortinet, Inc, 2023. [Online]. Available: https://www.fortinet.com/resources/cyberglossary/api-key#:~:text=An%20application%20programming%20interface%20(API,secret%20token%20for%20authentication%20purposes.. [Accessed 11 June 2023]. |
| [33] | N. K. H. &. S. N. Bhattacharyya, "A review on IoT based monitoring systems for agricultural crops," *Journal of Ambient Intelligence and Humanized Computing ,* vol. 8, no. 5, pp. 717-731, 2017. |
| [34] | I. P. H. G. &. C. R. Ferreira, "Internet of Things (IoT) applications for traffic management: A survey," *Computer Networks,* vol. 151 , pp. 111-126, 2019. |
| [35] | J. Norman, "Kevin Ashton Invents the Term "The Internet of Things"," 2013. [Online]. |
| [36] | I. P. H. G. &. C. R. Ferreira, " Internet of Things (IoT) applications for traffic management: A survey," *Computer Networks,* vol. 151, pp. 111-126, 2019. |
| [37] | K. J. V. K. A. J. Vijay K. Varadan, RF MEMS and Their Applications, Arizona, USA: Wiley, 2002. |

# Appendices

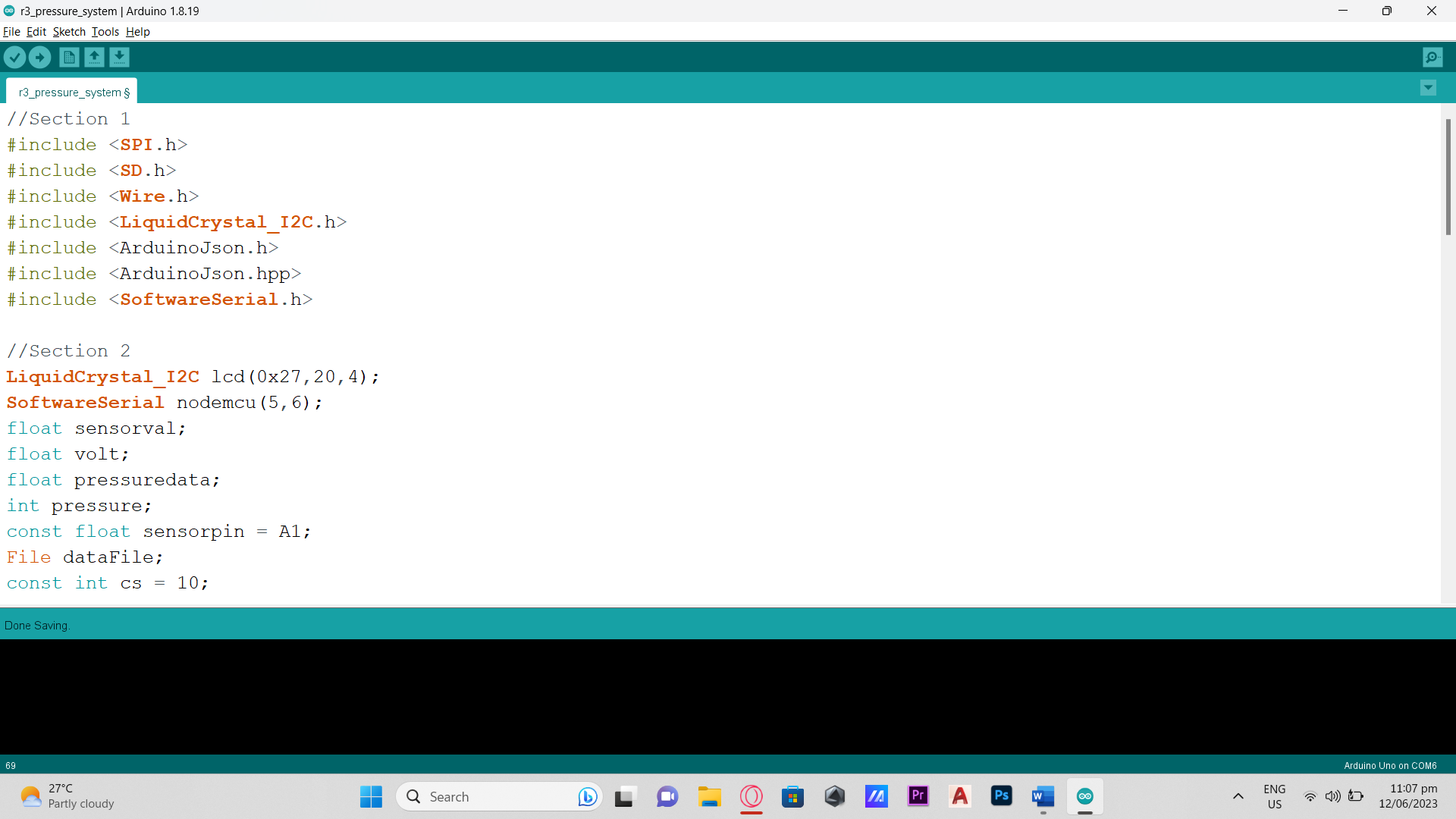
### Appendix A

Table A.1 Bill of Materials

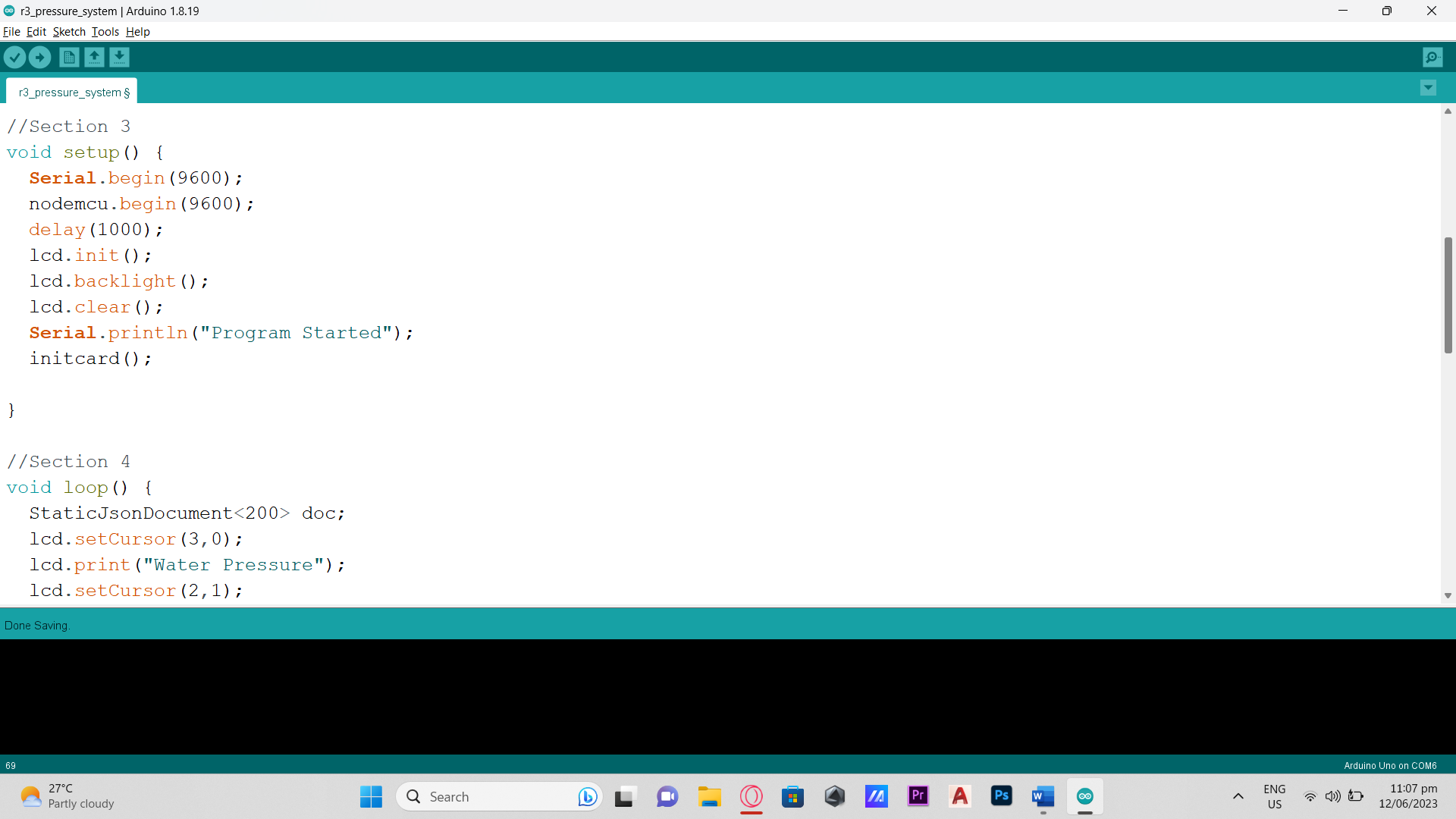
| Quantity | Unit | Description | Unit Cost | Amount |
| --- | --- | --- | --- | --- |
| 1 | pc | Plexiglass (2x2) | 400 | 400 |
| 1 | pc | Pressure Gauge | 200 | 200 |
| 1 | pc | UNO R3 board | 845 | 845 |
| 1 | pc | Pressure Transducer Transmitter Sensor | 749 | 749 |
| 1 | pc | ESP8266 ESP-12E WiFi development board | 160 | 160 |
| 1 | pc | 20x4 LCD Display I2C | 246 | 246 |
| 2 | pc | PCB Board | 28 | 56 |
| 2 | pc | GI Bushing RED | 37.3 | 74.6 |
| 2 | pc | GI Coupling RED | 39.25 | 78.5 |
| 2 | pc | GI Nipple S40 | 13.45 | 26.9 |
| 1 | pc | Shark SF2563 SS FLEXHOS | 113 | 113 |
| 1 | meter | Stranded Wire | 25 | 25 |
| 1 | pc | Switch | 10 | 10 |
| 1 | pc | Eveready 9V Battery | 93 | 93 |
|  |  |  | Total: | 3077 |

### Appendix B

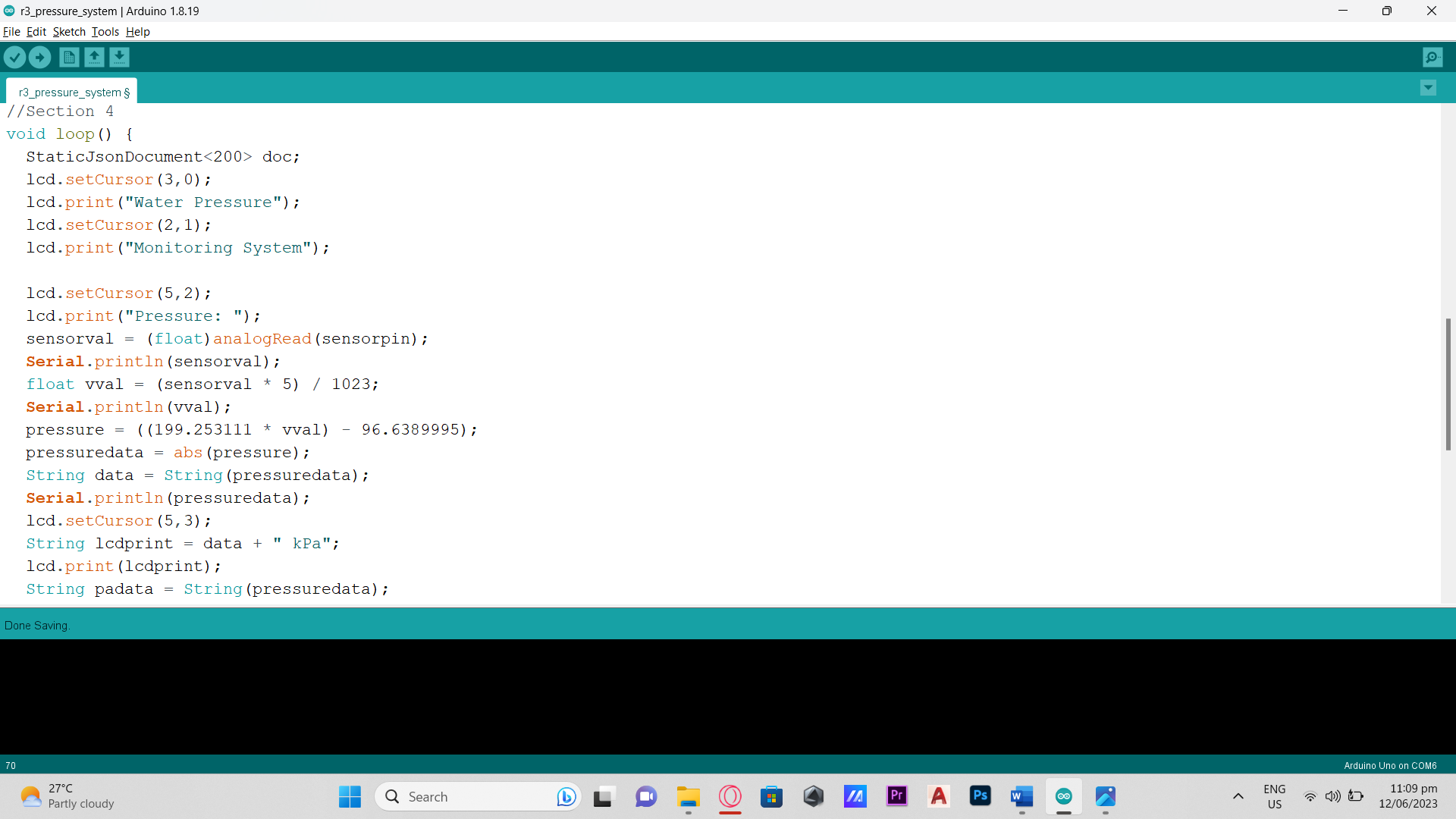
Arduino Uno R3 Code



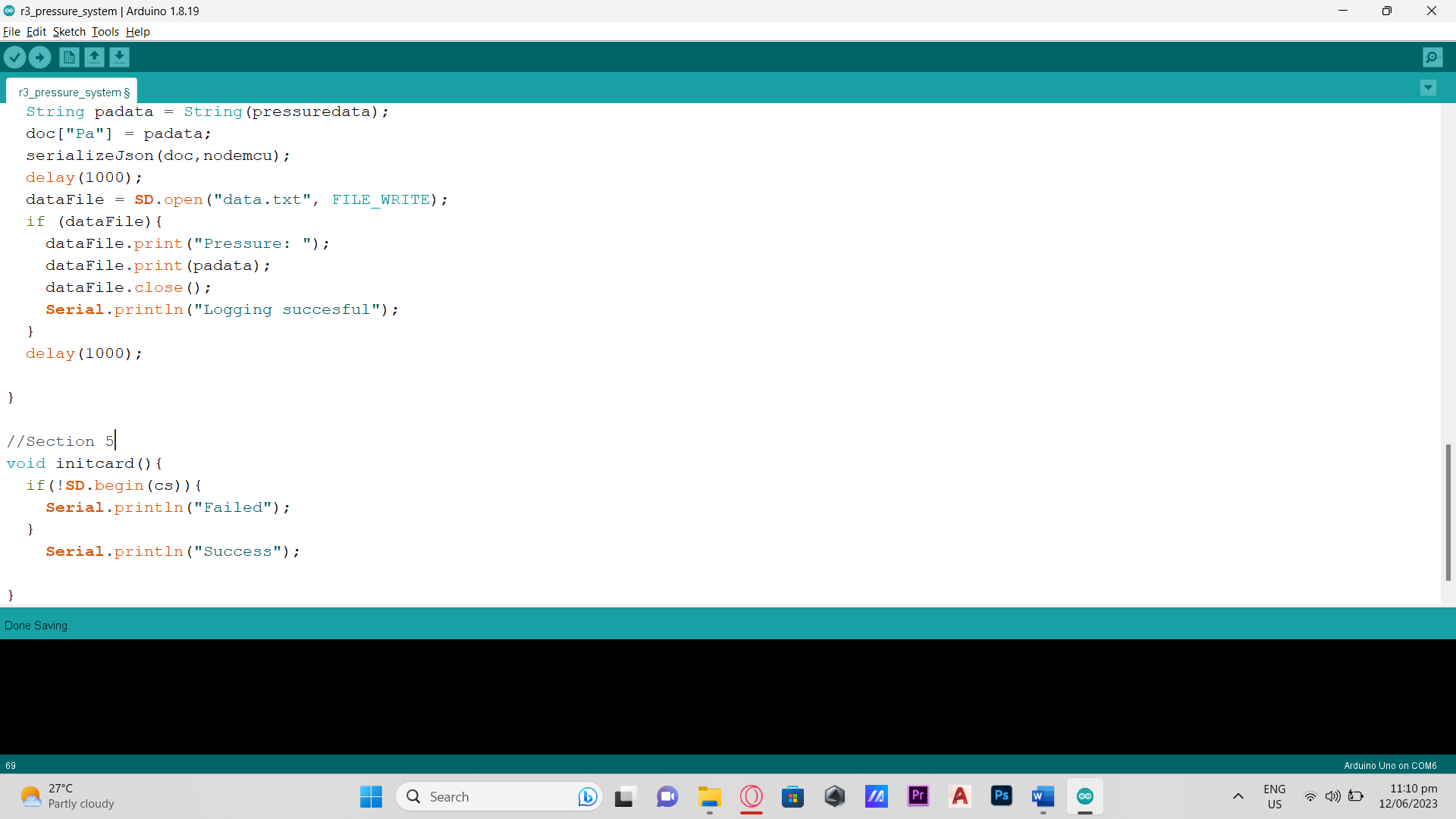
(a)



(b)



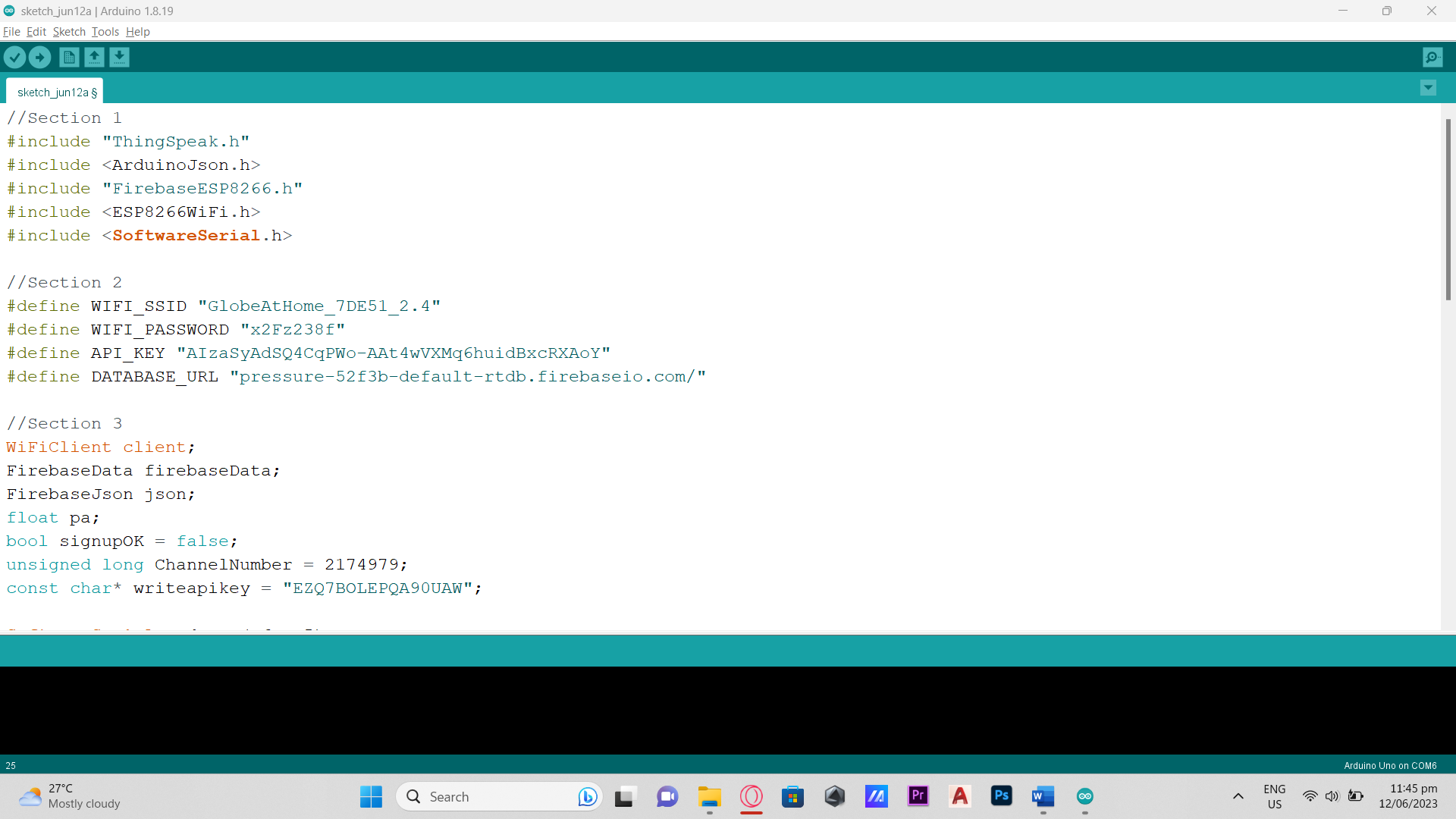
(c)



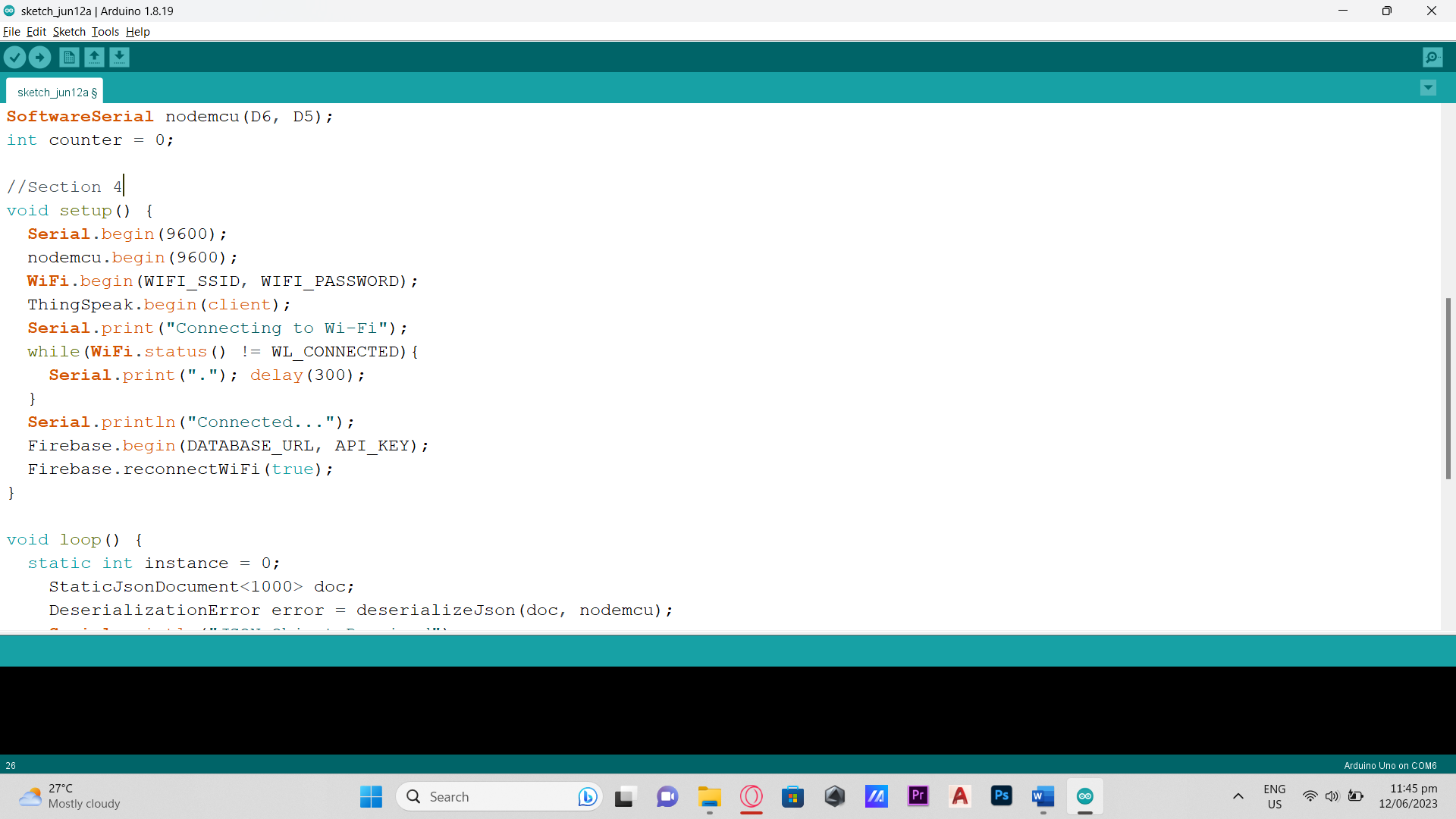
(d)

### Appendix C

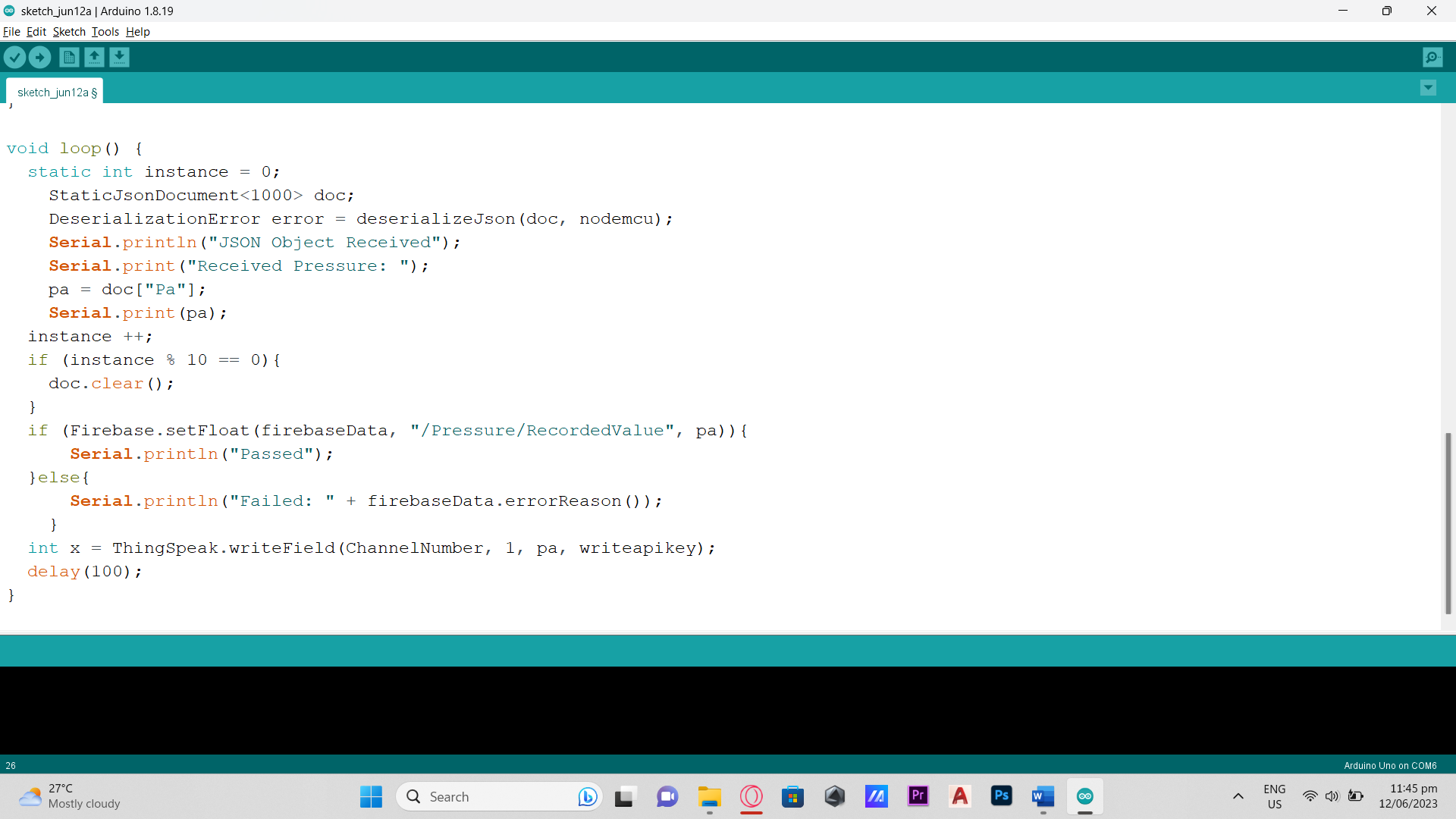
ESP8266 Wi-Fi Module Code



(a)



(b)



(c)

Appendix D

System Block Diagram



Appendix E

MIT App Inventor Code Blocks

# APPENDIX F

## Trial No.1 Data Recordings from Analog Pressure Gauge and Pressure Transducer

| Record No. | Time | Time Interval | DIGITAL | DIGITAL (mean) | ANALOG  (Manual Recording) |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 mins. | 30 sec | 287.69 | 272.37 | 275.8 |
| 30 sec | 288.5 |
| 30 sec | 265.02 |
| 30 sec | 248.27 |
| 2 | 2 mins. | 30 sec | 246.58 | 252.2275 | 255.106 |
| 30 sec | 246.1 |
| 30 sec | 258.4 |
| 30 sec | 257.83 |
| 3 | 2 mins. | 30 sec | 245.54 | 242.6475 | 241.317 |
| 30 sec | 243.69 |
| 30 sec | 241.21 |
| 30 sec | 240.15 |
| 4 | 2 mins. | 30 sec | 174.63 | 172.7625 | 172.369 |
| 30 sec | 173.56 |
| 30 sec | 172.24 |
| 30 sec | 170.62 |
| 5 | 2 mins. | 30 sec | 163.29 | 162.78 | 165.474 |
| 30 sec | 162.16 |
| 30 sec | 163.89 |
| 30 sec | 161.78 |
| 6 | 2 mins. | 30 sec | 237.43 | 235.955 | 234.422 |
| 30 sec | 236.45 |
| 30 sec | 235.46 |
| 30 sec | 234.48 |
| 7 | 2 mins. | 30 sec | 233.49 | 232.015 | 234.422 |
| 30 sec | 232.51 |
| 30 sec | 231.52 |
| 30 sec | 230.54 |
| 8 | 2 mins. | 30 sec | 229.59 | 229.8925 | 230.9744 |
| 30 sec | 230.54 |
| 30 sec | 229.89 |
| 30 sec | 229.55 |
| 9 | 2 mins. | 30 sec | 228.57 | 227.605 | 228 |
| 30 sec | 227.58 |
| 30 sec | 227.58 |
| 30 sec | 226.69 |
| 10 | 2 mins. | 30 sec | 148.56 | 147.3935 | 148.2373 |
| 30 sec | 147.17 |
| 30 sec | 147.014 |
| 30 sec | 146.83 |
| 11 | 2 mins. | 30 sec | 144.24 | 145.3725 | 144.79 |
| 30 sec | 145.87 |
| 30 sec | 146.46 |
| 30 sec | 144.92 |
| 12 | 2 mins. | 30 sec | 164.69 | 165.0125 | 165.474 |
| 30 sec | 163.54 |
| 30 sec | 167.19 |
| 30 sec | 164.63 |
| 13 | 2 mins. | 30 sec | 161.48 | 165.785 | 172.369 |
| 30 sec | 166.79 |
| 30 sec | 167.19 |
| 30 sec | 167.68 |
| 14 | 2 mins. | 30 sec | 165.12 | 165.345 | 165.474 |
| 30 sec | 165.39 |
| 30 sec | 165.9 |
| 30 sec | 164.97 |
| 15 | 2 mins. | 30 sec | 164.89 | 165.38 | 165.464 |
| 30 sec | 165.36 |
| 30 sec | 165.73 |
| 30 sec | 165.54 |
| 16 | 2 mins. | 30 sec | 115.73 | 103.515 | 103.421 |
| 30 sec | 112.36 |
| 30 sec | 93.17 |
| 30 sec | 92.8 |
| 17 | 2 mins. | 30 sec | 86.124 | 84.9885 | 89.6318 |
| 30 sec | 85.8 |
| 30 sec | 85.25 |
| 30 sec | 82.78 |
| 18 | 2 mins. | 30 sec | 157.8 | 161.9525 | 165.474 |
| 30 sec | 159.87 |
| 30 sec | 164.31 |
| 30 sec | 165.83 |
| 19 | 2 mins. | 30 sec | 166.73 | 165.745 | 165 |
| 30 sec | 165.67 |
| 30 sec | 165.28 |
| 30 sec | 165.3 |
| 20 | 2 mins. | 30 sec | 165.52 | 165.5575 | 165.474 |
| 30 sec | 165.74 |
| 30 sec | 165.43 |
| 30 sec | 165.54 |

# APPENDIX G

## Trial No.2 Data Recordings from Analog Pressure Gauge and Pressure Transducer

| Record No. | Time | Time Interval | DIGITAL | DIGITAL (mean) | ANALOG |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 mins. | 30 sec | 165.26 | 164.2125 | 165.474 |
| 30 sec | 165.45 |
| 30 sec | 163.65 |
| 30 sec | 162.49 |
| 2 | 2 mins. | 30 sec | 166.84 | 165.795 | 165 |
| 30 sec | 166.42 |
| 30 sec | 165.16 |
| 30 sec | 164.76 |
| 3 | 2 mins. | 30 sec | 165.13 | 165.2475 | 165.474 |
| 30 sec | 165.63 |
| 30 sec | 165.23 |
| 30 sec | 165 |
| 4 | 2 mins. | 30 sec | 174.78 | 172.6475 | 172.369 |
| 30 sec | 173.59 |
| 30 sec | 172.13 |
| 30 sec | 170.09 |
| 5 | 2 mins. | 30 sec | 163.98 | 163.015 | 165.474 |
| 30 sec | 162.28 |
| 30 sec | 164.37 |
| 30 sec | 161.43 |
| 6 | 2 mins. | 30 sec | 115.74 | 103.54 | 103.42 |
| 30 sec | 112.23 |
| 30 sec | 93.3 |
| 30 sec | 92.87 |
| 7 | 2 mins. | 30 sec | 234.53 | 232.5025 | 234.422 |
| 30 sec | 232.64 |
| 30 sec | 232.49 |
| 30 sec | 230.35 |
| 8 | 2 mins. | 30 sec | 148.9 | 147.65 | 151.685 |
| 30 sec | 147.48 |
| 30 sec | 147.57 |
| 30 sec | 146.65 |
| 9 | 2 mins. | 30 sec | 228.51 | 227.655 | 228 |
| 30 sec | 227.68 |
| 30 sec | 227.56 |
| 30 sec | 226.87 |
| 10 | 2 mins. | 30 sec | 229.85 | 229.9 | 227.53 |
| 30 sec | 230.35 |
| 30 sec | 229.53 |
| 30 sec | 229.88 |
| 11 | 2 mins. | 30 sec | 144.48 | 145.4175 | 144.79 |
| 30 sec | 145.79 |
| 30 sec | 146.5 |
| 30 sec | 144.9 |
| 12 | 2 mins. | 30 sec | 164.9 | 165.2775 | 165.474 |
| 30 sec | 163.79 |
| 30 sec | 167.95 |
| 30 sec | 164.47 |
| 13 | 2 mins. | 30 sec | 161.66 | 165.7675 | 172.369 |
| 30 sec | 166.59 |
| 30 sec | 167.49 |
| 30 sec | 167.33 |
| 14 | 2 mins. | 30 sec | 245.43 | 242.49 | 234.42 |
| 30 sec | 243.54 |
| 30 sec | 240.68 |
| 30 sec | 240.29 |
| 15 | 2 mins. | 30 sec | 165.8 | 165.345 | 165.464 |
| 30 sec | 165.31 |
| 30 sec | 165.93 |
| 30 sec | 164.34 |
| 16 | 2 mins. | 30 sec | 238.35 | 236.925 | 242.317 |
| 30 sec | 238.21 |
| 30 sec | 236.37 |
| 30 sec | 234.77 |
| 17 | 2 mins. | 30 sec | 86.18 | 84.435 | 89.6318 |
| 30 sec | 85.18 |
| 30 sec | 83.9 |
| 30 sec | 82.48 |
| 18 | 2 mins. | 30 sec | 157.82 | 161.9 | 165.474 |
| 30 sec | 159.72 |
| 30 sec | 164.32 |
| 30 sec | 165.74 |
| 19 | 2 mins. | 30 sec | 246.93 | 245.41 | 248.21 |
| 30 sec | 246.43 |
| 30 sec | 245.65 |
| 30 sec | 242.63 |
| 20 | 2 mins. | 30 sec | 288.12 | 282.35 | 289.58 |
| 30 sec | 288.34 |
| 30 sec | 287.17 |
| 30 sec | 265.78 |

# APPENDIX H

## Trial No.3 Data Recordings from Analog Pressure Gauge and Pressure Transducer

| Record No. | Time | Time Interval | DIGITAL | DIGITAL (mean) | ANALOG |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 mins. | 30 sec | 288.65 | 273.045 | 268.896 |
| 30 sec | 288.8 |
| 30 sec | 265.85 |
| 30 sec | 248.88 |
| 2 | 2 mins. | 30 sec | 154.65 | 150.5375 | 151.685 |
| 30 sec | 154.26 |
| 30 sec | 147.03 |
| 30 sec | 146.21 |
| 3 | 2 mins. | 30 sec | 161.12 | 165.9 | 172.369 |
| 30 sec | 167.24 |
| 30 sec | 167.72 |
| 30 sec | 167.52 |
| 4 | 2 mins. | 30 sec | 165.65 | 164.73 | 165.47 |
| 30 sec | 163.28 |
| 30 sec | 165.94 |
| 30 sec | 164.03 |
| 5 | 2 mins. | 30 sec | 164.25 | 165.7075 | 165.464 |
| 30 sec | 165.29 |
| 30 sec | 167.78 |
| 30 sec | 165.51 |
| 6 | 2 mins. | 30 sec | 115.46 | 103.285 | 110.316 |
| 30 sec | 112.08 |
| 30 sec | 93.11 |
| 30 sec | 92.49 |
| 7 | 2 mins. | 30 sec | 86.12 | 84.365 | 82.737 |
| 30 sec | 85.23 |
| 30 sec | 83.36 |
| 30 sec | 82.75 |
| 8 | 2 mins. | 30 sec | 157.5 | 161.83 | 165.47 |
| 30 sec | 159.37 |
| 30 sec | 164.64 |
| 30 sec | 165.82 |
| 9 | 2 mins. | 30 sec | 166.07 | 165.4525 | 165 |
| 30 sec | 165.48 |
| 30 sec | 165.23 |
| 30 sec | 165.03 |
| 10 | 2 mins. | 30 sec | 246.48 | 252.32 | 255.11 |
| 30 sec | 246.89 |
| 30 sec | 258.45 |
| 30 sec | 257.47 |
| 11 | 2 mins. | 30 sec | 144.34 | 145.1325 | 144.79 |
| 30 sec | 145.52 |
| 30 sec | 146.43 |
| 30 sec | 144.24 |
| 12 | 2 mins. | 30 sec | 169.94 | 166.8025 | 172.369 |
| 30 sec | 168.65 |
| 30 sec | 164 |
| 30 sec | 164.62 |
| 13 | 2 mins. | 30 sec | 257.7 | 246.01 | 248.21 |
| 30 sec | 243.5 |
| 30 sec | 241.89 |
| 30 sec | 240.94 |
| 14 | 2 mins. | 30 sec | 174.16 | 172.88 | 172.369 |
| 30 sec | 173.56 |
| 30 sec | 172.84 |
| 30 sec | 170.96 |
| 15 | 2 mins. | 30 sec | 162.89 | 162.63 | 158.58 |
| 30 sec | 162.87 |
| 30 sec | 163.59 |
| 30 sec | 161.15 |
| 16 | 2 mins. | 30 sec | 237.48 | 235.99 | 234.42 |
| 30 sec | 236.36 |
| 30 sec | 235.24 |
| 30 sec | 234.87 |
| 17 | 2 mins. | 30 sec | 233.13 | 231.84 | 234.422 |
| 30 sec | 232.34 |
| 30 sec | 231.63 |
| 30 sec | 230.26 |
| 18 | 2 mins. | 30 sec | 229.32 | 229.6925 | 227.527 |
| 30 sec | 230.65 |
| 30 sec | 229.48 |
| 30 sec | 229.32 |
| 19 | 2 mins. | 30 sec | 228.32 | 227.53 | 228 |
| 30 sec | 227.45 |
| 30 sec | 227.93 |
| 30 sec | 226.42 |
| 20 | 2 mins. | 30 sec | 166.46 | 165.085 | 165.474 |
| 30 sec | 165.64 |
| 30 sec | 165.79 |
| 30 sec | 162.45 |