



FACULTY: COMPUTER SCIENCE AND ENGINEERING
FINAL THESIS PROJECT (BACHELOR THESIS)



TOPIC:
SENSOR BASED WATER QUALITY MONITORING SYSTEM

Suggested mentor:
Dijana Capeska Bogatinoska, PhD

Student:
Elena Simonoska 937

STATEMENT

I, student Elena Simonoska, enrolled at the University for Information Science and Technology - “St. Paul the Apostle” – Ohrid in Faculty of Computer Science and Engineering with index number 937, under full material, moral and criminal responsibility declare that I am the author of this paper, titled Sensor based Water Quality Monitoring System, which is an original work that I have written under the supervision of Dijana Capeska Bogatinoska, Ph.D.

Ohrid

Signature

Abstract

Water pollution is one of the biggest threats today which affects the lifecycle of the whole ecosystem. This problem can be solved by protecting affected and preventing non-affected environments. If pollution is detected in early stages critical situations can be avoided. For having insight into water quality, it is important to follow water parameters constantly and in real-time and to react immediately when a parameter is out of range.

Conventional methods of testing water include collecting water samples manually. This process is time consuming, requires human effort and usually gives incorrect results. The reason for mistakes using laboratory testing is the influence of conditions during transport of water samples. At this point smart solution becomes significant innovation and, it can replace traditional methods and to address related aforementioned issues.

This paper presents a detailed review of the project in the field of water quality monitoring. The project includes usage of microcontroller, four sensors for water parameters and application for visual representation of collected data. The developed model is tested in several phases to prove uninterruptedly operation of the system.

The aim of the project was to develop low-cost, portable and efficient prototype which can be used for many purposes. The final result is a system which measures temperature, turbidity, pH and water level and displays them in a user-friendly way.

Keywords: IoT, Raspberry Pi, Water Quality, Water Parameters

Contents

1. Introduction	12
1.1. Problem Overview	12
1.2. Thesis Overview	13
1.3. What has been done	14
1.4. What has been achieved	14
2. State of the Art	14
3. Project Overview	15
3.1. Motivation	15
3.2. Environmental Impact	15
3.2.1. Impact on Ohrid Lake	16
3.3. Technical Challenge	16
3.4. Limitations	17
3.5. Alternatives	17
3.5.1. Laboratory Testing	17
3.5.2. Remote Sensing	17
3.5.3. Change in Organism's Behaviour	17
3.5.4. Sensor Based Solution	18
3.5.5. Preferred Solution	18
3.6. Choosing Microcontroller	18
3.7. Project Requirements	19
3.7.1. Functional Requirements	19
3.7.2. Non-Functional Requirements	20
3.7.3. Performance Requirements	21
3.7.4. Hardware and Software Requirements	21
3.8. Analysis	22
3.9. Deliverables	23
4. General Definition of Water Quality	23
4.1. Water Parameters	24
4.1.1. pH	24
4.1.2. Temperature	25
4.1.3. Turbidity	25
4.1.4. Water Level	26
4.2. Classification of Water	26
4.3. Summary	26
5. Hardware Description	27
5.1. Raspberry Pi	27
5.1.1. Origin and history of Raspberry Pi	27
5.1.2. A guided tour on Raspberry Pi	29
5.2. Sensors	30
5.2.1. Classification of sensors	31
5.2.2. Application of sensors	31

5.3. pH sensor	33
5.3.1. How does pH Sensor work?	33
5.4. Temperature Sensor	33
5.4.1. How does Temperature Sensor work?	34
5.5. Turbidity Sensor	34
5.5.1. How does Turbidity Sensor work?	34
5.6. Distance Sensor	35
5.6.1. Resistor	35
5.6.2. How does Distance Sensor work?	35
5.7. Wiring Components	35
5.8. Summary	36
6. Programming Language Justification	38
7. Development of a system	39
7.1. Laravel Application	40
7.1.1. Model View Controller	40
7.1.2. Middleware	40
7.1.3. Migration	40
7.1.4. Register and Login	41
7.1.5. Visitors	42
7.1.6. Admin	42
7.1.7. User	42
7.1.8. Classification	45
7.1.9. Laravel Summary	46
7.2. Hardware Coding	47
7.3. Design Summary	50
8. Visual Design	52
8.1. Admin	52
8.2. User	52
8.3. Visitors	53
8.4. Flowchart of application	54
9. Testing and Results	55
9.1. Unit Testing	56
9.1.1. pH Unit Testing	56
9.1.2. Temperature Unit Testing	57
9.1.3. Turbidity Unit Testing	58
9.1.4. Water Level Unit Testing	58
9.2. Integration Testing	59
9.3. System Testing	60
10. Use Cases	61
10.1. Smart Lake	61
10.2. Drinking Water Station	61
10.3. Fishpond	62
10.4. Specie Detection	62
11. Future Work	62

12. Conclusion	63
13. References	64
14. Appendices	66

List of figures:

Figure 1. System's interfaces	22
Figure. 2 Raspberry Pi Layout	29
Figure 3. Sensor working principle	30
Figure 4. Relationship between sensors, IoT and Smart City	33
Figure 5. Wiring Components	38
Figure 6. Sequence Flow Diagram	39
Figure 7. MVC Architecture	40
Figure 8. Classification of Water	46
Figure 9. Relationship between voltage and turbidity	49
Figure 10. Waterfall Model	51
Figure 11. User Dashboard	52
Figure 12. Temperature page	53
Figure 13. Temperature Graph	53
Figure 14. Structure of pages	54
Figure 15. Flowchart	55
Figure 16. Testing Phases	56
Figure 17. Liquid Samples	57
Figure 18. Temperature Sensor Readings	57
Figure 19. Turbidity Sample 1	57
Figure 20. Turbidity Sample 2	58
Figure 21. Turbidity Sample 3	58
Figure 22. Distance Sample 1	58
Figure 23. Distance Sample 2	59
Figure 24. Integration Testing, Sample 1	59
Figure 25. Integration Testing, Sample 2	60
Figure 26. Integration Testing, Sample 3	60
Figure 27. Parameter values obtained	60
Figure 28. Parameter in dashboard	61

List of tables:

Table 1. Functional Requirements of User	19
Table 2. Functional Requirements of Admin	20
Table 3. Non-Functional Requirements	20
Table 4. Performance Requirements	21
Table 5. Hardware and Software Requirements	22
Table 6. Summary of designed and ready components	23
Table 7. Classification of Water Parameters	24
Table 8. pH Classification	25
Table 9. Classification of Water	26
Table 10. Summary of Raspberry Pi model development	28
Table 11. pH Sensor Summary	32
Table 12. Temperature Sensor Summary	33
Table 13. Turbidity Sensor Summary	34
Table 14. Distance Sensor Summary	35
Table 15. Classification of Sensors	37
Table 16. Raspberry Pi Specification	37
Table 17. Laravel Application Summary	47
Table 18. Task Completion	50
Table 19. pH Comparison of expected and obtained values	57
Table 20. Distance Comparison of expected and obtained values	59

List of abbreviations:

ADC Analog to Digital Converter
GPIO General Purpose Input Output
I/O Input/Output
IoT Internet of Things
NTU Nephelometric Turbidity Units
OS Operating System
PCB Printed Circuit Board
RPi Raspberry Pi
TSS Total Suspended Solids
UN United Nations
URI Uniform Resource Identifier
V Volt
WHO World Health Organization
Ω Ohm

Acknowledgement

I would like to express my gratitude to my professor and mentor, Dijana Capeska Bogatinoska, Ph.D., who supported and guided me through the realisation of this project. I am very grateful to professor Ile Dimitrievski, Ph.D., who assisted me also. On this occasion I would also like to thank my family for their support and encouragement through the years of study.

1. Introduction

Water is vital for humanity, and it is one of the core elements of people to live. It is also essential for the survival of all other living beings since water is a necessity of every organism and at the same time, water is a habitat for numerous numbers of species. In today's era of constant evolution, urbanisation and introducing new solutions ensuring the safety of water is one of the biggest challenges.

The rapid pace of society and emphasising the latest advancements lead to an escalation in destroying water resources. Quality is affected by many factors including sewage, discharge from industry, agriculture, and natural disasters such as floods or droughts. Worth to mention factor is the lack of awareness among people.

According to the United Nations there are five important reasons for safe and clean water to help communities to thrive physically, mentally, economically, socially and spiritually.

1. Sustainable Development

Available water needs to meet the demands of people now without endangering water for future generations or even emptying the resources of water.

2. Socio Economic Development

Lack of clean water is a direct cause of poverty. People are often forced to spend hours finding and collecting pure water. Due to devoting to this task people cannot work and children cannot dedicate themselves to education.

3. Energy and food production

Water is used in agriculture for food production. Irrigating plants with unsafe water leads to diseases.

4. Health and Survival

When there is no pure water, communities have no other option than to use contaminated water directly from swamps.

5. Healthy Ecosystems

The Earth is composed of around 70% of water. An ecosystem is broader and includes several environments, such as water, weather, climate, etc. For the well-being of the whole ecosystem, each part needs to function conveniently.

1.1. Problem Overview

Decreasing water quality is harmful for the health, environment and economy. Consequences of water pollution are [1]:

1. Ruining of Biodiversity:

Aquatic species are very sensitive to any change in the environment, and they behave to pollution in different ways. The most extreme responses are death or migration, and less significant but not negligible consequence is reduction in reproductive capacity. Pollution is increasing the growth of algae which is increasing the difficulty for animals to find food. Because of this many species can be lost. Any animal or plant occupies space in food chain; losing any kind of specie results in destruction of the whole food chain.

2. Scarcity of drinkable water:

If water quality is not constantly maintained, it may result in losing water resources in rural as well as in urban areas.

3. Disease:

WHO [2] states the fact that around 2 billion people around the world are deprived of safe water thus they must use water from unprotected springs, lakes, ponds, rivers, streams, etc.

Whether water is used for drinking or hygiene it increases the number of diseases.

4. Mortality:

Facts collected by WHO states that around 830 000 people die each year because of unsafe drinking water, sanitation, or hygiene.

1.2. Thesis Overview

This paper is dedicated to describing the importance of water and aims to develop an application for monitoring water quality. The thesis reviews the problem from an environmental and technical aspect.

The following sections provide a detailed description of each segment of the system starting from the initial idea and motivation to the results and final conclusions.

To accomplish the project special hardware is required which needs to be programmed to give correct results. Detail description of the hardware is given in the 5th section. It shows the interconnection of devices and explains the purpose of every device used.

Above mentioned hardware is used to interact with water, that is to measure parameters, but it is preferable to display that data in a convenient way. For that purpose, the application provides a user dashboard where values are available. Another feature is the navigation menu where users can access several pages. A full description of the user interface part is given in the 8th section.

1.3. What has been done

At the very beginning, it was concluded which parameters will be measured. According to them, a survey was conducted to find the most suitable sensors, microcontroller, and additional hardware, while at the same time adhering to the budget.

Having needed hardware, the next step was to create an interconnection between sensors and the microcontroller. In the first stage, sensors were tested whether they give correct results or not. After their functionality was proved, the intention was to send sensor readings to Firebase to make them available on the webpage.

The idea when developing this application was to create a complete guide for the users in a way that they can find everything related to the system. For that purpose, the application includes pages where news related to the system are shared, users can contact providers, access frequently asked questions, read a full guide of using the system, etc. In this way, the user can learn the application and decide whether to use it or not.

The application allows registering of new users or logging of an existing user. When a user logs he/she is redirected to the dashboard which visually represents collected data through tables and charts.

1.4. What has been achieved

The traditional way of examining water includes taking samples and investigating them in laboratories. To reduce the time for collecting water, cost, and human resources the system was developed in a way it operates on remote stations and sends data to the user.

Achievements of the system include measuring on defined periods, displaying a notification on every parameter as well classifying water according to current (last measured) data, and real-time analysis of data.

The final product was a small system that can be placed on a water resource to constantly measure desired parameters. The system is a prototype that can be improved in many ways and be used in large-scale problems.

2. State of the Art

In the Introduction Section sources [1] and [2] were used to learn more about the influence of water on the environment as well as every life being. WHO has conducted many researches regarding to water importance and those numbers are used to reflect the impact of water pollution on human health.

In Project Overview Section research paper [4] was used, where the author has taken into consideration several alternatives for measuring water quality. That information was used to justify the intention of developing a system based on sensors. Comparing alternatives, it was concluded that the proposed system surpasses other solutions from several aspects.

In General Definition Section following resources were used: [6], [7], [8], [9] and [10]. Those articles were used to learn detailly about what water quality is and its parameters. Article [10] was used for summary where the author has classified water according to its parameter's values.

In the Hardware Description Section several books and articles were examined to explain the hardware used. In books [11], [12] and [13] authors have thoroughly explained everything related to Raspberry Pi including origin, operating capabilities, development of models, models comparison, etc. Books [14], [15] and [16] were used to explain sensors as main components in the system. The knowledge gained from these books was implemented for classifying sensors used in the proposed system.

3. Project Overview

This section describes every step made before starting with the development of the system. Since system incorporates both hardware and software, they need to be inspected from every aspect, especially the hardware. Both subsystems need to cooperate and function in the desired manner.

3.1. Motivation

Regardless of whether water is used for drinking, domestic use, recreation or food production, it must be safe and readily available. Water is important for public health. It is crucial to establish and maintain water quality balance. If not, human health and the whole ecosystem could be severely damaged.

Today providing pure drinking water has become a significant challenge worldwide. Many international governing bodies such as the UN or WHO are involved in efforts to ensure safe water everywhere in the world. On 28 July 2010, the United Nations General Assembly adopted a decision which states that access to safe and clean water is a human right, and it is essential for full enjoyment of life [3].

The goal of this project is designing a prototype which is practical and low-cost. For that purpose, IoT technology was used. With this approach several sensors collect data, data is being monitored and analysed and inspected water sample is categorised.

3.2. Environmental Impact

Water is one of the most utilised natural resources by humans, animals, and plants for survival. Effects of water pollution are not just on water and ecosystem themselves, but it affects every living being, especially the ones for which water is habitat. As water pollution increases it is very expected that occasionally any water consumer will encounter unsafe water.

Water and climate are very closely related, affecting each other. Harmful substances become part of the water cycle. This affects the pH level of raindrops and results in acid rain.

Acid rain is however natural phenomenon, but the possibility of its occurrence increases with increasing pollution. It is a long-term effect because rain flows in soil affecting plants, and with the water cycle it is possible that the same water evaporates in air.

Having excessive concentration of chemical nutrients leads to dense growth of algae. This reduces oxygen in water and obstructs sunlight required for a number of vital processes. On the other hand, water bodies with low oxygen are more susceptible to climate changes.

Increasing the pollutants creates acidic lakes, algae blooms, deficit of oxygen, rising in temperature which make water inappropriate habitat. This forces animals to migrate and results in dead zones.

3.2.1. Impact on Ohrid Lake

Ohrid Lake is known as the deepest and oldest lake in Europe. It is considered as a historical and cultural world heritage site and it is under UNESCO's protection.

Lake is habitat for a number of endemic but also relict species. Those species are adapted to specific conditions. By changing their environment, they can either adapt or disappear. But they have existed in precisely defined conditions for about 3 million years and adjusting to a new drastically modified habitat is a very hard process.

In recent years there is an obvious increase of green algae. As previously mentioned, algae have a considerable impact on water conditions. Its presence threatens the survival of endemic species of algae. Green algae are result of wastewater and sewage.

Wastewater and sewage are an enormous environmental problem affecting the lake's flora and fauna but also human health. Samples tested from several points categorised water in 4th and 5th category.

It is also important to follow the level of water because adverse weather conditions could result in mixing sewage with atmospheric water and flowing that water into the lake.

Aforementioned reasons could put Lake on List of World Heritage in Danger.

3.3. Technical Challenge

The biggest challenge when developing this project was from the hardware design aspect. All sensors must be connected to Raspberry Pi. RPi does not have analog inputs, and for that purpose ADC was used. Two of the sensors: turbidity and pH are analog, they are first connected to ADC and ADC is connected to RPi.

Another specification of Raspberry Pi is that it is a 3.3V device, meaning that it cannot tolerate 5V input voltage. The board has only two 5V pins: one of them was used for ADC; other for ultrasonic sensor. ADC supports voltage from 3.3~5.0V. ADC, pH and turbidity sensor are from Gravity Series from DFRobot supplier, and their interconnection is straightforward since sensors just need to be plugged on the converter.

3.4. Limitations

The performance of the system depends on the components used. All components need to be selected in a way that they ensure compatibility and reliability. During selection of components their cost was taken into consideration. Due to limited budget only four sensors are used to inspect water quality.

Another very important obstacle was that components used must be waterproof. Testing water samples must be very careful to avoid damaging sensors.

One more thing to consider is that sensors have different operating temperatures. In order not to disturb the functioning of the sensors they must be tested in a controlled environment meaning that temperature of the sample must be within allowed range. Operating capabilities of each sensor are explained in 5th section.

3.5. Alternatives

There are several options on how one can measure water quality [4].

3.5.1. Laboratory

The most famous and most used solution includes testing water samples in the laboratory. This method is slow in delivering results because water should be collected and transported to the laboratory. Those results may not be reliable because water can be affected during transportation. Changing the value of one parameter can have an impact on other parameters. It is proved that changing temperature value changes pH value; if the temperature increases pH decreases. Another disadvantage of this solution is that it inspects a small amount of water.

3.5.2. Remote Sensing

This method observes water quality without any contact with the water body. Remote sensing is a process of detecting physical characteristics of an area (water body) by measuring reflected and emitted radiation. Remote sensors collect data by detecting the energy that is reflected from Earth. These sensors can be attached to satellites or aircraft [5].

The method requires special training to analysed images. Sometimes image which is analysed can be affected by another event that has not been taken into consideration during training. Also, a powerful sensing system that emits its own electromagnetic radiation could affect the sample which is investigated.

3.5.3. Change in Organism's Behaviour

This method inspects water by determining the behaviour of living organisms. All organisms have precisely defined living conditions, and they are highly affected by changing any parameter. At the beginning, the behaviour of organisms is monitored under normal

circumstances and then the response for controlled change is recorded and recognized. When water is tested if a recorded change in animal behaviour is found it means that some change in water parameters occurred.

A key factor in this methodology is living organisms. By following their behaviour information about their habitat (water) is obtained. However, observed information could be wrong because organisms may be affected by another factor. Also, this methodology is not desired because the goal of the system is to protect organisms, not to include them in testing.

3.5.4. Sensor-Based Solution

This method utilises sensors that collect data, sending data to the network and interface for the visual representation of collected data. This approach is used for developing a proposed system which is detaily explained in this paper.

3.5.5. Preferred Solution

Using IoT as a solution for inspecting water quality overcomes some of the disadvantages previously mentioned. The system has following advantages:

- Reduces the need for human impact and therefore reduces human errors.
- Increases reliability of inspected data by eliminating the possibility of a sample being contaminated during transportation.
- User-friendly representation of collected data.
- Faster response to changes in a water sample.
- The limitlessness of usage since it can be adapted to be used for different purposes.
- Constant monitoring of collected data.

3.6. Choosing Microcontroller

For successful implementation of the system, it is crucial to choose the proper microcontroller. The range of available microcontrollers is wide, and they mainly distinguish in scalability, functionality, flexibility, cost, power requirements, etc.

For this project, two options for microcontroller were observed: Arduino and Raspberry Pi. After research was made it was concluded that Raspberry Pi is the preferred option. RPi surpasses Arduino from several points of view:

- Architecture – RPi is a microprocessor because it runs an operating system on which multiple tasks can be performed; Arduino is a microcontroller and does not run an operating system and it is programmed for a single task.
- Connectivity – RPi has built-in Ethernet and Wi-Fi support while Arduino requires additional modules or shields to connect to the Internet.
- Expansion – both boards can be expanded with additional modules for adding more functionalities. Arduino uses shields for internet connectivity, SD Card,

LCD, or other, while Raspberry Pi provides all these features on the original board without additional shields. Adding many modules makes hardware complex and it loses portability possibility.

- Storage – Raspberry Pi provides a larger memory capacity.

3.7. Project Requirements

This subsection is dedicated to demands of a project to function properly. Requirements convey the expectations of users from the product.

3.7.1. Functional Requirements

Functional requirements are actions that must be performed by the system in order users to accomplish their tasks and utilise its capabilities. As they are two types of users, both have different functional requirements. Functional requirements for both types are given in tables 1 and 2.

USER		
ACTION	REQUIREMENT	EXPLANATION
Login	Respond with a message.	When user attempts to login, he must enter correct credentials (email and password). If they match user logging is successful, if not system gives error message to user.
Register	Respond with a message.	When user attempts to register, system checks whether provided email exists in database or not and checks whether other fields satisfy conditions. For successful registration email does not need to exist in database, password must be at least 8 characters and password confirmation must match. Otherwise, system will give error message.
Login	Redirect	When user logs, he must be redirected to his dashboard.
Contact Supplier	Respond with a message.	When user tries to send contact form to the supplier, system checks validity of form's fields. If all is within constraints, meaning that all fields are filled, and email is in correct format, system gives successful message to the user; if not warns what to do in order to complete an action.

Table 1. Functional Requirements of User

ADMIN		
ACTION	REQUIREMENT	EXPLANATION
Login	Respond with a message.	When an admin attempts to login, he must enter correct credentials (email and password). If they match, admin logging is successful; if not, the system gives an error message to the admin.
Login	Respond with a message.	When admin logs, he must be redirected to his dashboard.
Create new post	Respond with a message.	When an admin wants to write a new post, he is given a form with fields (title, content, image). For successful publishing of that article several conditions must be satisfied: all fields are required, content must be at least 50 characters and file must be image: jpeg, jpg, png, gif, svg).
Edit/Delete post	Respond with a message.	When a user attempts to edit or delete a post, the system must give him a response for successful editing or deleting a post.

Table 2. Functional Requirements of Admin

3.7.2. Non-Functional Requirements

This type of requirement defines a quality attribute of a system. They describe how well the system should function. Table 3 is a summary of non-functional requirements.

REQUIREMENT	EXPLANATION
Usability	System can be used for different purposes and, by user with different technical knowledge.
Accessibility	Available in PC and mobile mode.
Interoperability	Data must be successfully exchanged between hardware and database, and between database and application.
Learnability	System needs to be easy to learn how to work with.
Portability	It should be easy to move entire system from one place to another.
Scalability	System functioning needs to be equally efficient with increasing the number of users, or with attaching more sensors.
Security	When user logs he is redirected to user dashboard and, if he tries to access admin dashboard, he will be redirected to <i>Access Denied</i> page, saying that user has not authority for that action. If admin tries to access user dashboard, he will get same message.

Table 3. Non-Functional Requirements

3.7.3. Performance Requirements

This type of requirements refers to operating capabilities of a system in terms of accuracy, response time, memory. Performance capabilities of both hardware and software are crucial for proper functioning of system and proper management of data. They are detaily explained in table 4.

REQUIREMENT	EXPLANATION
Correct and fast sensor readings	Most important feature of a system; sensor readings must be correct or with minimal error rate.
Constant readings of sensors	Sensors are programmed to read on particular interval, more exact on every 5 minutes.
Updating database on every n minutes	System must insert into database every value measured; update database every 5 minutes.
Retrieve last measured data	Dashboard page refreshes on every 1 minute. System must return correct values (last updated record).
Categorise sample properly	System compares measured values with reference values and based on the result classifies the sample. For example, if $\text{pH} > 7$ system must classify it as base and no other outcome is acceptable.
Store data properly	System organises data into a table for better visual representation. Records in data are sorted in descending order based on date and time, meaning that last measured record is first in a table.
Correct data manipulation	Parameter pages display maximum, minimum value, their difference, and current value. System must return correct values.

Table 4. Performance Requirements

3.7.4. Hardware and Software Requirements

This type of requirements includes components that are necessary and without which the system cannot operate. Those requirements are prerequisites for the system. Summary of those requirements is given in table 5.

HARDWARE		SOFTWARE	
REQUIREMENT	EXPLANATION	REQUIREMENT	EXPLANATION
Power Supply	This is mandatory, RPi must be plugged on power.	Python	For programming Raspberry Pi and sensors.
Ethernet Cable	Connects RPi to laptop	PGP/Laravel	For storing and organising data, and to make sure that everything on the client side works.
SD Card	For installing OS.	HTML/CSS/JS	For user friendly appearance of application.
Breadboard	For connecting distance sensor, to avoid damaging Pi.	Firebase	For storing sensor data.
Correct value resistors	To reduce voltage, values of resistors are calculated with proper formula.	MySQL	For storing the rest of the data.

Table 5. Hardware and Software Requirements

3.8. Analysis

The system has five types of connections. Each interface has different purpose, and each is mandatory for proper functioning of a system. For one communication to be done, the previous must be completed. Interfaces are given in figure 1.

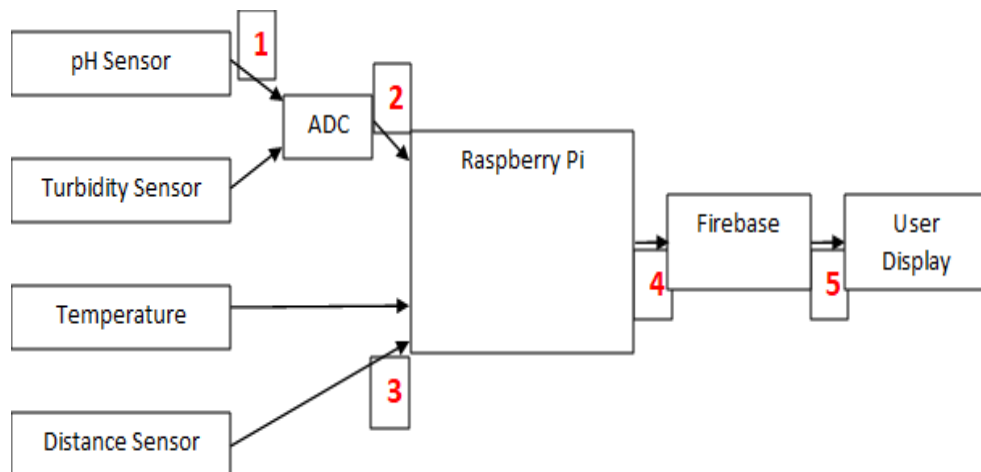


Figure 1. System's interfaces

User observes measured data on his device. Data is organised by system, but data must exist and must be taken from a database. Data in the database arrives from RPi. To collect data, Pi interacts with sensors.

3.9. Deliverables

COMPONENT	DESIGNED	READY
Power Supply		X
Ethernet Cable		X
SD Card – OS Installing	X	
Raspberry Pi		X
Ph Sensor		X
Temperature Sensor		X
Turbidity Sensor		X
Distance Sensor		X
ADC		X
ADC – RPi Connection	X	
pH Sensor-ADC Connection	X	
Turbidity Sensor - ADC Connection	X	
Temperature Sensor - RPi Connection	X	
Breadboard		X
Resistors		X
Distance Sensor – Breadboard Connection	X	
Voltage Divider	X	
Breadboard – Raspberry Pi Connection	X	

Table 6. Summary of designed and ready components

4. General Definition of Water Quality

Water quality describes a broad range of parameters related to how water is categorised. It is related to physical, chemical or biological conditions of water [6]. Usually,

reference values of measured parameters are determined according to set of standards which is acceptable for the desired purpose of water used. Water is used in many applications and for all of them water quality is shaped differently. Standards for drinking, industrial, agriculture, aquaculture water or any other type of water are different.

4.1. Water Parameters

Obviously, water is present in any aspect of human life. Although parameters value can differ from one application to another, the water parameters that are used to classify water are same for any sample. Three main classes of water parameters are: physical, biological and chemical [7]. Table 7 shows elementary classification of parameters.

PHYSICAL	CHEMICAL	BIOLOGICAL
Turbidity	pH	Bacteria
Temperature	Alkalinity	Algae
Colour	Chlorine	Viruses
Taste and Odour	Hardness	
Solids	Dissolved Oxygen	
Electrical Conductivity	Biological Oxygen Demand	

Table 7. Classification of Water Parameters

Proposed system determines the condition of water based on four characteristics: pH, temperature, turbidity and depth. Following subsections describe each parameter.

4.1.1. pH

It is one of the most important factors when measuring quality. By measuring pH, it is measured how much acidic or basic water is. According to the chemistry background, acidic water contains more hydrogen ions. On the other hand, basic water contains more hydroxyl ions [8].

The spectrum of pH values ranges from 1-14. The solution of 1 will be categorised as acid while of 14 as base. Normal drinking water's value is 7. Most of the water plants and animals require an environment of specific pH. Acidic water can harm fishes and reduce the number of eggs [7] due to its corrosive effects. Water with a pH of 11 or higher causes irritation. Table 8 displays pH values of several substances.

RANGE	TYPE	SUBSTANCE
<1	Acidic	Battery Acid
1.0 - 1.5		Gastric Acid
2.5		Vinegar
3.3 - 4.2		Orange Juice
5.0 - 5.03		Coffee
6.5 - 6.8		Milk
7	Neutral	Pure Water
7.5 - 8.4	Basic	Sea Water
11.0 - 11.5		Ammonia
12.5		Bleach
13.0 - 13.6		Lye

Table 8. pH Classification

4.1.2. Temperature

By measuring temperature, the degree of heat present in object or substance is determined. It depends on kinetic motion of molecules in that substance. As molecules move and collide, they transfer their kinetic energy to thermal. Changes of temperature in water is result of molecular motion which defines internal thermal energy of water [4]. Temperature can be measured according to different scales: Celsius, Fahrenheit and Kelvin.

Temperature range is extremely wide. When measuring temperature as a reference value is taken the value that is considered suitable in given situation. Reference value of drinking water is different from temperature of fishponds for example, or industrial water temperature cannot be compared with the temperature of the lake.

Temperature is very important because it affects status of water. It affects conductivity, chemical and biological reactions, maximum dissolved oxygen concentration, pH etc. Temperature is crucial for organisms within ecosystems because they have preferred regimes and they are not adjustable to changes.

4.1.3. Turbidity

Turbidity refers to water clearness. When measuring turbidity, it is observed the ability of light to pass through water. It measures the numbers of particles found in water. Turbidity is caused by plant waste, clay, sand or any other particle. As greater the concentration of particles is as greater the turbidity is. It is measured in NTU [9].

Excess turbidity can cause many illnesses especially if water is used for drinking. Also, high turbidity water affects the overall wellbeing of organisms living within.

4.1.4. Water Level

This parameter describes the importance of the depth of the water. It does not measure chemical or biological condition of sample but measures the level of water in closed system or in water surfaces like rivers, lakes, seas or oceans.

Value depends on the system where quality is measured. Reference value of depth varies for different purposes. Monitoring water level is used to detect absence of water in system or exceeding borders or checking whether level is within permissible limits.

4.2. Classification of Water

Every source of water can be exposed to contamination due to agricultural, industrial or domestic activities. Water can be harmed by heavy metals, pesticides, chemicals, oils and any other pollutants.

In previous parts parameters were explained taking into consideration the chemistry and physics behind their behaviour. Depending on their values water can be categorised into four types summarised in table 9 [10].

TYPE	SAFE TO DRINK	SAFE IN HOUSEHOLD	PRESENCE OF CHEMICALS	PRESENCE OF VIRUS/BACTERIA
Potable	✓	✓	X	X
Palatable	✓	✓	✓	X
Contaminated	X	X	✓	X
Infected	X	X	X	✓

Table 9. Classification of Water

4.3. Summary

Water is most important element in our lives. Uses of water are countless and there is no domain where water is not included. Referring to table 9 in previous section, not all water is suitable for drinking, watering or household use. However, this does not make the water unusable or polluted.

Water quality can be influenced by many factors including land use, land cover, land management, climate, rainfall intensity, chemicals and many more. Quality varies from time

to time, and it is very important to follow the conditions of water constantly especially if examined water has an impact on live organisms.

5. Hardware Description

Hardware is essential and one of the integral parts of the project. To achieve initial intention appropriate devices must be chosen. This part shortly describes every piece of hardware used and explains their specifications. In the last subsection there is visual representation of the system.

5.1. Raspberry Pi

Raspberry Pi is single-board computer. It is computer as any other desktop, laptop or smartphone but it comes in compact and low-cost solution. The size of a computer is equivalent to credit-card size, and it is built on single PCB. Despite its tiny size, this device is capable for tasks that any bigger and more powerful computer can accomplish though not as quickly [11].

5.1.1. Origin and history of Raspberry Pi

The very first intention to develop Raspberry Pi boards can be found in the University of Cambridge's Computer Laboratory in 2006. The concerns among several computer scientists were motivation of developing today's worldwide used devices. The idea was to teach students to understand computers rather than only using them for different purposes. The team worked on creating cheap and accessible device in order to bring students closer to technical aspects of computers.

The first Raspberry Pi model was *Raspberry Pi Model B* introduced in 2012. Since then, the board has been revisited many times and today it is available in A, B, A+ and B+ as well. Comparing models, it is concluded that B is more powerful and provides ethernet connectivity while A is cheaper and suitable for low-scale projects [12]. The latest version is Raspberry Pi 4 B available with 1, 2, 4 and 8GB RAM. Table 10 is summarization of RPi models developed over the years [13].

FAMILY	MODEL	YEAR	SoC	MEMORY	ETHER NET	WIRE LESS	GPIO
Raspberry Pi 1	B	2012	BCM2835	256 MB/512 MB	Yes	No	26
	A	2013			No		
	B+	2014		256 MB	Yes		40
	A+	2014		512 MB	No		
Raspberry Pi 2	B	2015	BCM2836/7	1 GB	Yes	No	40
Raspberry Pi Zero	Zero	2015	BCM2835	512 MB	No	No	40
	W/WH	2017				Yes	
	2 W	2021	Raspberry Pi SiP RP3A0	1 GB			
Raspberry Pi 3	B	2016	BCM2837 A0/B0	1 GB	Yes	Yes	40
	A+	2018	BCM2837B0	512 MB	No		
	B+			1 GB	Yes(Gigabit Ethernet)		
Raspberry Pi 4	B	2019	BCM2711	1GB/2GB/4GB/8GB	Yes (Gigabit Ethernet)	Yes (Dual Band)	40
	400	2020		4GB			
Raspberry Pi Pico	N/A	2021	BCM2711	264 KB	No	No	40
	W/WH	2022	RP2040			Yes	

Table 10. Summary of Raspberry Pi model development

Ten years after the first Raspberry Pi was sold, the total number of boards distributed worldwide is approximately 40 million, taking into consideration any available model. Rapid demand for boards as well as shortage of components caused limitation to the production of boards.

5.1.2. A guided tour on Raspberry Pi

Raspberry Pi computer does not look like traditional computers. It is small card that comes unpacked such that any component is visible. Besides its components and their extraordinary capabilities Pi provides possibility to plug various peripherals. Figure 2 shows the layout of Raspberry Pi.

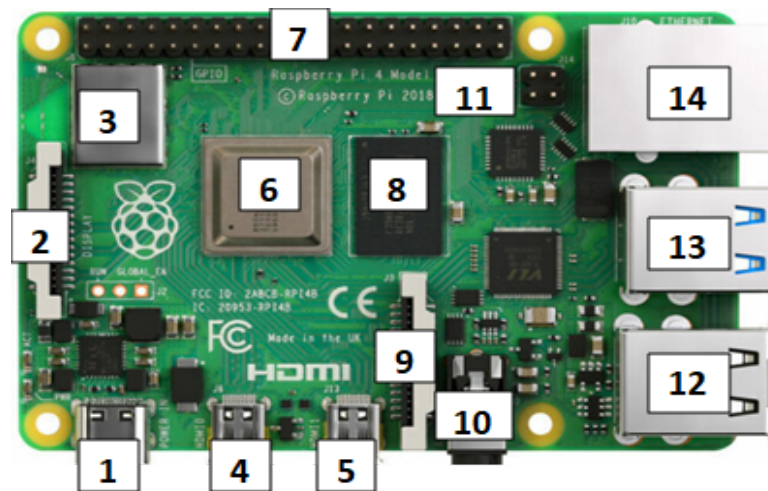


Figure. 2 Raspberry Pi Layout

1. USB Type-C Power Port – it is used to connect RPi to power source. For that purpose, any mobile charger can be used but it is recommended to use official power supply.
2. Display Connector – designed for use with a Raspberry Pi Touch Display.
3. Radio – gives ability to communicate with devices wirelessly. It can act as two components: Wi-Fi radio, for connecting to computer network; and A Bluetooth radio for connecting to peripherals or sending and receiving data from nearby devices.
4. HDMI 0 (High-Definition Multimedia Interface) - used to connect RPi to computer monitor, TV or projector. The port carries quality audio and video signals.
5. HDMI 1
6. System-on-chip (SoC) - integrated circuit that includes CPU and GPU.
7. GPIO (General Purpose Input/Output) - 40 metal pins which allow Pi to interface with additional hardware as sensors, buttons, LEDs etc.
8. RAM – it holds everything that is done on Raspberry Pi and that will be written to SD Card only if it is saved.
9. Camera Connector- allows to use specially designed camera module.

10. AV jack (audio-visual) - supports audio and video signal.
11. PoE (Power over Ethernet) - used to connect PoE HAT which allows Pi to receive power from a network connecting rather than USB port.
12. USB ports – used to connect any compatible peripheral to Raspberry Pi. Through the ports keyboard, mice, flask drives can be attached to the Pi.
13. Ethernet Port – through this port Raspberry Pi is connected to computer network using a cable.

Before starting to use Raspberry Pi a procedure of installing OS must be conducted, unless SD Card has been pre-installed. To setup Pi several components are required: flashed SD Card, power supply and Ethernet Cable. Raspberry Pi runs on Linux OS supporting several of its distributions including official Raspbian OS, Ubuntu Mate, Snappy Ubuntu Core, etc.

5.2. Sensors

A sensor is a device which detects and responds to different types of signals and convert them to electrical signal. It can be stimulated by physical, chemical or biological input signal. After signal is being processed it gives electrical signal such as current or voltage as response [14].

Figure 3 describes how sensors work. Generally, all sensor's processing starts by taking signal from environment. Depending on purpose of the sensor it can take temperature, pressure, humidity, resistance or other parameter as an input. First step is done by *sensing unit* by detecting changes in parameters and generating electrical signals. Signal is then controlled by *signal conditioning* unit to verify that it satisfies conditions of next-level processing. After successfully finishing signal examination it is converted from analog to digital mode and sent to microprocessor. *Application Algorithm* unit process received data based on program which is previously written here and generates output. This part implements formulas for converting electrical signal to value by applying different mathematical formulas and using different functions. After result is obtained it is convenient to represent in human readable form, which is task of *Local User Interface*. It is device that monitor values measured by sensor. *Memory* is used to store received and processed data. In the step done by *Communication Unit*, output signals are transmitted to main station.

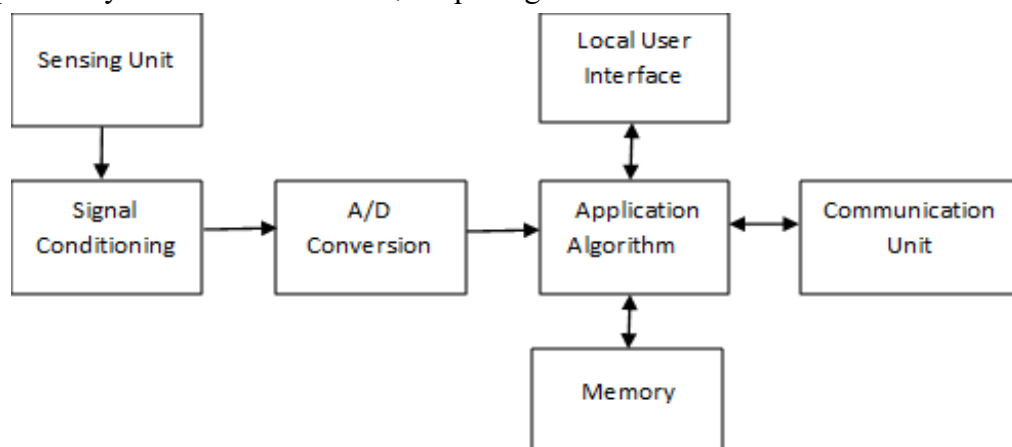


Figure 3. Sensor working principle

5.2.1. Classification of sensors

Sensors can be classified based on several criteria. Combining information from following resources [15] and [16] this paper provides several aspects for classifying sensors:

1. Analog or Digital

The mayor difference between them is in type of output they produce. Analog sensor produces continuous output signal or voltage which is proportional to quantity which is measured, while digital produces discrete digital output signal or voltage which is digital representation of quantity to be measured.

2. Active or Passive

Sensor belongs to one of the groups depending on whether it requires external power source to operate or not. Active sensors are the one that require external power known as excitation signal. Sensor modifies that signal and produces output. On the other hand, passive sensors do not require additional source and directly generate an electrical signal.

3. Contact or Non-Contact

Classification according to this methodology is obvious. Contact sensors require physical contact with parameter they measure, while non-contact do not require contact.

4. Absolute or Relative

Absolute signal produces a signal which is independent of the measurement conditions. It provides absolute reading of parameter it measures. Opposite of independency of absolute signal relative sensor produces a signal that relates to some special case. Output depends on fixed or variable measurement.

5. Parameter it measures

Sensors can be used to examine various parameters. One sensor can be dedicated to measure single stimulus or more. Depending on its dedication sensor can obtain temperature, humidity, distance, pressure or any other value.

5.2.2. Application of sensors

Sensors are very popular today and they find wide range of applications. They are used in medicine, automotive industry, aerospace, agriculture, even in our everyday live. However, sensors are present in many more branches. In this paper, three applications will be explained detailly as they are introduction to the main part of this article which is dedicated to describing the whole process of water quality monitoring system development.

1. Embedded Systems

Embedded system is system specially designed for particular purpose. The system contains two subsystems: hardware and software. Usually, embedded system is part of larger system, and it is dedicated to solving only specific task. As any other computer it has its own processor, memory and possibility for attaching various I/O devices.

The reason of mentioning this application is to make analogy with previously mentioned project. The intention of that project is to measure and monitor quality of a water, that is it has specific purpose. As a controller Raspberry Pi microcontroller is used with its memory and capability for processing received data. RPi supports various I/O devices and for the purpose of project several sensors are attached to GPIO.

2. IoT – Internet of Things

IoT is group of billions of objects where any device has its own intelligence, meaning it is dedicated to performing specific task. All devices share collected and processed data through IP networks [17]. IoT is big network supporting wide range of different devices as members. Sensors are very important for creating solutions using IoT.

The reason of mentioning IoT is that suggested system use exactly four sensors to observe any change in measured water parameters and to send them in database for further processing.

3. Smart City

A smart city refers to city which deploys large number of sensors to collect data from different sources in order to improve efficiency and quality of life. Concept of smart city emphasizes sustainability and energy efficiency. To achieve this, automation and connectivity to internet is mandatory. By automation is meant that minimal human input is required since devices are programmed in a way that they collect data all the time or they follow schedule. Connectivity to internet refers to previously mentioned IoT, that is all data collected is sent somewhere [18].

Using systems as water quality monitoring in large extent contribute to building smart cities. System can be used for controlling many aspects: pure drinking water, fishponds, smart agriculture etc. It must be taken into consideration that implementing this solution is only one segment of smart city.

Above mentioned implementations complement each other. Each part is one step closer to building smart city. It starts by making a system that uses several sensors, considering system as embedded since it has specific purpose. Having different solutions based on sensors contributes to create IoT, network where every device communicates with each other. All those embedded systems and IoT devices make a city to be so called smart. The process is visually represented in figure 4.

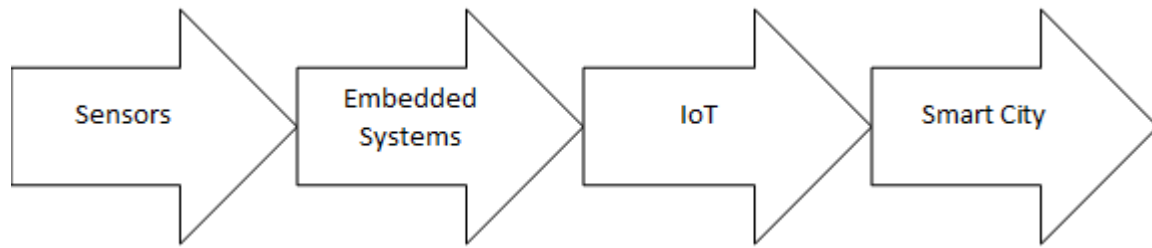


Figure 4. Relationship between sensors, IoT and Smart City

5.3. pH sensor

A pH sensor is a scientific device used to accurately measure acidity and alkalinity in water and other liquid substances. Sensor is bought from DF Robot supplier, and it comes with BNC connector, sensor circuit board, analog cable and probe. Since the sensor is analog it must be connected to analog to digital converter. Table 11 is summary of sensor's specifications.

SPECIFICATION	VALUE
Module Power	5.00 V
Measuring Range	0-14
Measuring Temperature	0 - 60 °C
Accuracy	$\pm 0.1\text{pH}$
Response Time	$\leq 1\text{min}$

Table 11. pH Sensor Summary

5.3.1. How does pH Sensor work?

Ph sensor is made of glass and on the bottom it has a bulb where sensor is located. Glass electrode has bulb which is designed to be responsive to hydrogen-ion concentration. When probe is immersed in solution that is tested hydrogen ions in solution exchange for other positively charged ions on the glass bulb and create electrochemical potential across the bulb. The electronic amplifier detects the difference in electrical potential between two electrodes and convert it to pH units.

5.4. Temperature Sensor

A temperature sensor is device used to measure the temperature of its environment and convert the input data to electronic data for further processing. Sensor is bought from DF Robot supplier, and it comes with digital sensor cable, terminal sensor adapter and waterproof digital temperature sensor. Since the sensor is digital there is no need for ADC. Table 12 is summary of sensor's specifications.

SPECIFICATION	VALUE
Module Power	3.0 -5.5 V
Measuring Range	-55 – 125 °C
Accuracy	± 0.5 °C
Response Time	≤ 750 ms

Table 12. Temperature Sensor Summary

5.4.1. How does Temperature Sensor work?

The sensor belongs to group of one wire sensors. It has ADC which converts analog signal to the digital output with the resolution up to 12 bits. It has embedded 4.7 kΩ resistor which is used as a pull-up resistor.

5.5. Turbidity Sensor

Turbidity sensor is a device used to measure the level of turbidity, or opaqueness. It is bought from DF Robot, and it comes with four parts: sensor adapter, probe, cable, analog cable. It is in a group of analog sensors which require ADC. Table 13 is summary of sensor's specifications.

SPECIFICATION	VALUE
Module Power	5.0 V
Operating Temperature	5°C~90°C
Output Method	Analog/Digital
Response Time	≤ 500 ms

Table 13. Turbidity Sensor Summary

5.5.1. How does Turbidity Sensor work?

The working principle of sensor is based on using light to detect particles in water. Probe sends light beam in water and light is scattered by suspended particles. Light detector is usually placed at 90-degree angle in relation to light source. Detector measures the amount of light that is reflected. The density of particles in water is determined by amount of reflected light. It measures light transmittance and scattering rate which changes based on total suspended solids in water. TSS and turbidity are proportional.

5.6. Distance Sensor

Distance sensor is device for measuring distance from place where it is positioned to the first obstacle it encounters. It is bought from Logging supplier, and it comes as single part. However, for connecting distance sensor to RPi several additional hardware is required: resistors, jumper wires and breadboard.

SPECIFICATION	VALUE
Module Power	5.00 V
Measuring Range	(0,3-4) m
Accuracy	± 0.3 cm
Response Time	~ 32 ms

Table 14. Distance Sensor Summary

5.6.1. Resistor

Resistance is opposite of the current flow through electrical circuit. As a voltage is applied across a conductor or wire, free electrons start move in some direction. During this process electrons collide with other particles. To reduce the flow of current or to divide the voltage, resistors are used in electronic circuits [19].

The current in electronic circuits is small usually measured in milli-Amperes, micro-Amperes or nano-Amperes and resistors used should be measured in kilo-ohms. Resistors are available in different values and different wattage.

5.6.2. How does Distance Sensor work?

Distance sensor is also known as ultrasonic sensors. It comes with separate transmitter and receiver. The first step in determining distance is done by transmitter (TRIGGER) which transmits ultrasonic signals. Signals travel until it hit the obstacle and then bounce back. When signals arrive back, the receiver (ECHO) accepts them. The distance is determined based on amount of time taken to transmit and receive signal. Sensor sends 5V signal on the ECHO pin of the RPi and since Pi works at 3.3 V resistor(s) is used to reduce the voltage.

5.7. Wiring Components

The list of components required is wide and it includes:

1. Raspberry Pi 4 Board
2. Analog to Digital Converter
3. Ph Sensor
4. Turbidity Sensor

5. Temperature Sensor
6. Distance Sensor
7. Three pieces of 1k Ω resistor
8. Jumper Wires
9. Breadboard

The procedure of attaching components to RPi is as follow:

1. ADC is connected to board in following order:
 - GND Wire (Black) to Pin 6
 - VCC Wire (Red) to Pin 4
 - Data Wire (Green) to Pin 3
 - Clock Line (Blue) to Pin 5
2. Ph and Turbidity sensors are analog, they must be connected to ADC. Gravity sensor series provide very convenient way of attaching gravity analog sensors to the converter. Both sensors have three wires, and they are plugged to channel 0 and 1 of the ADC.
3. Temperature sensor is digital, and it is directly connected to board in following approach:
 - GND Wire (Black) to Pin 9
 - VCC Wire to pin 1
 - Signal Line (Green) to Pin 7
4. As mentioned previously distance sensor requires additional components. To calculate what resistor value is required to reduce the voltage equation was used.

$$V_{out} = V_{in} \times (R_2 / (R_1 + R_2)) \quad (1)$$

$$V_{out}/V_{in} = R_2 / (R_1 + R_2) \quad (2)$$

Any combination of resistors that satisfies equation can be used. In proposed system as R1 is chosen 1k Ω resistor. By changing R1 by its value the value of another resistor is obtained. Vin is changed with 5 because input voltage is 5V, Vout with 3.3 because it must be decreased to 3.3V for output voltage.

$$3.3/5.0 = R_2 / (1000 + R_2) \quad (3)$$

$$0.66 = R_2 / (1000 + R_2) \quad (4)$$

$$0.66 \times (1000 + R_2) = R_2 \quad (5)$$

$$660 + 0.66R_2 = R_2 \quad (6)$$

$$660 - 0.34R_2 \quad (7)$$

$$1941 = R_2 \quad (8)$$

The value for second resistor is approximately 2k Ω but the system uses two 1k Ω resistors in serial connection. Figure 5 shows how are components connected.

5.8. Summary

For the purpose of project four sensors were used. They are classified according to previously mentioned methods (see table 15).

	pH	TEMPERATURE	TURBIDITY	DISTANCE
Analog	✓		✓	
Digital		✓		✓
Active				
Passive	✓	✓	✓	✓
Contact	✓	✓	✓	
Non-Contact				✓
Absolute		✓		
Relative	✓		✓	✓

Table 15. Classification of Sensors

As a microcontroller Raspberry Pi 4 Model B is used. Its performances are described in table below.

Raspberry Pi 4 Model B	
Processor	Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
RAM	4 GB
Wireless Network	2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0
Wired Network	Gigabit Ethernet
USB Ports	2 USB 3.0 ports; 2 USB 2.0 ports
GPIO	40 pins
Video Outputs	2 × micro-HDMI ports (up to 4kp60 supported)
Supported Codecs	H.265 (4kp60 decode), H264 (1080p60 decode, 1080p30 encode)
Operating Temperature	0 – 50 °C

Table 16. Raspberry Pi Specification

This section is concluded with the sketch of the hardware part of the system showing how are all components connected.

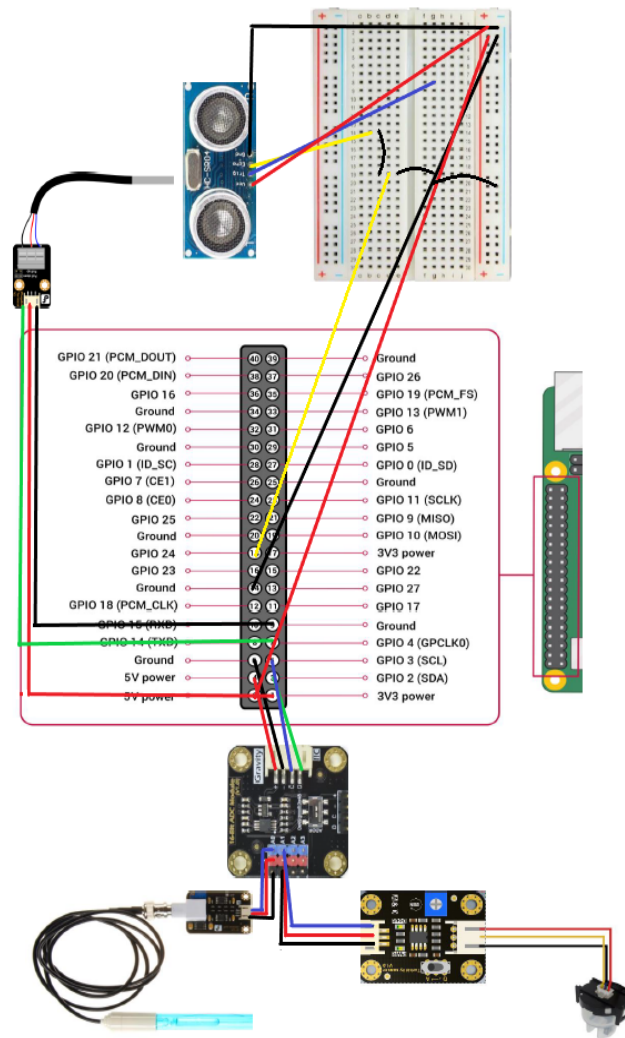


Figure 5. Wiring Components

6. Programming Language Justification

For the development of application few programming languages and techniques were used. The whole application is divided in three parts: Raspberry Pi, Front-End and Back-End.

- Raspberry Pi

Raspberry Pi operates on Linux. Linux comes in few distributions including Debian and official supported Raspberry Pi OS is based on it. To program Pi, Python scripting language is used [20]. Python is chosen as main language by Raspberry Pi Foundation because it is powerful, used for different purposes and easy to use. Python is preinstalled on Raspberry Pi OS. The communication between user and Pi is done using terminal, known as command line interface.

All hardware-related codes are written in scripts.

- Front-End

For creating user friendly application HTML, CSS and JS are used. Every page is written in HTML and there are styled with CSS. Pages contain JS code to make them dynamic and interactive.

- Back-End

As back-end programming language PHP is used. More specific, Laravel PHP framework is implemented. The reason of choosing Laravel was because it provides every feature required to build modern web application: routing, validation, caching, queues, file storage, middleware and allows very easy communication with database. Laravel is MVC (Model View Controller) framework. Interface is divided into three interconnected logics. Model is responsible for maintaining data and it is connected to database. Model responds to controller requests. View is everything that user see. It generates UI for the user. Views are created by data collected through model. Controller is considered as main component because it enables interconnection between views and models. Controllers are used to manipulate data. It accepts data from model and sends to view.

- Database

For the purposes of the project two databases were used: MySQL and Firebase. Firebase is used for interacting with sensors, and Laravel display data on user dashboard Firebase data. MySQL database is used for storing users registered, news posted, etc.

7. Development of a system

This section is dedicated to explaining modules used for building Water Quality Monitoring Project. The system encompasses measuring instruments, communication between microcontroller and database and, communication between database and Laravel application. Figure 6 is representation of how data is delivered to user.

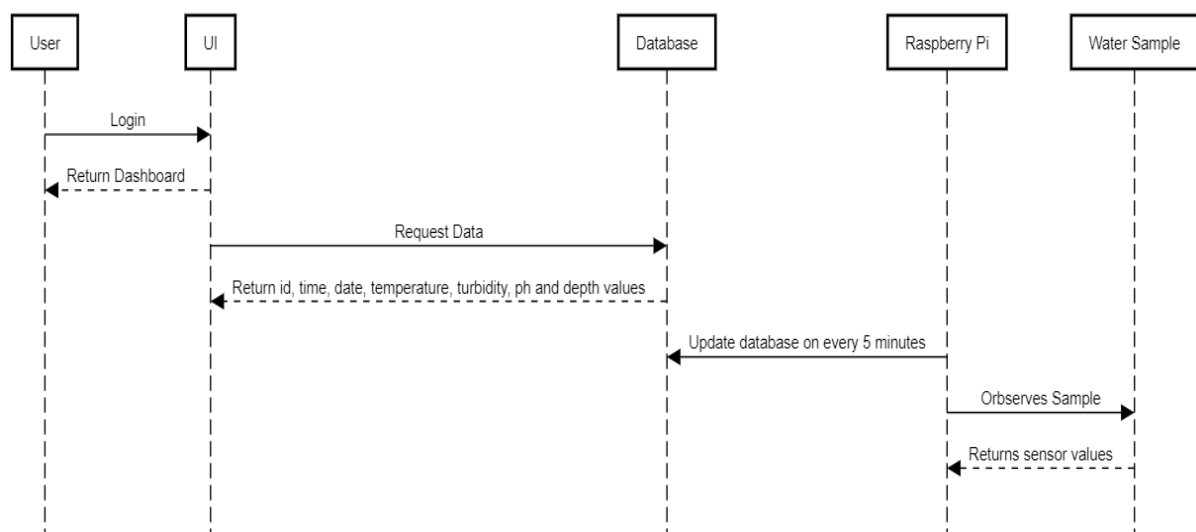


Figure 6. Sequence Flow Diagram

7.1. Laravel Application

The starting point of a system is an application which explains all useful information related to system. Any user, registered or not, could visit a site and can access all items from navigation menu.

7.1.1. Model View Controller

When using application user makes request, that is he perform an action. Action can be different, for example: access page, submit form, post news, update news, etc. Depending on action, route supports different method. Commonly used methods in Laravel are get, post, put, patch, delete. In order request to be successful the route for request must exist. This is done by registering routes. All requests are routed to appropriate controller. Routes accept URI with a closure which defines which page to be opened on provided URI. All routes are defined in routes/web.php.

Then, if route for request exists Laravel inspects controller associated to that route. Controllers contain functions that are used to manage data. Controllers group related request handling logic into a single class.

Controller interfaces with model requesting data. Model then interacts with database, retrieve data requested by controller and give that data to controller. Controller passes data to view, that is what user see as a result of his request.

The procedure of handling user requests is given in figure 7.

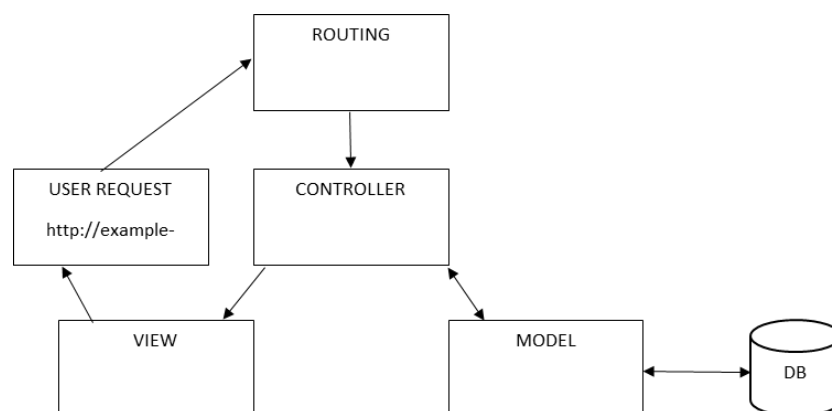


Figure 7. MVC Architecture

7.1.2. Middleware

Middleware is used for filtering user requests. If there is a middleware on route, Laravel inspects whether request satisfies logic defined in middleware or not. If it is true than user is redirected to desired page, if not he is redirected to some other page.

7.1.3. Migration

This feature allows easily creating tables and migrating them to database. Laravel provides schema builder for tables and there is no need of MySQL code to create a table. User defines fields of table and by using single command table is moved to database.

Defining the database to which tables are migrated is done in *.env* file. Following code is used for connecting to MySQL database:

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=water
DB_USERNAME=root
DB_PASSWORD=
```

Following code is used to connect to firebase:

```
FIREBASE_CREDENTIALS = "firebase-connection.json"
FIREBASE_DATABASE_URL = "https://waterparameters-default-rtdb.firebaseio.com/"
```

7.1.4. Register and Login

When registering, user needs to provide fields defined in *create_users_table*. Important field in this table is *is_admin* field, which determines whether user is admin or basic user. By default, this field is 0, and to make a user admin, this field need to be changed in database to 1. In *email* field there is constraint defined, which indicates that existing email cannot be used again; email cannot be same for more than one user. Following code defines users table.

```
public function up()
{
    Schema::create('users', function (Blueprint $table) {
        $table->id();
        $table->string('name');
        $table->string('email')->unique();
        $table->timestamp('email_verified_at')->nullable();
        $table->string('password');
        $table->boolean('is_admin')->nullable();
        $table->rememberToken();
        $table->timestamps();
    });
}
```

When user tries to login, he inserts his email and password. Depending on *is_admin* column he is redirected to admin or user dashboard. Redirecting is done by checking the routes. Routes are defined as follow:

```
Route::get('/dashboard', function () {
    return view('dashboard');
})->middleware(['auth','is_user'])->name('dashboard');
Route::get('/admin/home', function () {
    return view('adminHome');
})->middleware(['auth','is_admin'])->name('admin-dashboard');
```

To return admin or user dashboard, middleware is inspected. The purpose of *is_user* middleware is to check whether *is_admin* column is 0. If it is, then user dashboard is returned. If user tries to access admin dashboard, it will result in *access denied* page. Same logic is applied for admin logging. Another middleware applied for both user is *auth*, which

checks whether user exists in database. The order of middleware is important; first checks whether user exists, then user role. Middewares are used to increase security of an application.

7.1.5. Visitors

Visitor is any person who is not registered on application. Application allows everyone to access pages, to read its capabilities and to decide whether he will use the application or not. All pages must be registered in routes.

```
Route::view('/', 'welcome');
Route::view('/aboutUs', 'aboutUs');
Route::view('/news', 'news');
Route::view('/shop', 'shop');
Route::view('/q&a', 'q&a');
Route::view('/contact', 'contact');
Route::view('/guide', 'guide');
Route::view('/accessDenied', 'accessDenied');
Route::view('/aboutUs-sensors', 'aboutUs-sensors');
Route::view('/aboutUs/aboutUs-application', 'aboutUs-application');
```

7.1.6. Admin

The main task of admin is to provide support to users. Admin possibilities are to write posts, to update them and to receives emails from contact form available. Admin requests are defined with following routes:

```
Route::controller(PostController::class)->group(function(){
    Route::get('admin/posts/index', 'index');
    Route::get('admin/posts/create', 'create');
    Route::get('admin/posts/{post}/edit', 'edit');
    Route::get('admin/posts/{post}', 'show');
    Route::post('post', 'store');
    Route::delete('admin/posts/{post}', 'destroy');
    Route::put(' ', 'update');
});
Route::get('admin/users/index', [UserController::class, 'index']);
```

PostController defines functions which enable admin to manage posts. *UserController* is used to allow admin to see all registered users.

7.1.7. User

Term User refers to all people who use application for following water parameters. User has access to his dashboard, which provides visual representation of measured data. Data is retrieved from database and organised in table. To get dashboard following route is used:

```
Route::controller(ParameterController::class)->group(function(){
    Route::get('/parameters','table')->middleware(['auth','is_user']);
});
```

Table is manipulated by *ParameterController*.

```
class ParameterController extends Controller
{
    public function __construct(Database $database)
    {
        $this->database = $database;
        $this->tablename = 'parameters';
    }
    public function table()
    {
        $parameters= array_reverse($this->database
                                   ->getReference($this->tablename)
                                   ->orderByChild('date','time')->getValue());
        $pHs = array_reverse($this->database
                             ->getReference($this->tablename)
                             ->orderByChild('date','time')->getValue());
        $currentPh = current(array_column($pHs, 'ph'));
        $temperatures = array_reverse($this->database
                                      ->getReference($this->tablename)
                                      ->orderByChild('date', 'time')->getValue());
        $currentTemp = current(array_column($temperatures, 'temperature'));
        $turbidities = array_reverse($this->database
                                     ->getReference($this->tablename)
                                     ->orderByChild('date','time')->getValue());
        $currentTurb = current(array_column($turbidities, 'turbidity'));
        $levels = array_reverse($this->database
                               ->getReference($this->tablename)
                               ->orderByChild('date','time')->getValue());
        $currentLevel = current(array_column($levels, 'level'));
        return view('/parameters',['parameters' => $parameters,
                                   'currentPh' => $currentPh,
                                   'currentTemp'=>$currentTemp,
                                   'currentLevel' => $currentLevel,
                                   'currentTurb' => $currentTurb]);
    }
}
```

Controller has a function *table*, which creates variable *\$parameters* and here stores data of *parameters* table from database and order that data first by date, then by time. Ordering is used to show data in convenient way, starting with last measured data. This function also returns current pH, temperature, turbidity and depth values. Controller retrieves *dashboard* view.

Application provides graphs for better visual representation of measured parameters. This is achieved by using separate controllers for all parameters. In total four controllers are used:

PhController, *TemperatureController*, *TurbidityController* and *LevelController*. The same logic is applied for all of them, the difference is only in parameter observed. Following code refers to pH parameter:

```
public function phTable()
{
    $phs = array_reverse($this->database
                        ->getReference($this->tablename)
                        ->orderByChild('date','time')->getValue());
    $maxPh = max(array_column($phs, 'ph'));
    $minPh = min(array_column($phs, 'ph'));
    $differencePh = $maxPh - $minPh;
    $currentPh = current(array_column($phs, 'ph'));
    return view('/ph/phChart',
    ['phs' => $phs, 'maxPh' => $maxPh,
    'minPh' => $minPh, 'differencePh' => $differencePh,
    'currentPh' => $currentPh]);
    return view('/ph/parameters', ['currentPh' => $currentPh]);
}
public function phChartDay()
{
    $data = array_column($this->database
                        ->getReference($this->tablename)
                        ->orderByChild('date', 'time')->getValue(), 'ph');
    $labels = array_column($this->database
                        ->getReference($this->tablename)
                        ->orderByChild('date', 'time')->getValue(), 'date');
    return view('/ph/phChartDay', compact('labels', 'data'));
}
```

phTable function creates table with only pH values, retrieves maximum and minimum measured values as well as their difference. Retrieved values can be used in two views: phChart and parameters.

Graphs are created using JavaScript. Following code is example of ph graph:

```
<script type="text/javascript">
    var labels = {{Js:: from($labels)}};
    var phsToday = {{Js:: from($data)}};
    const data = {
    labels: labels,
    datasets: [{
    label: 'pH Today',
    backgroundColor: 'rgb(255, 69, 0)',
    borderColor: 'rgb(255, 69, 0)',
    data: phsToday,
    }]
    };
    const config = {
    type: 'line',
```

```

    data: data,
    options: {}
  };
  const myChart = new Chart(
    document.getElementById('phChartAll'),
    config
  );
</script>

```

For utilizing mentioned features following routes are used:

```

Route::controller(PhController::class)->group(function(){
    Route::get('/phChart',phTable');
    Route::get('/phChartDay', 'phChartDay');
});

```

Note: Controllers and routes for all parameters are same, only pH is renamed to Turbidity, Temperature or WaterLevel depending on parameter.

7.1.8. Classification

Based on currentTemp, currentTurb, currentPh and currentLevel values, application determines the status of water. They are retrieved by following code:

```

    $phs = array_reverse($this->database
        ->getReference($this->tablename)
        ->orderByChild('date','time')->getValue());
    $currentPh = current(array_column($phs, 'ph'));
    $temperatures = array_reverse($this->database
        ->getReference($this->tablename)
        ->orderByChild('date', 'time')->getValue());
    $currentTemp = current(array_column($temperatures, 'temperature'));
    $turbidities = array_reverse($this->database
        ->getReference($this->tablename)
        ->orderByChild('date','time')->getValue());
    $currentTurb = current(array_column($turbidities, 'turbidity'));
    $levels = array_reverse($this->database
        ->getReference($this->tablename)
        ->orderByChild('date','time')->getValue());
    $currentLevel = current(array_column($levels, 'level'));

```

Following figure show how is water classified based on measured parameter.

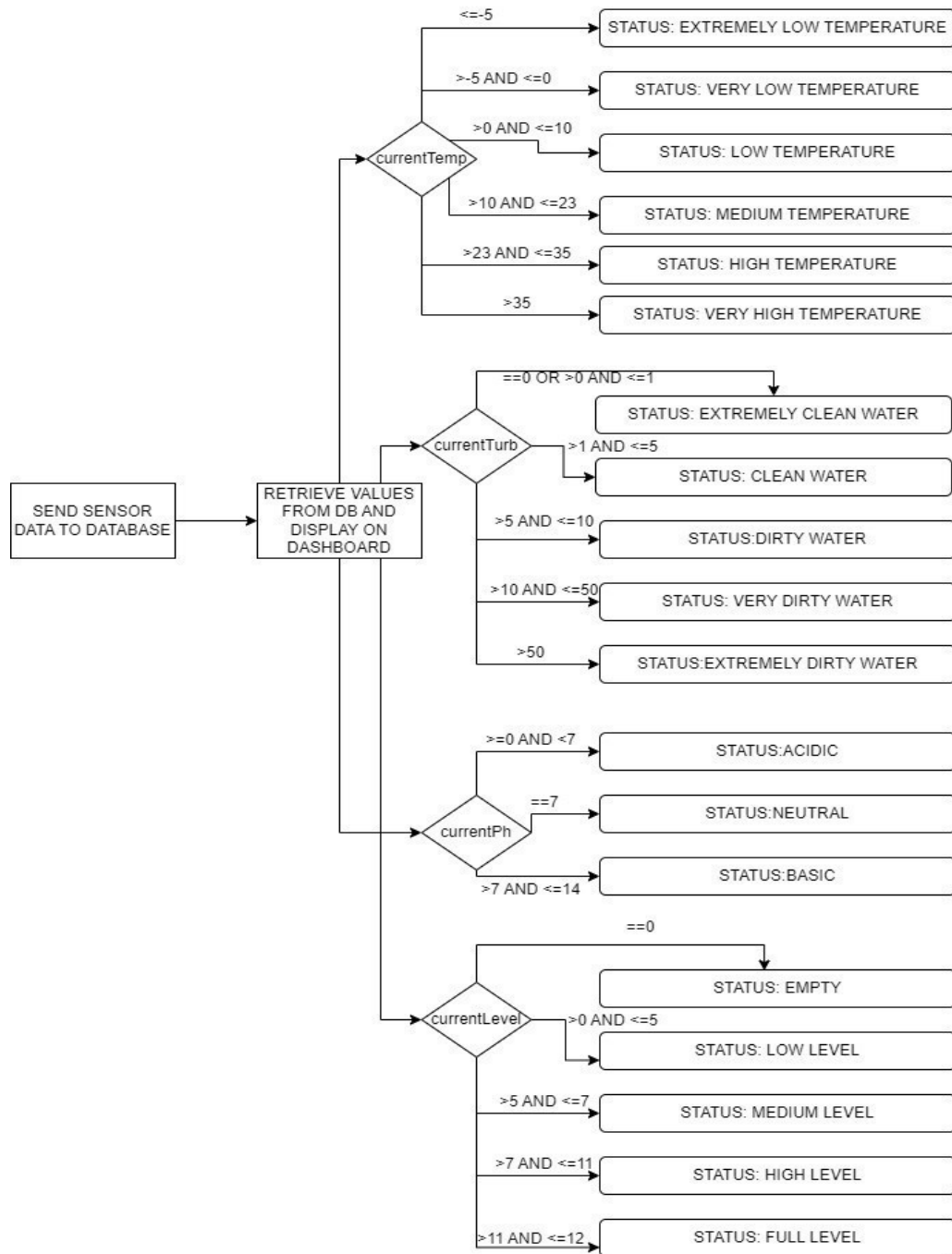


Figure 8. Classification of Water

7.1.9. Laravel Summary

Table 17 is summary of all views, models, controllers and tables used in application.

VIEWS	MODELS	CONTROLLERS	MIGRATIONS
auth(folder)	Contact	Auth	create_users_table
components(folder)	Parameter	ContactController	create_posts_table
layouts (folder)	Post	LevelController	create_contact_table
level (folder)	User	ParameterController	
ph (folder)		PhController	
post (folder)		PostController	
temperature (folder)		TemperatureController	
turbidity (folder)		TurbidityController	
user-post (folder)		UserController	
users (folder)		UserPostController	
all navbar pages			
page not found			
access denied			

Table 17. Laravel Application Summary

Views are styled with CSS and all style sheets are located in *public/css* folder. JavaScript files are in *public/js* folder and public folder also stores images that are part of the application.

7.2. Hardware Coding

The design of hardware part is detailly explained in section 5. After all components have been correctly connected, microcontroller and sensors are programmed using Python.

To read values from analog sensors (turbidity and ph) a code for ADC must be implemented. The code is placed in *DFRobot_ADS1115.py*. From this file, *read_voltage(channel_number)* function is used.

Following code is *SensorReadings.py* file and it is used to display values measured by sensors.

At the beginning several libraries are included:

- sys - allows interacting with the python run-time interpreter.
- time and datetime - supplies classes for manipulating dates and times.
- os - provides functions for creating and removing a directory (folder), fetching its contents, changing and identifying the current directory, etc. before interacting with operating system, os must be imported.
- glob – this module is imported to use glob() function for temperature sensor. Python method glob() returns a list of files or folder that matches path specified in pathname argument.
- RPi.GPIO as GPIO - supports referring to GPIO pins
- ADS1115 – this is function which is defined in DFRobot_ADS1115
- firebase – for communication with database

```
import sys
import time,datetime
import os
import glob
import RPi.GPIO as GPIO
from firebase import firebase
from DFRobot_ADS1115 import ADS1115
```

FOLLOWING CODE IS USED TO READ PH SENSOR:

```
def get_ph_value():
    adc1 = ads1115.read_voltage(1)    //ph sensor is analog, it is attached on channel 1 on ADC
    raw_ph_value = adc1['r']    //read_voltage function returns dictionary, ['r'] is used to take only
                                //the voltage value

    ph_value = raw_ph_value * (3.3/4095.0)
    ph_value = 4.5 * ph_value
    ph_value = round(ph_value,2) //round function is used to round value on two decimals
```

FOR TURBIDITY SENSOR:

```
def get_turbidity_value():
    adc2 = Ads1114.read_voltage(2) //turbidity sensor is analog, it is attached on channel 2 on ADC
    value = adc2['r']
    volt = value / 1000 //read_voltage gives voltage in milivolts, they are converted to volts
    if volt < 2.5: //limit of sensor is 2.5V, if voltage is below, it will give negative values
        turbidity_value = 3000 //for every voltage below 2.5 turbidity is 3000 NTU
    else:
        turbidity_value = -1120.4*volt*volt+5742.3*volt-4352.9 //otherwise this formula calculates
                                                                turbidity

    turbidity_value = round(turbidity_value,2)
```

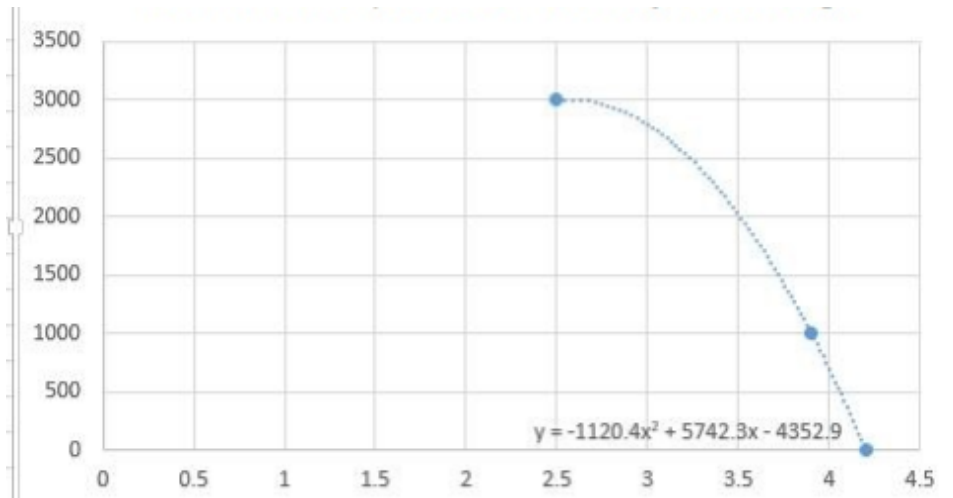


Figure 9. Relationship between voltage and turbidity

FOR TEMPERATURE SENSOR:

```
os.system('modprobe w1-gpio')
os.system('modprobe w1-therm')
base_dir = '/sys/bus/w1/devices/'

device_folder = glob.glob(base_dir + '28*')[0]
device_file = device_folder + '/w1_slave'

def read_temp_raw():
    f = open(device_file, 'r')
    lines = f.readlines()
    f.close()
    return lines

def read_temp():
    lines = read_temp_raw()
    while lines[0].strip()[-3:] != 'YES':
        time.sleep(0.2)
        lines = read_temp_raw()
    equals_pos = lines[1].find('t=')
    if equals_pos != -1:
        temp_string = lines[1][equals_pos+2:]
        temp_c = float(temp_string) / 1000.0
        return temp_c
```

FOR WATER LEVEL SENSOR:

```
def get_level_value():
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    TRIG = 23
    ECHO = 24
    GPIO.setup(TRIG,GPIO.OUT)
    GPIO.setup(ECHO,GPIO.IN)
    GPIO.output(TRIG, False)
    GPIO.output(TRIG,True)
    time.sleep(0.00001)
    GPIO.output(TRIG,False)
```



```

while GPIO.input(ECHO)==0:
    pulse_start = time.time()
    while GPIO.input(ECHO)==1:
        pulse_end = time.time()
        pulse_duration = pulse_end - pulse_start
        distance = pulse_duration * 17150
        distance = round(distance,2)
        distance_final = X - distance
        distance_final = round(distance_final,2)

```

Note: X is the depth of tank in which water is.

TO INSERT DATA TO DATABASE FOLLOWING CODE IS USED:

```

url = 'https://waterparameters-default-rtdb.firebaseio.com'
firebase = firebase.FirebaseApplication(url)
def update_firebase():
    Date = datetime.now().date()
    Time = datetime.now().strftime("%H:%M:%S")
    Temp = get_temperature_value()
    Turb = get_turbidity_value()
    Ph = getT_ph_value()
    Depth = get_level_value()
    Data = {"date" : date, "time" : time, "temperature" : temp, "turbidity" : turb, "ph" : ph, "level" : level}
    Firebase.post('parameters',data)
While True:
    Update_firebase()
    Time.sleep(300)

```

7.3. Design summary

Following table is summary of tasks that had to be completed for proper functioning of system.

TASK	IMPLEMENTATION	TASK COMPLETION
Correct connection between sensors and microcontroller	All components were connected following their requirements (voltage, ADC, resistors, etc.)	✓
Correct sensor readings	Formulas used for reading values must be correct	✓
Send data to database	Using firebase library in Python code for providing connection to localhost	✓
Retrieve data from database and display in user dashboard	Using Laravel Controllers	✓
Login/Register	Using Laravel Breeze Package	✓
Correct Navigation	Route for every page was created	✓

Admin Capabilities	Using controllers and middleware	✓
User Capabilities	Using controllers and middleware	✓

Table 18. Task Completion

For the development of the system waterfall model was followed.

The process started by defining which hardware is required. Next, it was decided that for front-end HTML, CSS and JS and, for back-end PHP, MySQL and Firebase databases will be used.

Software requirements define the services that system provides. At this point, it was decided on users type and their capabilities.

In design phase it was defined which programming languages will be used, interconnection between modules, defining tables which will be used. It was decided that for coding hardware Python will be used, and for application Laravel Framework.

In Coding Part, the physical design specifications were turned into a working code.

Testing is checking for system correctness. At this point any failure or error was solved. Testing is done in several phases described in details in section 7.

Maintenance phase is responsible for solving any further problems with the system. For example, sensors used in system have limited lifetime and they have specific storage conditions. To ensure proper functioning of system and to extend lifetime of a system, it must be maintained whole time.

Figure 10 is visual representation of steps performed during development of system.

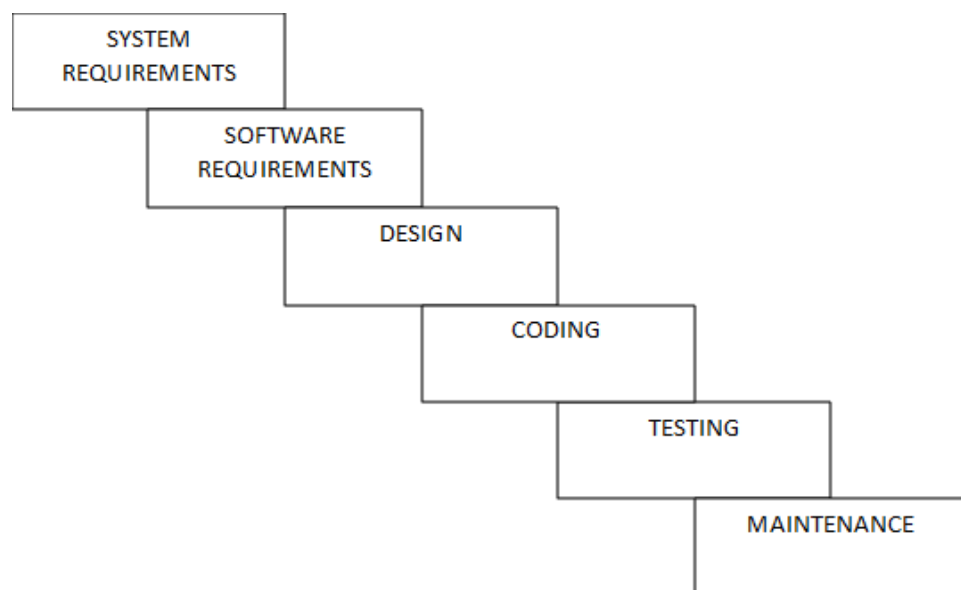


Figure 10. Waterfall Model

8. Visual Design

This chapter explains functionalities of an application. It describes the purpose of each part of the application. The chapter is mainly dedicated to application's design. Application supports two types of users:

- Admin
- Authenticated User

8.1. Admin

One of the important features of application is to provide constant support to its users. It is a responsibility of an admin. The job of admin is to ensure that whenever a change in system is made it is immediately reported to users. However, there is no limitation in publishing posts meaning that, admin can post any article that users may find usable and interesting. Also, admin has mailbox where he receives messages from users.

8.2. User

When user logs in he/she is redirected to parameters dashboard. This page provides menu where user can choose among parameters and, there are links to full guide on using a system and news related to it. Figure 11 shows how page looks like.

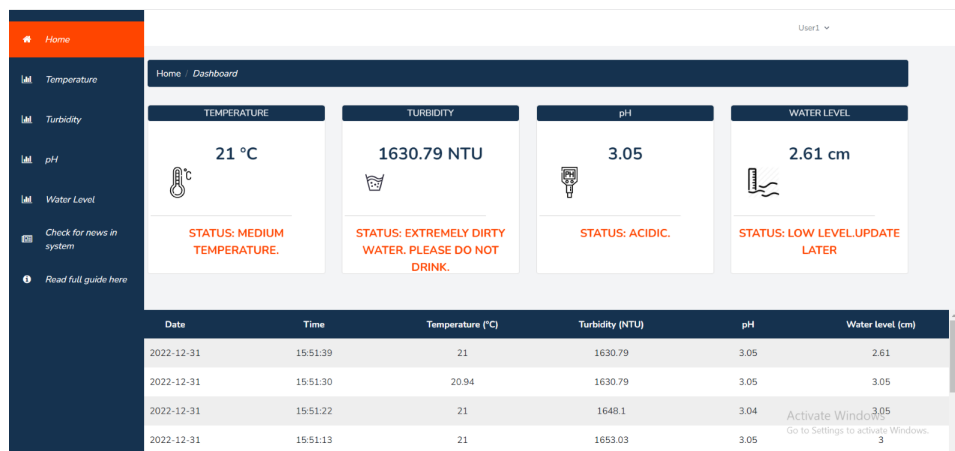


Figure 11. User Dashboard

Above screenshot is home page which contains cards showing the current values and notification based on parameter value and below there is table which represent measured data starting from latest added record. Data is organised in six columns: date, time, temperature, turbidity, pH and depth.

All menu items and cards are clickable and when they are clicked, they bring user to desired parameter. Figure 12 shows what appear after temperature card, or temperature sidebar item is clicked. Below there is message that shows the status of particular parameter, current temperature, minimum, maximum and their difference. Same page with corresponding parameter appears when clicking the rest of items.

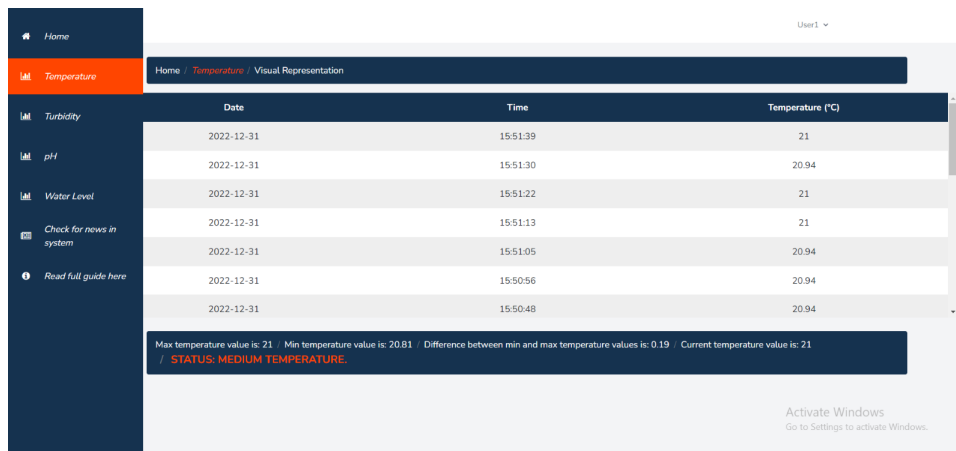


Figure 12. Temperature page

Additionally, user can observe data in visual representation. This feature is available for all parameters. Figure 13 shows how does it look like for temperature.

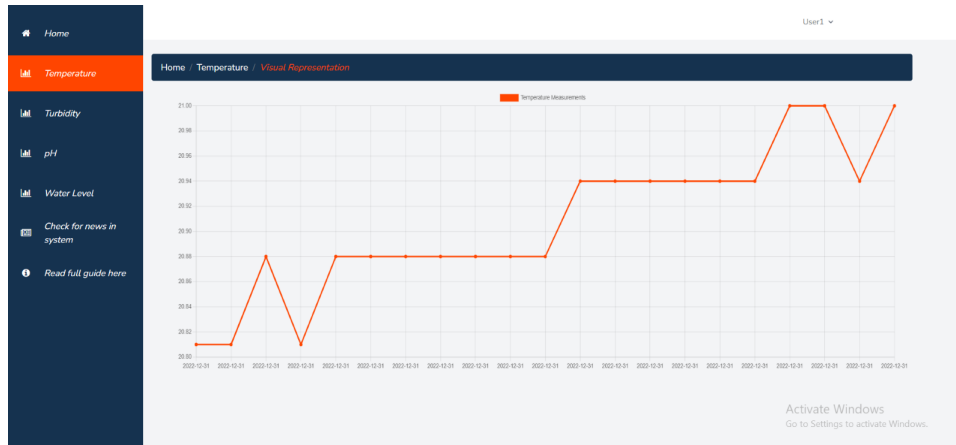


Figure 13. Temperature Graph

8.3. Visitors

Although application is indented for users who will register and use it, it can be accessed by anyone. Web application contains several parts and all of them have same structure shown in figure 14. Each page has navigation bar with clickable item, content that is different depending on page, loaded, and footer.

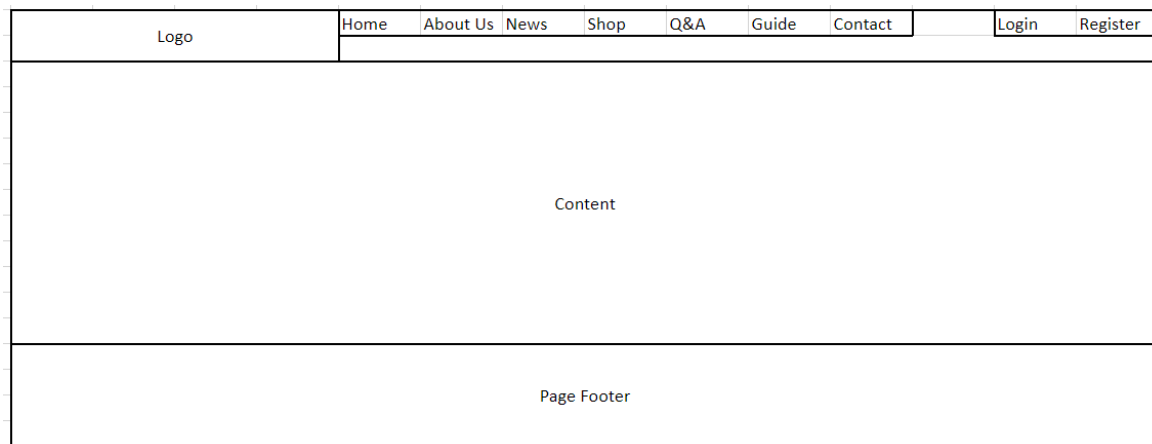


Figure 14. Structure of pages

Explanation of the content and purpose of navbar's links:

- Home – welcome page used to introduce application to users and to attract their attention.
- About Us – explain the purpose of application and how it is achieved; has links to two additional pages.
- News – this page displays the news that are published by admin. The purpose of this page is to introduce any change made in a system and to keep all news up to date.
- Shop – accessing this page user can see which hardware was used in a system and where from components are bought.
- Q&A – most frequent asked questions related to system.
- Guide – this page is user manual of system.
- Contact – allows users to keep in touch with providers in a way they can send emails to admin.
- Login/Register - logging existing users or registering new ones

8.4. Flowchart of application

Following figure shows the navigation of application.

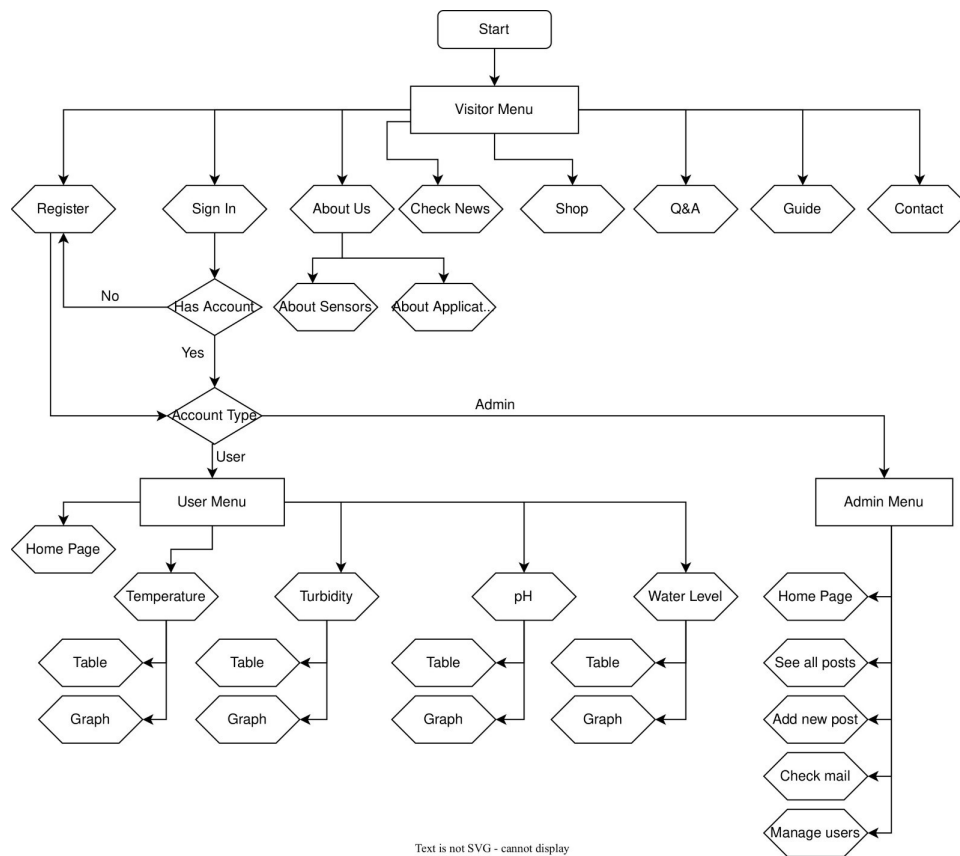


Figure 15. Flowchart

9. Testing and Results

The intention of this project was to develop portable, stand-alone system for the measurements of the pH, temperature, turbidity and depth of water.

Testing is very important phase of developing a system because it helps finding and fixing errors in code in early stages to save a lot of time and effort. During development of system many testings were made to adjust the formulas used to calculate final values as well to find the correct interval of measuring since all sensors have different response time.

To prove that every sensor gives correct results, as well that same correct results are obtained when sensors work as a whole several phases of testing were conducted. The procedure of testing is given in figure 16. For that purpose, different samples of water were used to prove that system works in different environments.

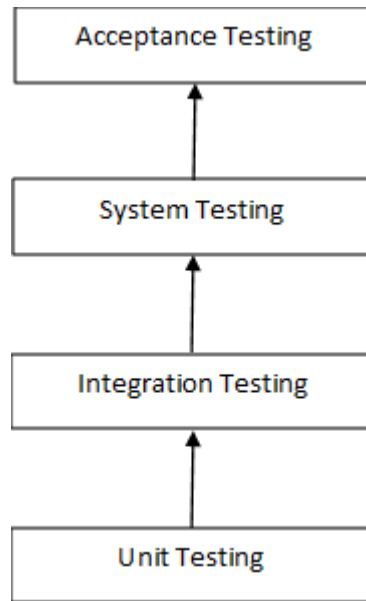


Figure 16. Testing Phases

9.1. Unit Testing

This is a very first step in proving functionality of sensors. At this stage every unit is tested one by one to check accuracy and reliability of results. This is used to validate that each unit of the software performs as designed.

9.1.1. pH Unit Testing

At this stage value of each sample was tested and depending on measured value sample is categorized as acidic, neutral or basic. For proving correctness of system measurements obtained values were compared with values that are proven for particular sample. As reference, table from source [21] was used. Figure 17 shows samples that were used. The results from measurements together with reference values are given in table 19.



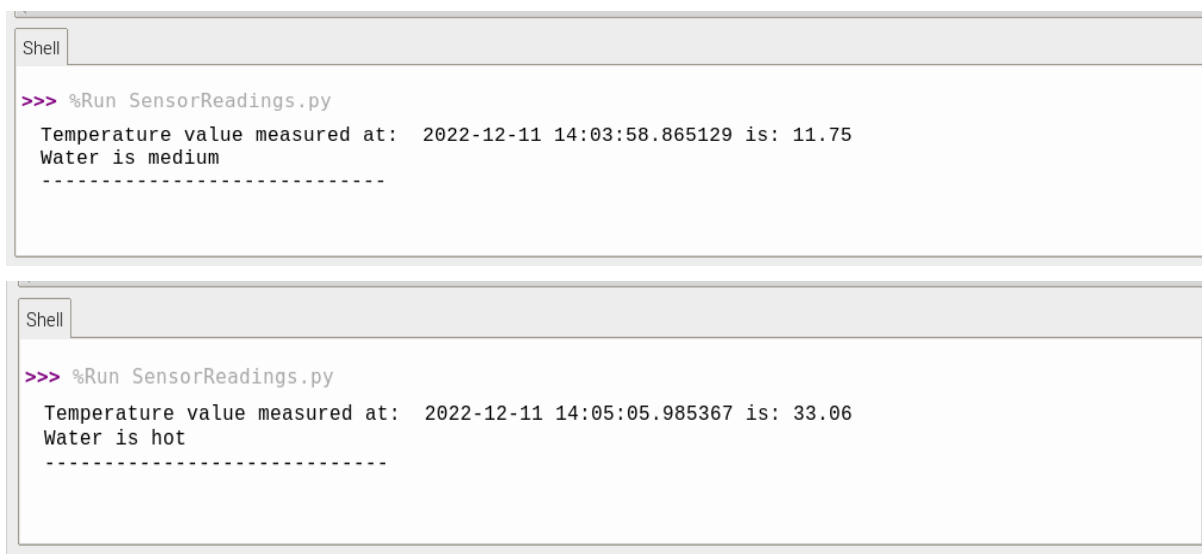
Figure 17. Liquid Samples

SUBSTANCE	VINEGAR	WINE	RAIN	MILK	PURE WATER	BAKING SODA
Expected Value	2.8	4	6.5	6.8	7	8.3
Expected Category	Acidic	Acidic	Acidic	Acidic	Neutral	Basic
System Value	2.76	4.09	6.38	6.69	7.24	8.19
System Category	Acidic	Acidic	Acidic	Acidic	Basic	Basic
Error Rate	1.42%	2.25%	1.88 %	1.61 %	2.25 %	1.32 %

Table 19. pH Comparision of expected and obtained values

9.1.2. Temperature Unit Testing

To check temperature readings two samples were used. Values obtained by sensors are given in figure 18.



```

Shell
>>> %Run SensorReadings.py
Temperature value measured at: 2022-12-11 14:03:58.865129 is: 11.75
Water is medium
-----

Shell
>>> %Run SensorReadings.py
Temperature value measured at: 2022-12-11 14:05:05.985367 is: 33.06
Water is hot
-----

```

Figure 18. Temperature Sensor Readings

9.1.3. Turbidity Unit Testing

Turbidity sensor was tested on following three samples.

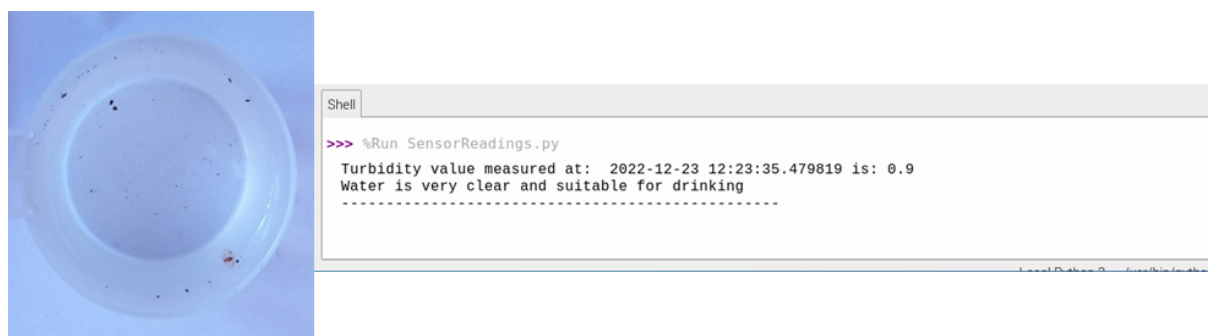


Figure 19. Turbidity Sample 1



Figure 20. Turbidity Sample 2

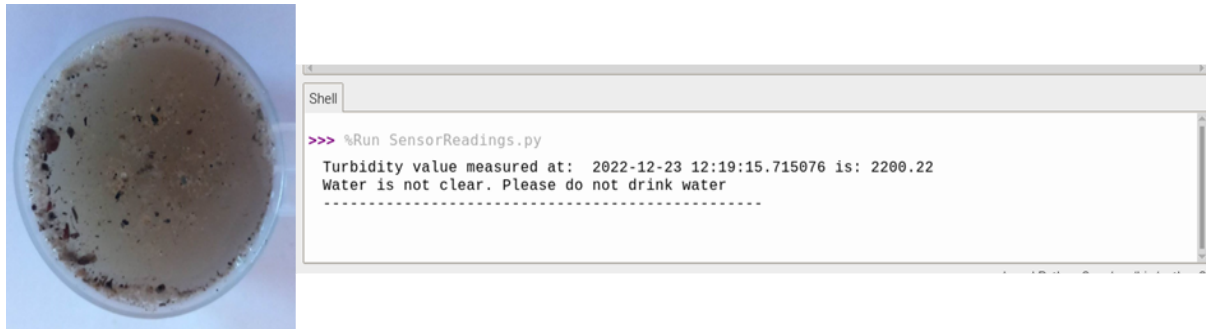


Figure 21. Turbidity Sample 3

Note: Turbidity sensor react on particles in water, not on water colour.

9.1.4. Water Level Unit Testing

The comparison of measurements with ruler and sensor are done on two different distances and the results are summarised in table 20.



Figure 22. Distance, Sample 1

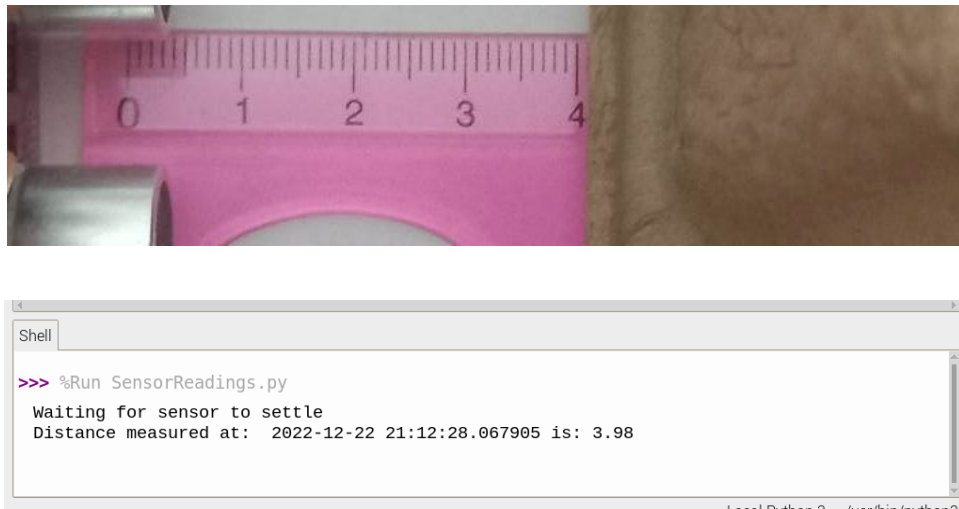


Figure 23. Distance, Sample 2

	FIGURE 1	FIGURE 2
Expected Value	10.00 cm	4.00 cm
System Value	10.27 cm	3.98 cm
Error Rate	2.7%	0.5%

Table 20. Distance Comparision of expected and obtained values

9.2. Integration Testing

This phase proves that all sensors work together and measure water parameters. The result of this step testing is given in figure below. Three samples were tested.

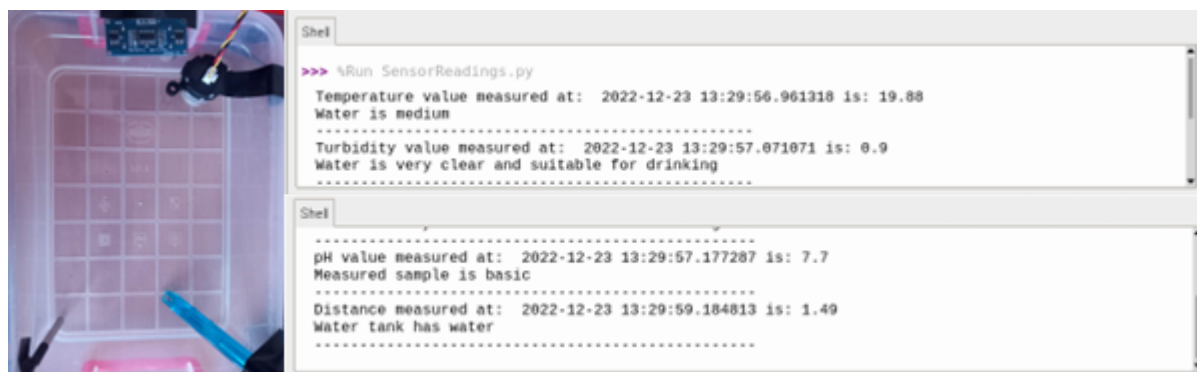


Figure 24. Integration Testing, Sample 1

Sample 1: clear water; pH is within range; turbidity is expected to be between 0 and 1

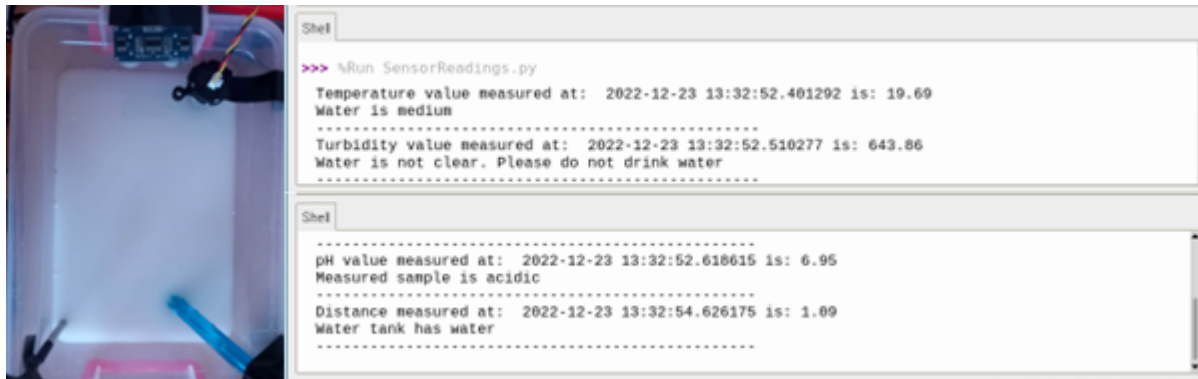


Figure 25. Integration Testing, Sample 2

Sample 2: mix of milk and water; pH of milk reduced the pH of water, previously water had pH of 7.7 when milk is added it is decreased to 6.95; milk results in greater turbidity.

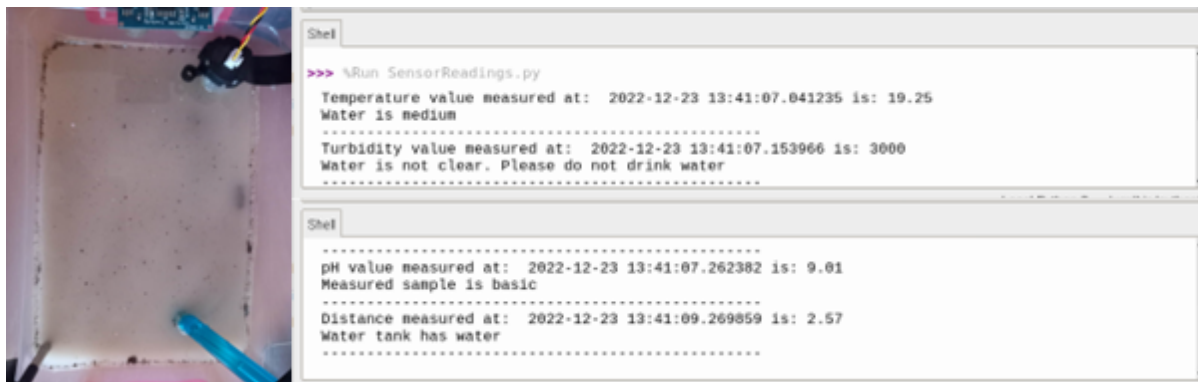


Figure 26. Integration Testing, Sample 3

Sample 3: mix of water, milk, sand and baking soda; pH of previous solution is affected by baking soda, from 6.95 to 9.01; sand drastically increases turbidity.

9.3. System Testing

This phase of testing aims to prove that sensors together with communications with database and with Laravel application work.

First, parameters were measured, than they are sent to database and displayed on user dashboard. Figure 27 shows parameters values.

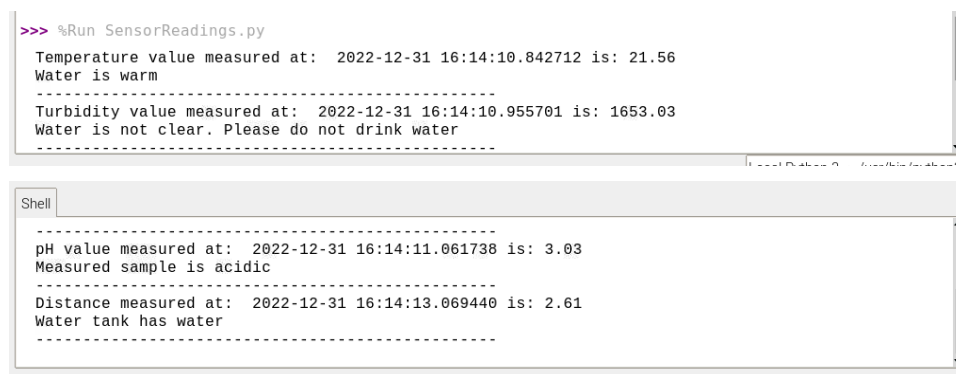


Figure 27. Parameter values obtained

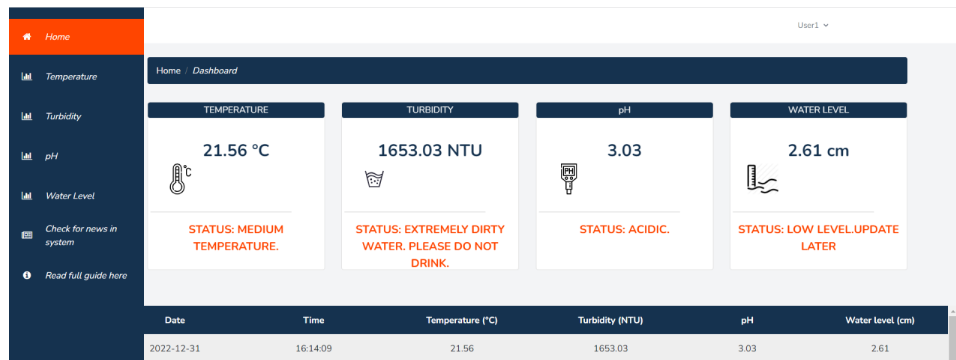


Figure 28. Parameter in dashboard

Figure 28 proves that inserted data into database is retrieved on dashboard. Table also starts with recent added record. Time, date and values are same in dashboard and in Shell.

10. Use Cases

10.1. Smart Lake

The term *Smart* proves the usage of IoT technology to follow the condition of lake, and to detect possible pollution in early stages. The impact of harmed water, as well as environmental impact on Ohrid Lake initiate a need of implementing proposed system on Lake; or any other water body.

By following water level city will be protected from floods and will overcome mentioned problem of mixing sewage with atmospheric water. Also, it is common case with Ohrid Lake when level recedes. In drastic conditions ecological disaster could happen as it is case with Prespa Lake, which has receded for even 50 metres. It entails the destruction of its entire ecosystem.

Temperature in great extent influences biological activities of organisms, influences other water parameters and determines the quality of life of its residents. Referring to touristic meaning of Ohrid Lake, it is useful to have insight into its temperature. Also, temperature change is symptom for other weather conditions as air temperature and rain.

When it comes to organisms living in water, it is crucial to enable them to live in desired conditions. Measuring turbidity and pH helps in maintaining clean environment.

10.2. Drinking Water Station

Potable water is crucial for public health. Referring to facts provided by WHO it is worrying how many people are influenced by unsafe water.

This system can be used on water stations to inspect water immediately before it is consumed. Early detection of inconsistencies helps in preventing people from diseases. The need for controlling water is especially expressed in rural areas where people are limited with water resources.

10.3. Fishpond

Water Quality Monitoring System can be used to help fishpond farmers to provide proper environment for fishes. The system continuously observes and monitor water conditions to ensure growth and survival of fishes.

Temperature influences reproductive activities of fishes. Also, it affects demand for oxygen. As temperature increases, it holds less oxygen. Additionally, fishes use more oxygen due to increased respiration rates.

Fishes have an acceptable pH range between 6.5 and 9.0. Every value out of the range affects fishes in different ways: limiting fish growth, limiting reproduction, damaging fish skin and even death [22].

Turbidity also has great impact on fishes live. High turbidity limits the process of photosynthesis and production of oxygen is very small. High concentration of particles harms respiratory system of fishes and could result with death.

When it comes to water level, it also affects the demand for oxygen. If level decreases concentration of fishes increases in reduced water volume.

Implementing system in fish farming helps farmers to control the environment according to fishes living conditions. If it is measured that a parameter is without range, preventive action can be taken on time to avoid losses and, at the same time to increase productivity.

10.4. Specie Detection

Knowing the fact that aquatic organisms have specific living conditions, system can be used to inspect water body to detect presence or absence of particular specie. All parameters highly affect existence of organisms. Water body is measured by system and values are compared with organism's convenient conditions. If the values are far from the tolerable values, it eliminates the possibility of organisms existence at inspected point.

Using system for this purpose can be used by scientists as starting point in research. It reduces human involvement, cost and time.

11. Future Work

The system successfully accomplishes the task for which it is intended. However, it can be improved and upgraded and to be used for many more purposes.

There is a number of additional parameters which can be measured by appropriate hardware. The system can be enriched with more sensors to give more precise insight into water quality.

The system is used in many applications and different type of purpose requires a different type of parameters that should be measured. The improvement can be done in a way that system gives freedom to the user to choose which parameters will he measure. Specific type of application may not require monitoring all parameters supported by system. Additional feature for user choosing capabilities could be defining of reference values. Current system compares measured values with constant values defined by developer, but in future user will decide which values will be reference values for his system. This feature could even expand the set of applications. The point of interest in this project is water, but system can be adapted to be applicable in testing different liquids. Having the possibility to choose reference values, system can be implemented in food factories, wineries, breweries, etc.

From technical point of view, system can be improved in a way that communications between modules are done wirelessly. Using different communication standards, Raspberry Pi can be configured to wirelessly communicate to the Internet and wirelessly interface to devices and sensors.

Another improvement could be done in power supply. For now, system is dependent on constant current flow, and it limits the place where system could be physically positioned. Powering RPi with batteries or power banks will improve system to great extent. Alternative power supply is also useful for ensuring stability in cases of current defects.

12. Conclusion

In recent years with the enlargement of industry, agriculture, exploration of natural resources and with the growth of the world population, water is susceptible to various damages in different parameters. To protect and to recover water it is crucial to implement new improved techniques which will overcome the issues with the old ones.

The approach followed for writing this paper was to introduce every step, from motivation to reading data on web page. It includes detail description of every part implemented, from both aspects, hardware and software.

Proposed system is IoT solution which uses Raspberry Pi as a microcontroller, four sensors and, additional hardware for creating successful interconnection between them. It is supported by web site with all necessary information. The aim of developing web site was to guide people with different technical backgrounds to effectively use the system.

The final product of project is smart and real-time water quality monitoring system, which can be easily installed near the target area. System can be used for different purposes and could help in preventing environment and living beings. Continuous following of parameters prevent from harmful consequences.

References:

- [1] Lakshmikantha, V., Hiriyannagowda, A., Manjunath, A., Patted, A., Basavaiah, J., & Anthony, A. A. (2021b). IoT based smart water quality monitoring system. *Global Transitions Proceedings*, 2(2), 181–186.
Available online [here](#).
- [2] *Drinking-water*. (2022b, March 21).
Available online [here](#).
- [3] UN General Assembly, *The human right to water and sanitation: resolution / adopted by the General Assembly*, 3 August 2010, A/RES/64/292
Available online [here](#).
- [4] Azzuni, A. (2015). Design, implementation, and evaluation of an online water quality monitoring system in Lake Saimaa, Finland. *Research Gate*.
Available online [here](#).
- [5] Wang, X., & Yang, W. (2019). Water quality monitoring and evaluation using remote sensing techniques in China: a systematic review. *Ecosystem Health and Sustainability*, 5(1), 47–56.
Available online [here](#).
- [6] Dunea, D. (2022). *Water Quality: Factors and Impacts*. Intechopen.
Available online [here](#).
- [7] O'Donnell, D. (2022, April 27). *Three Main Types of Water Quality Parameters Explained*. Sensorex.
Available online [here](#).
- [8] Geetha, S., & Gouthami, S. (2016b). Internet of things enabled real time water quality monitoring system. *Smart Water*, 2(1).
Available online [here](#).
- [9] Pasika, S., & Gandla, S. T. (2020b). Smart water quality monitoring system with cost-effective using IoT. *Heliyon*, 6(7), e04096.
Available online [here](#).
- [10] Omer, N. H. (2019). Water Quality Parameters. In (Ed.), *Water Quality - Science, Assessments and Policy*. IntechOpen.
Available online [here](#).
- [11] Halfacree, G. (2020). *The Official Raspberry Pi Beginner's Guide (The Official Raspberry Pi Beginner's Guide: How to use your new computer)* (4th ed.). Raspberry Pi Press.
- [12] Dennis, A.K. (2016) *Raspberry pi computer architecture essentials: Explore raspberry pi's architecture through innovative and Fun Projects*. Birmingham, UK: Packt Publishing.
Available online [here](#).
- [13] Gay, W. (2018). *Advanced Raspberry Pi: Raspbian Linux and GPIO Integration* (2nd ed.). Apress.
- [14] Charan Patel, B., G R and Goel, N. (2020). Introduction to sensors. *Research Gate*
Available online [here](#).

- [15] Fraden, J. (2015). *Handbook of Modern Sensors: Physics, Designs, and Applications* (5th ed. 2016). Springer.
Available online [here](#).
- [16] Perlstein, D. (2019, May 28). *Classifying Different IoT Sensors & Their Uses – Axonize Types of IoT Sensors*. Axonize.
Available online [here](#).
- [17] Simone, C., Ferrari, Picone, & Veltri, L. (2018). *Internet of Things: Architectures, Protocols and Standards* (1st ed.). Wiley.
Available online [here](#).
- [18] *Smart Cities | National Geographic Society*. (n.d.).
Available online [here](#).
- [19] Wild, J. E. M. (2021). *Electrical Engineering Step by Step: Basics, Components & Circuits explained for Beginners*. 3dtech.
- [20] Donat, W. (2018b). *Learn Raspberry Pi Programming with Python: Learn to Program on the World's Most Popular Tiny Computer* (2nd ed.). Apress.
Available online [here](#).
- [21] *Periodic Table of the Elements - pH*. (n.d.).
Available online [here](#).
- [22] Stevens, R. (n.d.). *Fish Pond Water Quality: As Simple as Chemistry 101*. Noble Research Institute.
Available online [here](#).

14. Appendices

14.1. Appendix 1: Connection between ADC and Raspberry Pi

The description of used ADC including description, specification, instruction, connection diagram and source code is available [here](#).

14.2. Appendix 2: pH Sensor

The description of pH sensor used and manual how to use it is available [here](#).

14.3. Appendix 3: Turbidity Sensor

This appendix includes description of sensor, specification and equation used for calculating turbidity value. It is available [here](#).

14.4. Appendix 4: Distance Sensor

[This tutorial](#) was followed for connecting sensor to RPi.

14.5. Code Repository

The code for the project can be found [here](#).