

**Interactive Aquarium**

Aman Khaware (1847209)

Vikash Singh (1847263)

Kumar Navin Barnwal (1847267)

Under the guidance of

Dr. Sandeep J

IOT project report submitted in partial fulfillment of the requirements of IV semester Master of Computer Applications, CHRIST (Deemed to be University)

March - 2020



CERTIFICATE

*This is to certify that the report titled* ***Interactive Aquarium*** *is a bona fide record of work done by* ***Aman Khaware*** ***(1847209), Vikash Singh (1847263) and Kumar Navin Barnwal (1847267)*** *of CHRIST (Deemed to be University), Bangalore, in partial fulfillment of the requirements of IV Semester Master of Computer Applications during the year 2020.*

**Head of the Department Project Guide**

Valued-by:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Name | : |
| 1. |  | Register Number | : |
|  |  | Examination Centre | : CHRIST (Deemed to be University) |
| 2. |  | Date of Exam | : |

# ACKNOWLEDGEMENTS

### First of all, we thank the lord almighty for his immense grace and blessings showered on me at every stage of this work. we are greatly indebted to our Head of the Department of Computer Science, Prof. Joy Paulose, CHRIST (Deemed to be University) for providing the opportunity to take up this project as part of my curriculum.

We are greatly indebted to our MCA Course Coordinator, Prof. Tulasi, for providing the opportunity to take up this project and for helping with her valuable suggestions.

we are deeply indebted to our project guide Dr. Sandeep J for his assistance and valuable suggestions as a guide. He made this project a reality.

we express our sincere thanks to Dr. Arul Kumar N, and Dr. Gobi R, lecturers of the Department of Computer Science, CHRIST (Deemed to be University), for their valuable suggestions during the course of this project. Their critical suggestions helped us to improve the project work.

Acknowledging the efforts of everyone, their chivalrous help in the course of the project preparation and their willingness to corroborate with the work, their magnanimity through lucid technical details lead to the successful completion of my project.

we would like to express our sincere thanks to all our friends, colleagues, parents and all those who have directly or indirectly assisted during this work.

**ABSTRACT**

The basic job of any aquarium owner is to feed the fishes, clean the water based on the quality of water, temperate and frequently process the required medication. Its maintenance is one of the crucial tasks. The purpose of this project is to build an Interactive Aquarium with the target customers as hotel, restaurant, office employee or any individual owning a Fresh Water Aquarium and wants to automate the task of maintenance. The system is supposed to monitor the physical changes in the water. The product will provide reading of temperature control, pH level etc. both in textual and audio form (for blind people).

The purpose of the project is to monitor the water quality of an aquarium based on PH

level, chlorine, alkaline, temperature, etc. Tracking the observatory change obtained

from the sensors and informing the user for any unusual activity in the form of text

through LCD screen and audio. The project is to be implemented with the purpose of reducing the maintenance cost of an aquarium given that the Aquarium interacts with the user. Even blind user can experience the status of an aquarium through audio.

**TABLE OF CONTENTS**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Sl. No.** | | | **Content** | | | **Page No.** | | |
| **1** | | | **Introduction** | | | **1** | | |
|  | | | 1.1 Overview of the system | | | 1 | | |
| **2** | | | **System analysis** | | | **2** | | |
|  | | | 2.1 Existing system | | | 2 | | |
|  | | | 2.1.1 Limitations of the existing system | | | 2 | | |
|  | | | 2.2 Proposed system | | | 2 | | |
|  | | | 2.2.1 Advantages of proposed system | | | 2 | | |
|  | | | 2.3 Literature survey | | | 3 | | |
|  | | | 2.4 Software tools used | | | 3 | | |
|  | | | 2.4.1 Front end | | | 3 | | |
|  | | | 2.4.1.1 Html | | | 3 | | |
|  | | | 2.4.1.2 Bootstrap | | | 4 | | |
|  | | | 2.4.1.3 JavaScript | | | 4 | | |
|  | | | 2.4.2 Backend | | | 5 | | |
|  | | | 2.4.2.1 MySQL | | | 5 | | |
|  | | | 2.5 Software used | | | 5 | | |
|  | | | 2.5.1 GitHub | | | 5 | | |
|  | | | 2.5.2 Sublime text | | | 5 | | |
| **3** | | | **System Requirements** | | | **7** | | |
|  | | | 3.1 System model | | | 7 | | |
|  | | | 3.2 User Requirements | | | 7 | | |
|  | | | 3.2.1 Functional requirements | | | 7 | | |
|  | | | 3.2.1.1 User | | | 7 | | |
|  | | | 3.2.1.2 Volunteer/NGO | | | 7 | | |
|  | | | 3.2.1.3 Admin | | | 8 | | |
|  | | | 3.2.1.4 Contact Us | | | 8 | | |
|  | 3.2.2 Non-Functional requirements | | | | | 9 |
|  | 3.3 System requirements | | | | | 9 |
|  | 3.3.1 Hardware requirements | | | | | 9 |
|  | 3.3.2 Software requirements | | | | | 9 |
|  | 3.3.2.1 Front end | | | | | 9 |
|  | 3.3.2.2 Back end | | | | | 10 |
|  | 3.3.2.3 Software used | | | | | 10 |
| **4** | **Design specification** | | | | | **11** |
|  | 4.1 Architectural design | | | | | 11 |
|  | 4.2 Data flow diagram | | | | | 13 |
|  | 4.2.1 Context diagram (Level 0 DFD) | | | | | 13 |
|  | 4.2.2 Level 1 DFD | | | | | 14 |
|  | 4.2.3 Level 2 DFD | | | | | 14 |
|  | 4.3 Entity relationship diagram | | | | | 15 |
|  | 4.4 Database design | | | | | 16 |
|  | 4.4.1 Admin | | | | | 16 |
|  | 4.4.2 User Info | | | | | 16 |
|  | 4.4.3 NGO Info | | | | | 16 |
|  | 4.4.4 Volunteer Info | | | | | 17 |
|  | 4.4.5 Complaint details | | | | | 17 |
|  | 4.4.6 Location Detail | | | | | 18 |
|  | 4.4.7 Contact Detail | | | | | 18 |
| **5** | **Implementation details** | | | **19** | | |
|  | 5.1 Source code | | | 19 | | |
|  | 5.1.1 Waste Management System | | | 19 | | |
|  | 5.1.1.1 signup1.php | | | 19 | | |
|  | 5.1.1.2 signup.php | | | 23 | | |
|  | 5.1.2 User | | | 24 | | |
|  | 5.1.2.1 beanemployee.php | | | 24 | | |
|  | 5.1.2.2 employee.php | | | 29 | | |
|  | 5.1.2.3 complaint.php | | | 29 | | |
|  | 5.1.2.4 complaintconfirm.php | | | 33 | | |
|  | 5.1.2.5 history.php | | | 34 | | |
|  | 5.1.3 NGO | | | 39 | | |
|  | 5.1.3.1 view\_complaint.php | | | 39 | | |
|  | 5.1.3.2 history.php | | | 42 | | |
|  | 5.1.4 Admin | | | 44 | | |
|  | 5.1.4.1 add\_location.php | | | 44 | | |
|  | 5.1.4.2 add\_location1.php | | | 48 | | |
|  | 5.1.4.3 hotspot\_area.php | | | 49 | | |
|  | 5.1.5 hotspot\_overall.php | | | 56 | | |
|  | 5.2 Screenshots | | | 64 | | |
|  | 5.2.1 Main Page | | | 64 | | |
|  | 5.2.2 Login Page | | | 64 | | |
|  | 5.2.3 User Home Page | | | 65 | | |
|  | 5.2.4 Complaint Page | | | 65 | | |
|  | 5.2.5 Complaint History Page | | | 66 | | |
|  | 5.2.6 Request Pending Page | | | 66 | | |
|  | 5.2.7 NGO/Volunteer Home Page | | 67 | | | |
|  | 5.2.8 Complaint Raised | | 67 | | | |
|  | 5.2.9 History Page | | 68 | | | |
|  | 5.2.10 Admin Panel Page | | 68 | | | |
| **6** | **Testing** | | **69** | | | |
|  | 6.1 Test plan | | 69 | | | |
|  | 6.2 Test cases | | 70 | | | |
|  | 6.2.1 Login form | | 70 | | | |
|  | 6.2.2 Registration form | | 71 | | | |
|  | 6.2.3 Complaint Raised Form | | 72 | | | |
|  | 6.2.4 User dashboard | | 73 | | | |
|  | 6.2.5 History | | 73 | | | |
|  | 6.2.6 Update Complaint | | 74 | | | |
|  | 6.2.7 Admin | | 74 | | | |
| **7** | **Conclusion** | | **75** | | | |
|  | 7.1 Future enhancement | | 75 | | | |
|  | References | |  | | | |

|  |  |  |
| --- | --- | --- |
| **Table No.** | **Table Name** | **Page No.** |
|  | **Design specification** | **11** |
| Table 4.1 | admin | 16 |
| Table 4.2 | user | 16 |
| Table 4.3 | NGO | 16 |
| Table 4.4 | volunteer | 17 |
| Table 4.5 | complaint | 17 |
| Table 4.6 | locality | 18 |
| Table 4.7 | contact | 18 |
|  | **Testing** | **69** |
| Table 6.1 | Login test cases | 70 |
| Table 6.2 | Registration form test cases | 71 |
| Table 6.3 | Complaint raised form test cases | 72 |
| Table 6.4 | User Dashboard test cases | 73 |
| Table 6.5 | History test cases | 73 |
| Table 6.6 | Update complaint test cases | 74 |
| Table 6.7 | Admin test cases | 74 |

|  |  |  |
| --- | --- | --- |
| **Fig. No.** | **Figure Name** | **Page No.** |
|  | **System Requirements** | **9** |
| Fig. 3.1 | System model | 9 |
|  | **Design specification** | **11** |
| Fig. 4.1 | Three tier architecture | 12 |
| Fig. 4.2 | Level 0 DFD | 13 |
| Fig. 4.3 | Level 1 DFD | 14 |
| Fig. 4.4 | Level 2 DFD | 14 |
| Fig. 4.5 | ER Diagram | 15 |
|  | **Implementation details** | **19** |
| Fig. 5.1 | Main Page | 64 |
| Fig. 5.2 | Login Page | 64 |
| Fig. 5.3 | User Home Page | 65 |
| Fig. 5.4 | Complaint Page | 65 |
| Fig. 5.5 | Completed History Page | 66 |
| Fig. 5.6 | Request Pending Page | 66 |
| Fig. 5.7 | NGO/Volunteer Home Page | 67 |
| Fig. 5.8 | Complaint Raised Page | 67 |
| Fig. 5.9 | History Page | 68 |
| Fig. 5.10 | Admin Panel Page | 68 |

1. **INTRODUCTION**

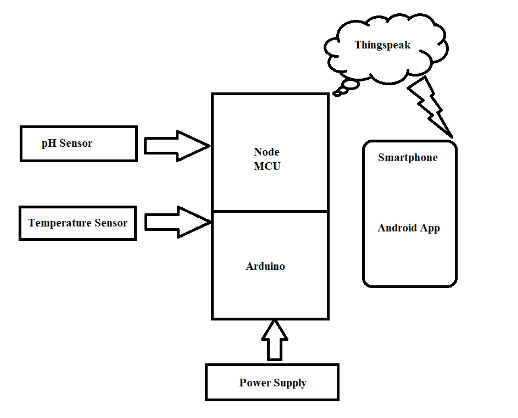
**1.1 OVERVIEW OF THE SYSTEM**

An aquarium is a tank wherein all water creatures such as fishes, rocks that fishes need for shelter, plants that they use as food and many more. It is used for different marine purposes and others use it as decor in their home. Different types of aquariums are used for decor such as saltwater aquarium, freshwater aquarium, etc. The maintenance of saltwater is very high and so as freshwater aquarium. This project is useful for the one who wants to have a freshwater aquarium in their home but cannot afford it due to high maintenance costs. Most of the customers are unaware of the steps to be taken for maintaining a healthy aquarium for the fishes.

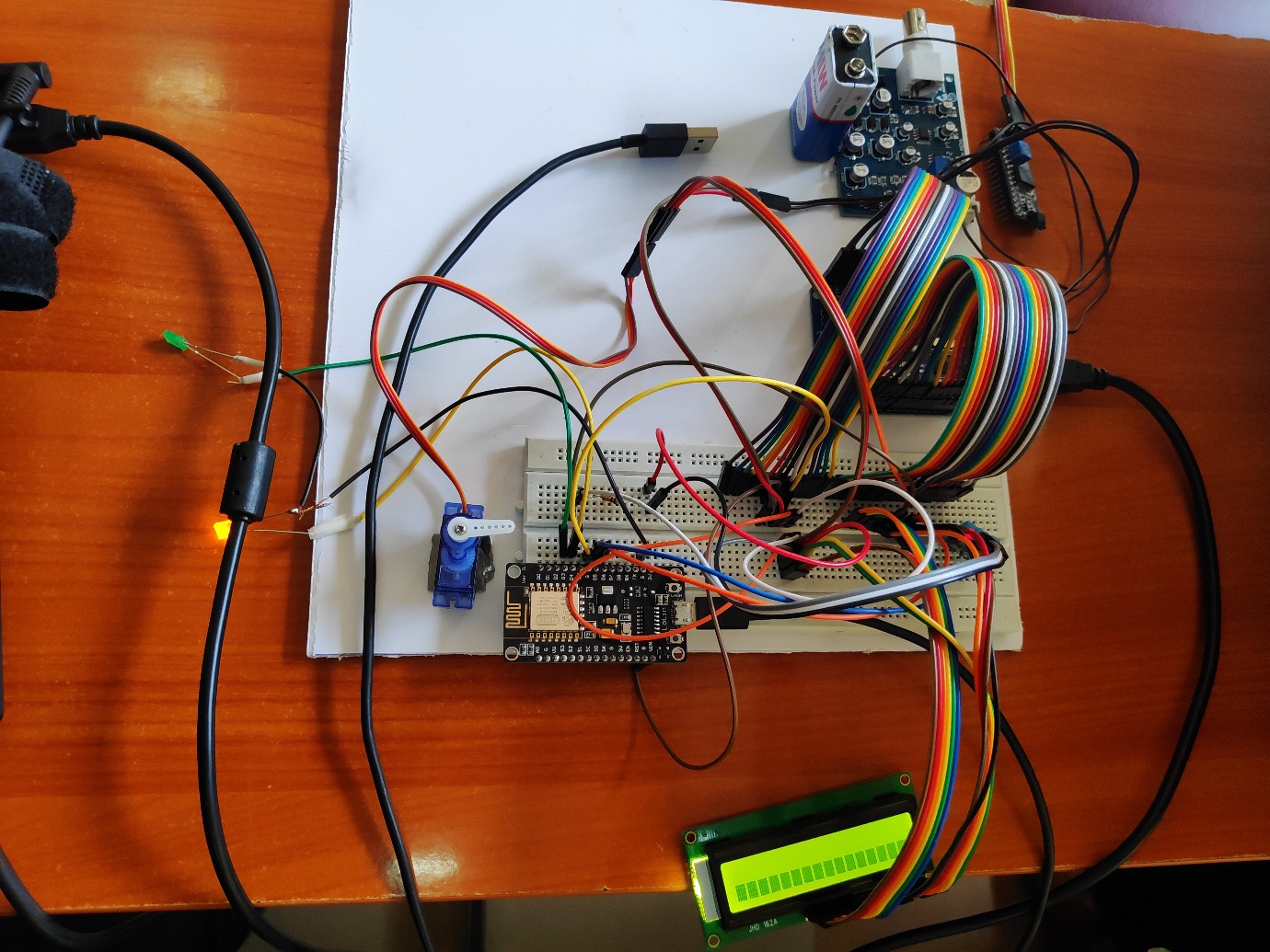
An interactive aquarium is designed for the customers to cut down the maintenance cost to half and maintain the aquarium themself. An aquarium gets contaminated due to multiple factors such as fishes releasing waste, water gets oily due to food provided by the customer to the fish, blooding due to fighting with other fishes disturbs the quality of water.

To check the quality of water required for fish is monitored using a ph sensor. When the quality of water is not healthy for the fishes’ user gets the notification. Another most important factor for the fishes is the humidity and temperature inside an aquarium. Whenever temperature raises or decreases, notification is sent to the user’s app. For feeding fish, a motor is installed with this project which will provide food to fish every 8 hours.

**1.2 BLOCK DIAGRAM**

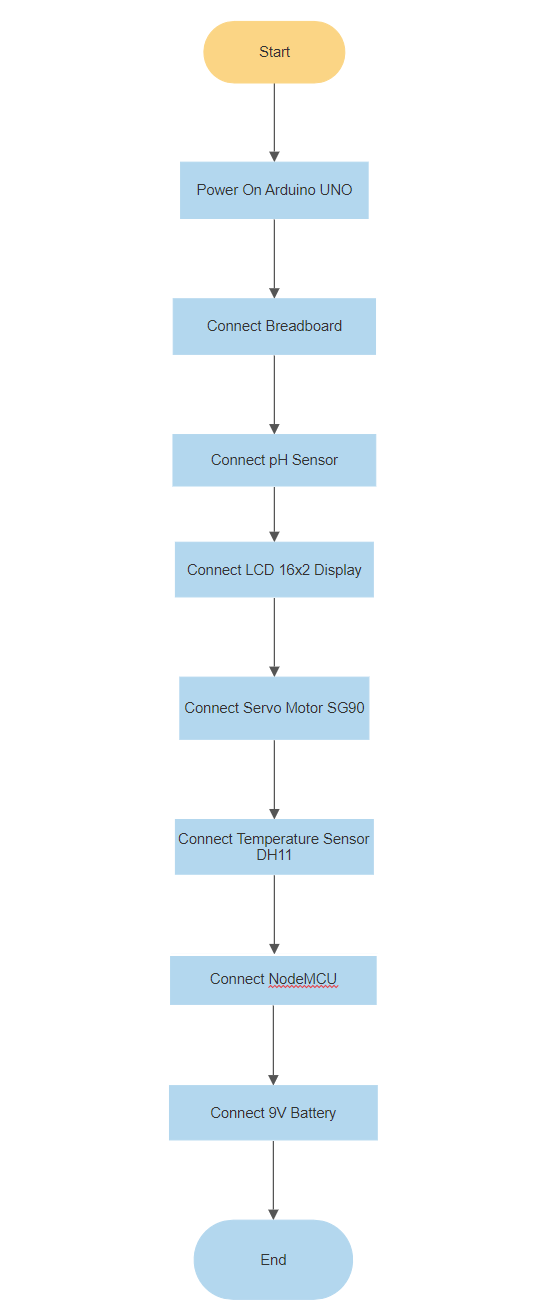


**Figure 1** Block Diagram



**Figure 2** Final Model

1. **SYSTEM REQUIREMENTS**
   1. **SYSTEM ARCHITECTURE**

.

* 1. **FUNCTIONAL REQUIREMENTS**

Initially the system starts and pH sensor electrode measures the pH value of the water of the aquarium. The water is being put in the container of the sensor. The pH value of the water is measured and the results are to be displayed on the display board. Through the results displayed, one can detect ph level of water. When the ph level is high monitor will display

message high and if the ph level is low then it will print low ph value. The dht11 sensor will monitor humidity and temperature inside an aquarium for the fishes. Feeder will be used for feeding fishes.

* 1. **REQUIREMENT SPECIFICATION**

Someof the requirements of the system are-

* + - Arduino Uno Board
    - ph Sensor Board
    - ph sensor electrode
    - Water level sensor (Probs)
    - Serial Monitor
    - Servo Motor
    - Node MCU
    - DHT11 sensor
  1. **SCHEDULE AND ESTIMATION**

The total cost of the project is Rs.2000/-. The project was started in the month of Novemberand planning to be done by end of February.

Schedule for the project: -

Week 1: Synopsis details.

Hardware details.

Circuit diagram.

Week 2: Detailed problem analysis and a literature survey.

Hardware details.

Hardware tools specification.

Week 3: The working of design.

Components requirement study.

Discuss the Connection.

Week 4: User interface code.

Week 5: Components for soldering.

Design platform.

Soldering.

Week 6: Serial input/output

Paperwork.

Report.Week 7: Testing of user interface code.

Design report.

Week 8: Validation and testing.

Draft report.

Week 9: Enhancement of design and serial interface.

Complete working project.

1. **SYSTEM DESIGN**
   1. **Modular Specification**

**Module 1 – Determining the pH value of the water**

This module determines the pH value of the water by using the pH sensor electrode.

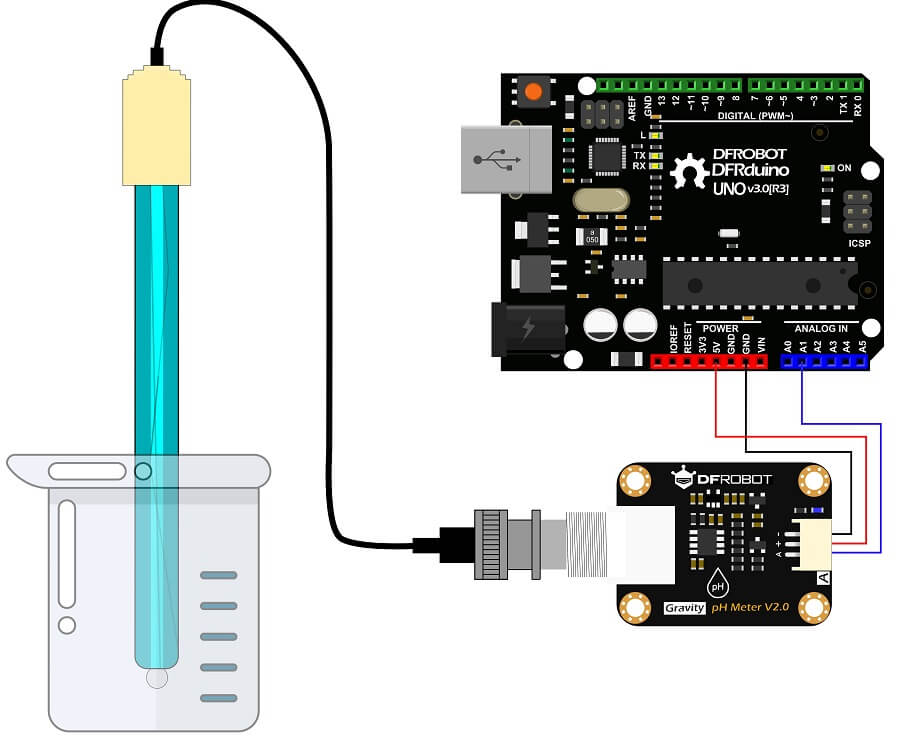
**Module 2 – Determining the humidity and temperature of an aquarium**

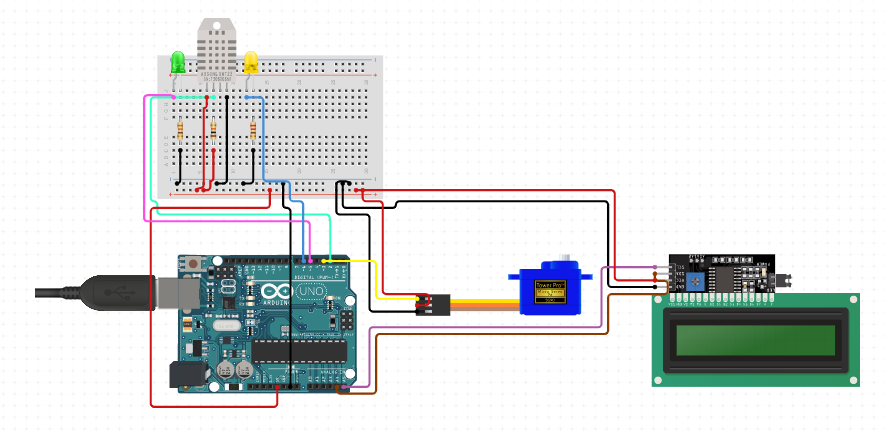
This module determinesthe humidity and temperature inside an aquarium for the fishes

**Module 3 – Feeder for feeding fish after every 8 hours**

This module provides food to the fishes after a set amount of time.

* 1. **Hardware circuit design**



****

* 1. **COMPONENT DESCRIPTION**

1. **Arduino Uno Board**-

The Arduino UNO is an open-source microcontroller board based on the [Microchip](https://en.wikipedia.org/wiki/Microchip_Technology) [ATmega328P](https://en.wikipedia.org/wiki/ATmega328P) microcontroller and developed by [Arduino.cc](https://en.wikipedia.org/wiki/Arduino). The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits.[[1]](https://en.wikipedia.org/wiki/Arduino_Uno#cite_note-Makerspace-1) The board has 14 Digital pins, 6 Analog pins, and programmable with the [Arduino IDE](https://en.wikipedia.org/wiki/Arduino#Software) (Integrated Development Environment) via a type B USB cable.[[4]](https://en.wikipedia.org/wiki/Arduino_Uno#cite_note-priceton-4) It can be powered by a USB cable or by an external 9 volt battery, though it accepts voltages between 7 and 20 volts.

General pin functions of Arduino Uno are-

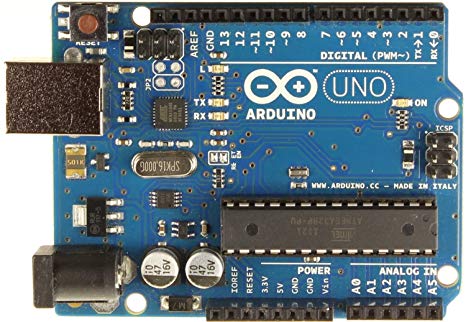
* **LED**: There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
* **VIN**: The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
  + - * **5V**: This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 20V), the USB connector (5V), or the VIN pin of the board (7-20V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board.
      * **3V3**: A 3.3-volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
      * **GND**: Ground pins.
      * **IOREF**: This pin on the Arduino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.
    - **Reset**: Typically used to add a reset button to shields which block the one on the board.

Special pin functions of Arduino Uno-

Each of the 14 digital pins and 6 Analog pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions2. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller. The Uno has 6 analog inputs, label led A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function.[[7]](https://en.wikipedia.org/wiki/Arduino_Uno#cite_note-website-7)

In addition, some pins have specialized functions:

* **Serial** / [**UART**](https://en.wikipedia.org/wiki/UART): pins 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
* **External Interrupts**: pins 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.
* **PWM** (**P**ulse **W**idth **M**odulation): 3, 5, 6, 9, 10, and 11 Can provide 8-bit PWM output with the analogWrite() function**.**
  + - **SPI** (**S**erial **P**eripheral **I**nterface): 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
    - **TWI** (**T**wo **W**ire **I**nterface) / I²C: A4 or SDA pin and A5 or SCL pin.
    - **AREF** (**A**nalog **REF**erence): Reference voltage for the analog inputs.



**Fig. Arduino Uno Board**

1. **ph Sensor Board-**

The analog pH sensor, specially designed for Arduino controllers is easy to use and can be used as a plug and play solution to measure pH value of a solution without any additional circuit required.

It has an LED which works as the Power Indicator, a BNC connector and PH2.0 sensor interface.  
To use it, we just connect the pH sensor with BNC connector, and plug the PH2.0 interface into the analog input port of any Arduino controller. With a simple program to ready analog voltage, and we will get the pH value easily.

**Working of pH Sensor**

The pH is a measure of the acidity or alkalinity of a solution; the pH scale varies from 0 to 14.

The pH indicates the concentration of hydrogen ions [H]+ present in certain solutions.

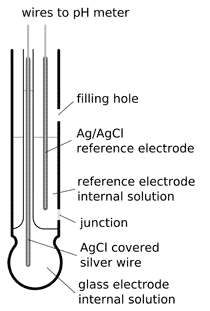
It can be quantified accurately using a sensor that measures the difference of potential between two electrodes: a reference electrode (silver/silver chloride) and a glass electrode is sensitive to hydrogen ion. This is what will form the probe. In addition, there are that use an electronic circuit to condition the signal appropriately, and that we can use this sensor with a microcontroller.



**Fig. ph Sensor Board**

1. **ph Sensor Electrode-**

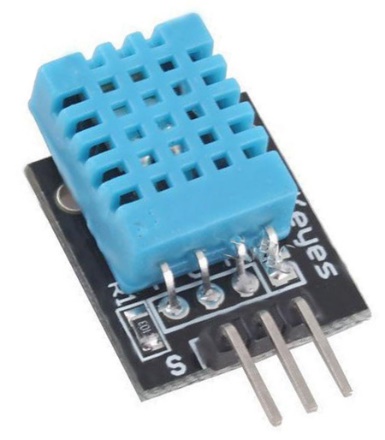
pH electrodes are constructed from a special composition glass which senses the hydrogen ion concentration.  This glass is typically composed of alkali metal ions that undergo an ion exchange reaction with the hydrogen ions in the test solution to generate a potential difference.  The bulb is filled with an acid solution (e.g. 0.1 molL-1 HCl).

****

1. **Humidity and Temperature sensor (DHT 11)-**

The DHT11 is a commonly used Temperature and humidity sensor. The sensor comes with a dedicated NTC to measure temperature and an 8-bit microcontroller to output the values of temperature and humidity as serial data. The sensor is also factory calibrated and hence easy to interface with other microcontrollers.

The sensor can measure temperature from 0°C to 50°C and humidity from 20% to 90% with an accuracy of ±1°C and ±1%.



1. **Servo Motor**

A servo motor is an electrical device which can push or rotate an object with great precision. If you want to rotate and object at some specific angles or distance, then you use servo motor. It is just made up of simple motor which run through servo mechanism. If motor is used is DC powered then it is called DC servo motor, and if it is AC powered motor then it is called AC servo motor. We can get a very high torque servo motor in a small and light weight packages.



1. **NodeMCU**

NodeMCU is a low-cost open source IoT platform. It initially included firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which was based on the ESP-12 module.



1. **Serial Monitor-**

The serial monitor is the 'tether' between the computer and your Arduino - it lets you send and receive text messages, handy for debugging and also controlling the Arduino from a keyboard! For example, you will be able to send commands from your computer to turn on LEDs.

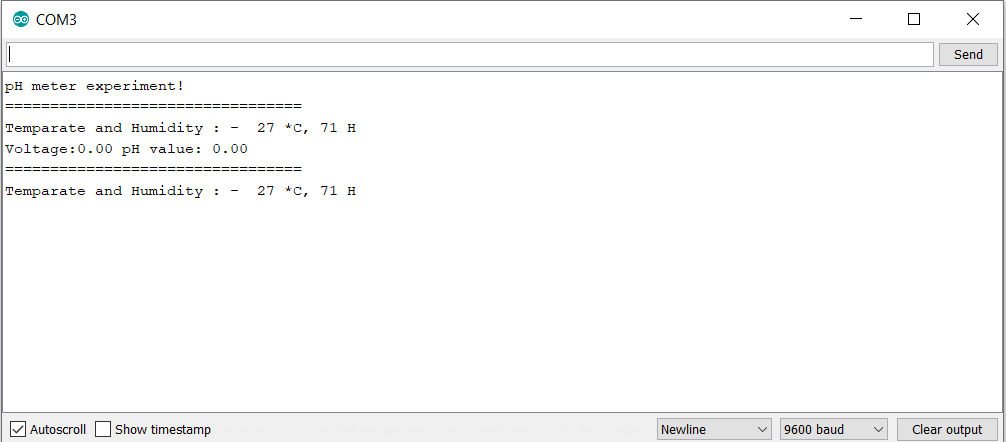


Fig. Serial Monitor

1. **Jumper Wires-**

A **jump wire** (also known as jumper wire, or jumper) is an electrical wire, or group of them in a cable, with a connector or pin at each end (or sometimes without them – simply "tinned"), which is normally used to interconnect the components of a breadboard or other prototype or test circuit, internally or with other equipment or components, without soldering. [1]

Individual jump wires are fitted by inserting their "end connectors" into the slots provided in a breadboard, the header connector of a circuit board, or a piece of test equipment.

There are different types of jumper wires. Some have the same type of electrical connector at both ends, while others have different connectors. Some common connectors are:

* Solid tips – are used to connect on/with a breadboard or female header connector. The arrangement of the elements and ease of insertion on a breadboard allows increasing the mounting density of

both components and jump wires without fear of short-circuits. The jump wires vary in size and colour to distinguish the different working signals.

* Crocodile clips – are used, among other applications, to temporarily bridge sensors, buttons and other elements of prototypes with components or equipment that have arbitrary connectors, wires, screw terminals, etc.
* Banana connectors – are commonly used on test equipment for DC and low-frequency AC signals.
* Registered jack (RJnn) – are commonly used in telephone (RJ11) and computer networking (RJ45).
* RCA connectors – are often used for audio, low-resolution composite video signals, or other low-frequency applications requiring a shielded cable.
* RF connectors – are used to carry radio frequency signals between circuits, test equipment, and antennas.



Fig. Jumper Wires

* 1. **TEST PLAN**

The objectives of testing include-

* To make sure that the pH sensor electrode is working fine and the pH values of the water are getting determined.
* To determine the correct pH values of the freshwater as the determination of water quality depends on the levels of the pH value of the water.
* To ensure that pH value of the water for freshwater lies between the range of 6.8 to 7.6 as the standard scale of pH is from 0-14.
* To determine DHT11 sensor is able to detect humidity and temperature inside an aquarium.
* LED light will determine whether nodeMCU is connected with server or not.
* To determine whether servo motor’s motor is working smooth

1. **IMPLEMENTATION**
   1. **TOOL/SOFTWARE DESCRIPTION**

Arduino is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board.

The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board -- you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program. Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package.

A program for Arduino hardware may be written in any programming language with compilers that produce binary machine code for the target processor. Atmel provides a development environment for their 8-bit AVR and 32-bit ARM Cortex-M based microcontrollers: AVR Studio (older) and Atmel Studio (newer).

**IDE**

The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It originated from the IDE for the languages Processingand Wiring. It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and syntax highlighting, and provides simple one-click mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for

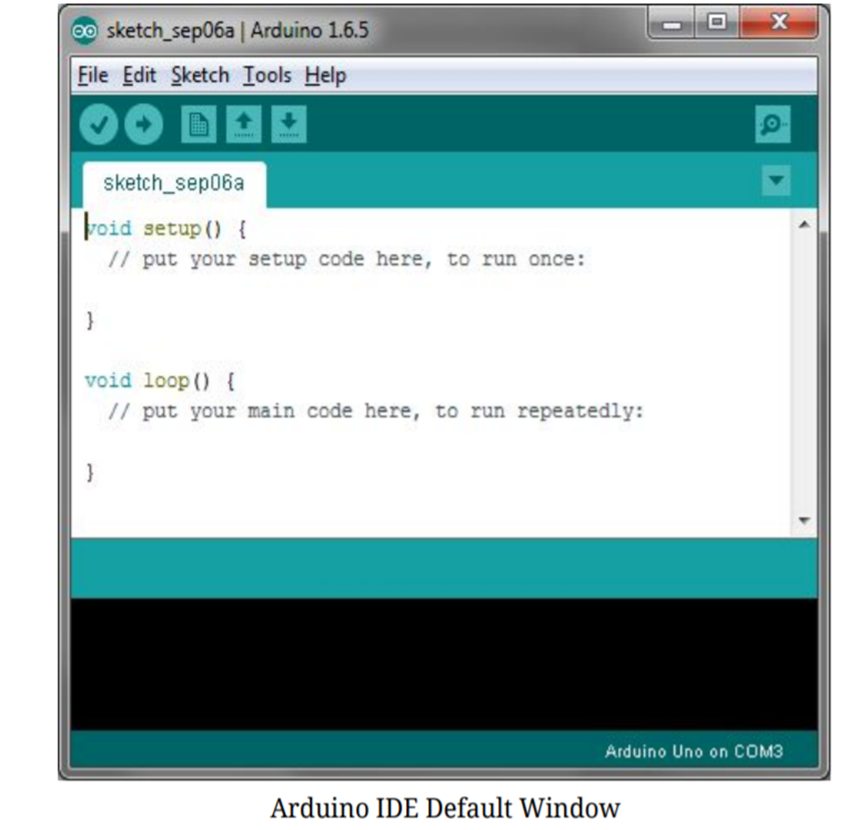
common functions and a hierarchy of operation menus. The source code for the IDE is released under the GNU General Public License, version 2

The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain, also included

with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

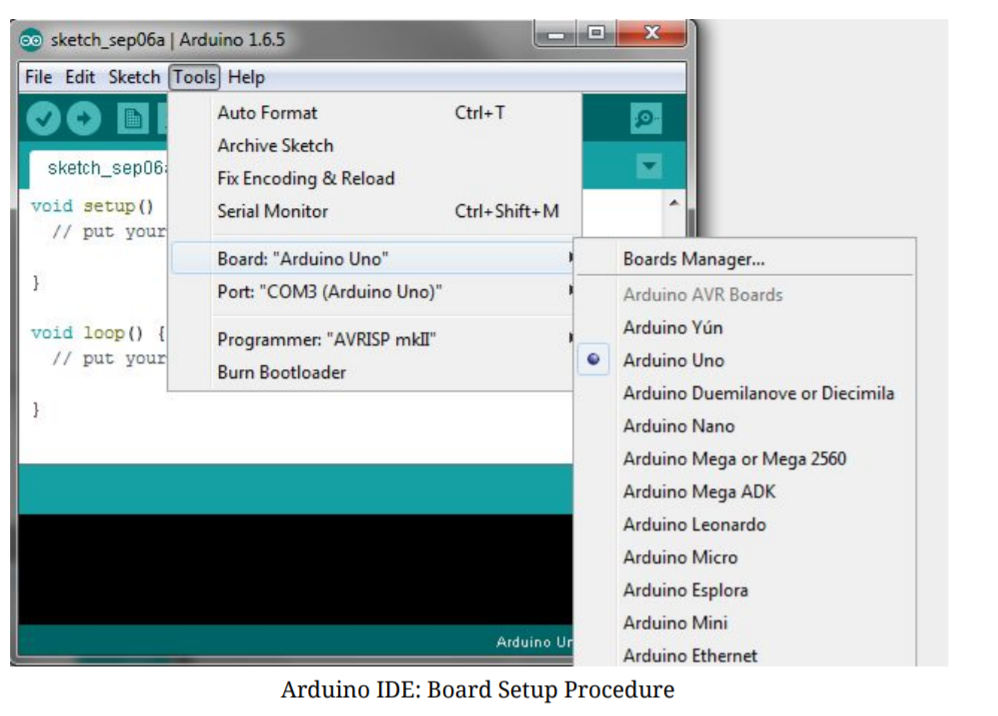
**Arduino IDE: Initial Setup**

Download Arduino Integrated Design Environment (IDE) here (Most recent version: 1.6.5): https://www.arduino.cc/en/Main/Software This is the Arduino IDE once it’s been opened. It opens into a blank sketch where you can start programming immediately. First, we should configure the board and port settings to allow us to upload code. Connect your Arduino board to the PC via the USB cable.

****

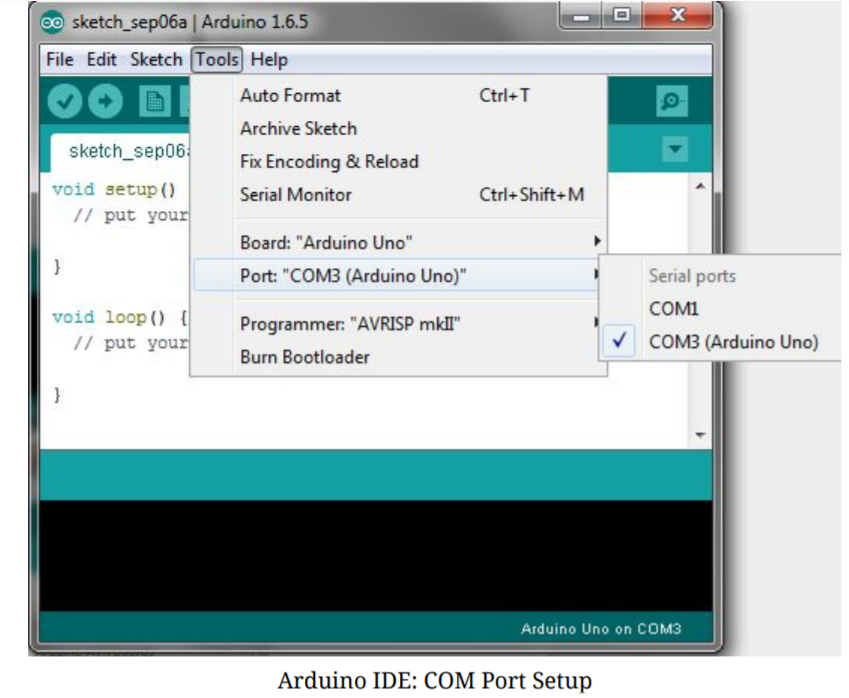
**IDE: Board Setup**

You have to tell the Arduino IDE what board you are uploading to. Select the Toolspulldown menu and go to Board.This list is populated by default with the currently available Arduino Boards that are developed by Arduino. If you are using an Uno or an Uno-Compatible Clone (ex. Funduino, SainSmart, IEIK, etc.), select Arduino Uno. If you are using another board/clone, select that board.



**IDE: COM Port Setup**

If you downloaded the Arduino IDE before plugging in your Arduino board, when you plugged in the board, the USB drivers should have installed automatically. The most recent Arduino IDE should recognize connected boards and label them with which COM port they are using. Select the Tools pulldown menu and then Port.Here it should list all open COM ports, and if there is a recognized Arduino Board, it will also give it’s name. Select the Arduino board that you have connected to the PC. If the setup was successful, in the bottom right of the Arduino IDE, you should see the board type and COM number of the board you plan to program. Note: the Arduino Uno occupies the next available COM port; it will not always be COM3.

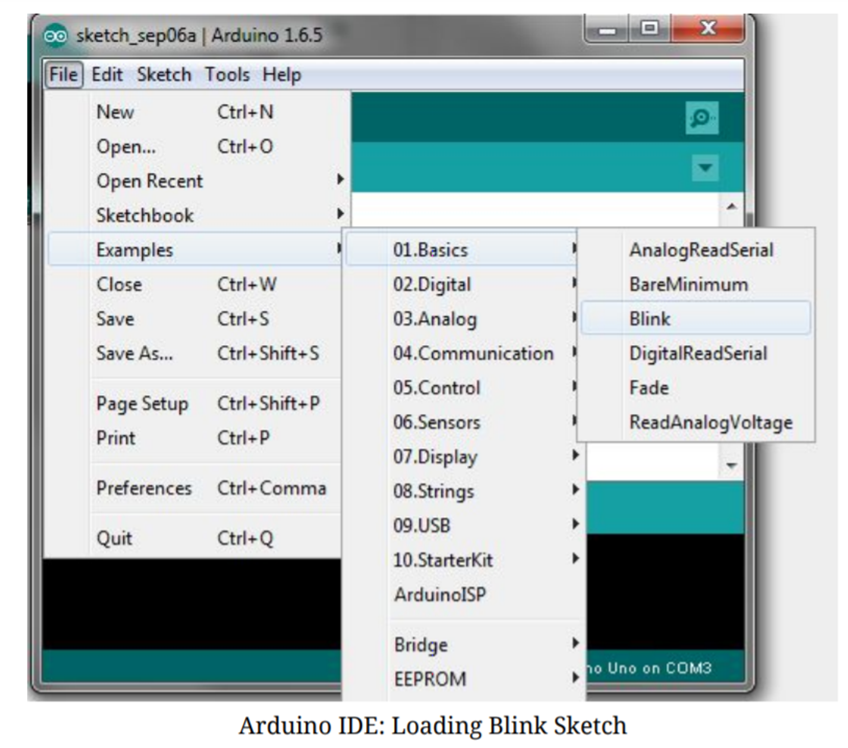
****

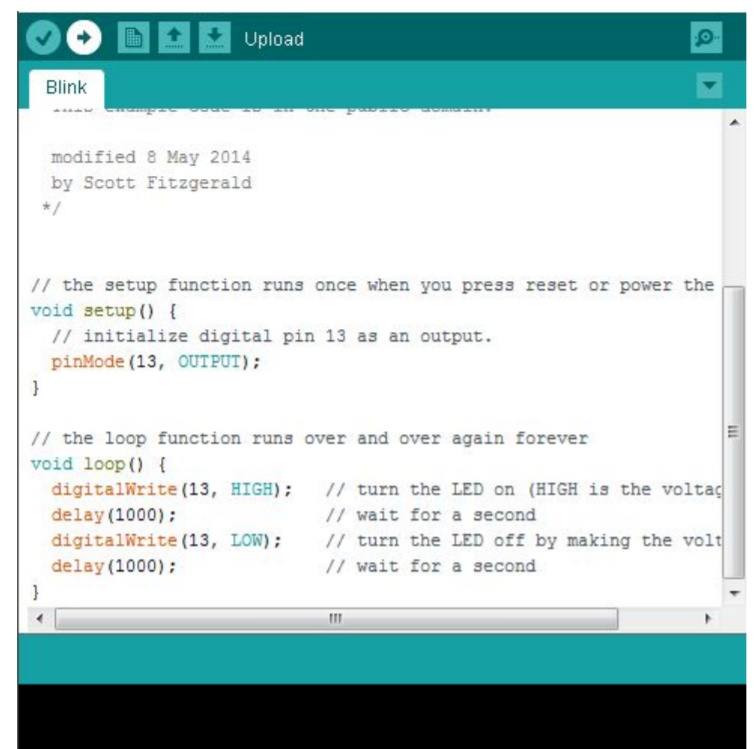
At this point, your board should be set up for programming, and you can begin writing and uploading code.

**Testing Your Settings: Uploading Blink**

One common procedure to test whether the board you are using is properly set up is to upload the “Blink” sketch. This sketch is included with all Arduino IDE releases and can be accessed by the Filepull-down menu and going to Examples, 01. Basics, and then select Blink. Standard Arduino Boards include a surface-mounted LED labeled “L” or “LED” next to the “RX” and “TX” LEDs, that is connected to digital pin 13. This sketch will blink the LED at a regular interval, and is an easy way to confirm if your board is set up properly and you were successful in uploading code. Open the “Blink” sketch and press the “Upload” button in the upper-left corner to upload “Blink” to the board.



****



**SKETCH**

A sketch is a program written with the Arduino IDE. Sketches are saved on the development computer as text files with the file extension .ino. Arduino Software (IDE) pre-1.0 saved sketches with the extension .pde.

A minimal Arduino C/C++ program consist of only two functions:

setup( ): This function is called once when a sketch starts after power-up or reset. It is used to initialize variables, input and output pin modes, and other libraries needed in the sketch.

loop( ): After setup() function exits (ends), the loop( ) function is executed repeatedly in the main program. It controls the board until the board is powered off or is reset.

Most Arduino boards contain a light-emitting diode (LED) and a current limiting resistor connected between pin 13 and ground, which is a convenient feature for many tests and program functions.[61] A typical program used by beginners, akin to Hello, World!, is "blink", which repeatedly blinks the on-board LED integrated into the

Arduino board. This program uses the functions pinMode( ), digitalWrite()and delay(), which are provided by the internal libraries included in the IDE environment. This program is usually loaded into a new Arduino board by the manufacturer.

#define LED\_PIN 13 *// Pin number attached to LED.*

**void** setup() {

pinMode(LED\_PIN, **OUTPUT**); *// Configure pin 13 to be a digital output.*

}

**void** loop() {

digitalWrite(LED\_PIN, **HIGH**); *// Turn on the LED.*

delay(1000); *// Wait 1 second (1000 milliseconds).*

digitalWrite(LED\_PIN, **LOW**); *// Turn off the LED.*

delay(1000); *// Wait 1 second.*

}

### **Libraries**

The open-source nature of the Arduino project has facilitated the publication of many free software libraries that other developers use to augment their projects.

**Libraries needed**

#include <LiquidCrystal.h> //lib for LCD

#include <Servo.h> //Library for Servo Motor

#include <ESP8266WiFi.h> //For Node Mcu ESP8266 Module

#include <SPI.h> // Working with LED lights in NodeMcu

* 1. **SOURCE CODE**
* **IOT Sample Source Code**

// -ve GnD

// A0 positive

#include <LiquidCrystal.h> //lib for LCD

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

int Contrast = 10;

//LCD Ard-UNO

//pin 1 to gnd

//pin 2 5v

//pin 3 6

//pin 4 12

//pin 5 gnd

//pin 6 11

//pin 11 5

//pin 12 4

//pin 13 3

//pin 14 2

//pin 15 5v

//pin 16 gnd

#include <Servo.h> //add '<' and '>' before and after servo.h

int servoPin = 8;

Servo servo;

int servoAngle = 0; // servo position in degrees

#include <SimpleDHT.h>

// for DHT11,

// VCC: 5V or 3V

// GND: GND

// DATA: 2

int pinDHT11 = 7;

SimpleDHT11 dht11(pinDHT11);

#define SensorPin A0 //pH meter Analog output to Arduino Analog Input 0

static unsigned long samplingTime = millis();

#define Offset 0.00 //deviation compensate

#define LED 13

#define samplingInterval 20

#define printInterval 800

#define ArrayLenth 40 //times of collection

int pHArray[ArrayLenth]; //Store the average value of the sensor feedback

int pHArrayIndex = 0;

void setup() {

pinMode(LED, OUTPUT);

Serial.begin(9600);

Serial.println("pH meter experiment!"); //Test the serial monitor

servo.attach(servoPin);

//Lcd Display code Starts Here

analogWrite(6, Contrast);

lcd.begin(16, 0);

lcd.setCursor(0, 1);

lcd.print(" Weight ");

lcd.print(" Measurement ");

delay(1000);

//lcd.clear();

//Lcd Display code ends Here

}

void loop() {

//Ph Sensor Code.....

static unsigned long printTime = millis();

static float pHValue, voltage;

if (millis() - samplingTime > samplingInterval) {

pHArray[pHArrayIndex++] = analogRead(SensorPin);

if (pHArrayIndex == ArrayLenth)pHArrayIndex = 0;

voltage = avergearray(pHArray, ArrayLenth) \* 5.0 / 1024;

pHValue = 3.5 \* voltage + Offset;

samplingTime = millis();

}

if (millis() - printTime > printInterval) {

Serial.print("Voltage:");

Serial.print(voltage, 2);

Serial.print(" pH value: ");

Serial.println(pHValue, 2);

digitalWrite(LED, digitalRead(LED) ^ 1);

printTime = millis();

}

//PH sensor code ends here

// TEMPERATUE sensor start here...

Serial.println("=================================");

// read without samples.

byte temperature = 0;

byte humidity = 0;

int err = SimpleDHTErrSuccess;

if ((err = dht11.read(&temperature, &humidity, NULL)) != SimpleDHTErrSuccess) {

Serial.print("Read DHT11 failed, err="); Serial.println(err);delay(1000);

return;

}

Serial.print("Temparate and Humidity : - ");

Serial.print((int)temperature); Serial.print(" \*C, ");

Serial.print((int)humidity); Serial.println(" H");

// DHT11 sampling rate is 1HZ.

delay(1500);

//TEMP sensor ends here

// Servo Motor SG90 COde starts here

//control the servo's direction and the position of the motor

servo.write(10); // Turn SG90 servo Left to 45 degrees

delay(2000); // Wait 1 second

servo.write(140); // Turn SG90 servo back to 90 degrees (center position)

delay(2000); // Wait 1 second

//end control the servo's direction and the position of the motor

//control the servo's speed

/\*\*\*

//if you change the delay value (from example change 50 to 10), the speed of the servo changes

for(servoAngle = 0; servoAngle < 180; servoAngle++) //move the micro servo from 0 degrees to 180 degrees

{

servo.write(servoAngle);

delay(50);

}

for(servoAngle = 180; servoAngle > 0; servoAngle--) //now move back the micro servo from 0 degrees to 180 degrees

{

servo.write(servoAngle);

delay(10);

}

//end control the servo's speed

\*\*\*/

// Servo Motor SG90 Code ends here

}

//Manual Function for PH Sensor starts here

double avergearray(int\* arr, int number) {

int i;

int max, min;

double avg;

long amount = 0;

if (number <= 0) {

Serial.println("Error number for the array to avraging!/n");

return 0;

}

if (number < 5) { //less than 5, calculated directly statistics

for (i = 0; i < number; i++) {

amount += arr[i];

}

avg = amount / number;

return avg;

}

else {

if (arr[0] < arr[1]) {

min = arr[0]; max = arr[1];

}

else {

min = arr[1]; max = arr[0];

}

for (i = 2; i < number; i++) {

if (arr[i] < min) {

amount += min; //arr<min

min = arr[i];

}

else {

if (arr[i] > max) {

amount += max; //arr>max

max = arr[i];

}

else {

amount += arr[i]; //min<=arr<=max

}

}

}

avg = (double)amount / (number - 2);

}

return avg;

}

//Manual Function for PH Sensor ends here

# TESTING

* 1. **TESTING ENVIRONMENT**

A testing environment is a setup of software and hardware for the testing teams to execute test cases. In other words, it supports test execution with hardware, software and network configured.

Test bed or test environment is configured as per the need of the Application Under Test. On a few occasion, test bed could be the combination of the test environment and the test data it operates.

Setting up a right test environment ensures software testing success. Any flaws in this process may lead to extra cost and time to the client.

Test Environment consists of elements that support test execution with software, hardware and network configured. Test environment configuration must mimic the production environment in order to uncover any environment/configuration related issues.

**Factors for designing Test Environment**

* Determine if test environment needs archiving in order to take back-ups.
* Verify the network configuration.
* Identify the required server operating system, databases and other components.
* Identify the number of license required by the test team.

Fish are "cold-blooded" and therefore assume the temperature of the water they live in. Water temperature is therefore the most important physical factor for fish survival and growth. Body temperature, and thus the water temperature, has an effect on level of activity, behavior, feeding, growth, and reproduction of the fish. Each species has its tolerance limits and optimum range. When water temperatures are outside the optimum range, fish body temperature will either be too high or too low and fish growth will be affected or the fish will even die.

Water quality gets affected due to many reasons such as fish producing waste, oily surface due to food given to the fishes etc. To identify this unhealthy environment with naked eye is very difficult.

Our project test different condition of water, humidity and temperature every 20 second. If any changes are found user gets a notification like change water, provide proper humidity and temperature for the fishes. It is very important to maintain a healthy environment for the fishes inside an aquarium. Hence the quality of water and humidity and temperature is being tested for following condition: -

* Ensure fishes are getting adequate humidity and temperature.
* Check water provided for fishes is safe or not.

We should take some factors in consideration-

* Check from where water is taken for filling aquarium for the fishes.
* Turn on/off the light whenever temperature is required.
* In case of high temperature or humidity decrease the intensity of light
* In case of low temperature or humidity increase the intensity of light
* To ensure the ammonia and nitrite levels have not reached harmful levels
  1. **TEST PLAN**

Test Plan is a dynamic document. The success of a testing project depends upon a well-written test plan document that is current at all times. Test Plan is more or less like a blueprint of how the testing activity is going to take place in a project.

Given below are few pointers on a Test Plan:

* + - Test Plan is a document that acts as a point of reference and only based on that testing is carried out within the QA team.
    - It is also a document that we share with the Business Analysts, Project Managers, Dev team and the other teams. This helps to enhance the level of transparency of the QA team’s work to the external teams.
    - It is documented by the QA manager/QA lead based on the inputs from the QA team members.
    - Test Planning is typically allocated with 1/3rd of the time that takes for the entire QA engagement.  The other 1/3rd is for Test Designing and the rest is for Test Execution.
    - This plan is not static and is updated on an on-demand basis.
    - The more detailed and comprehensive the plan is, the more successful will be the testing activity.

The test plan is that the project should be checked for all types of species of fishes. The pH level determined from the water will help in analyzing which types of fishes produces most waste, which type of fish can acclimatize in different situation. It will help in determining the humidity and temperature for the fishes.

* 1. **TEST CASES AND RESULTS**

In this time of need where garbage and waste has taken over our surrounding making it unhealthy and uninhabitable, this application will work as an interface between Naive users, the Employees and the NGOs to interact and work towards cleaner India. The aim is to provide a clean environment to the public and for that this application will target audiences such as City administration, District administrations/NGO(s), Municipalities, Waste trucks owning companies who will work towards cleaning the city. Users such as households, schools, restaurants, hospitals etc. will be using this application for raising issues regarding cleanliness.

This application is totally dedicated towards healthy and hygienic environment by creating awareness towards clean cities and its need.

* 1. **FUTURE ENHANCEMENT**

Currently, WASTE MANAGEMENT SYSTEM (WMS) which is developed as a website. Further, it can be extended to work as application on Android, OS, Windows and other platforms. Places outside the local limited range can also be included to increase the ease of usage for the users thereby providing much wider and better scope for the use of this website. The user interface is designed so as to cater to all the needs of a WMS.

**REFERENCES**

# [1]. "Introducing the MySQL 8 Document Store”

url: <https://www.apress.com/gp/book/9781484227244>

[2]. “MySQL 8.0 Reference Manual” url: “[https://downloads.mysql.com/docs/refman-8.0-en.pdf](https://downloads.mysql.com/docs/refman-8.0-en.pdf%20) “

[3]. Patrick Dalton, Paul Whitehead, “SQL SERVER BLACK BOOK”

Dream Tech New Delhi, 2005.

[4]. Roger Pressman, “Software Engineering A Practitioner’s Approach”, McGraw-Hill Company, 2001.