

A Project Report on

Automated Water Motor

Submitted in partial fulfillment of the requirements for the award of the Degree of
Bachelor of Technology

In

Department of Computer Science & Engineering

By

K. Sankar Suresh

G. V. N. Ramu

G. V. V. S. T. Bharath

B. Hosanna

14MQ1A0541

15MQ5A0510

15MQ5A0508

15MQ5A0507

Under the Esteemed Supervision of

Sri S. V. C. Gupta M.Tech.,

Professor



... Empowering Minds

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SRI VASAVI INSTITUTE OF ENGINEERING & TECHNOLOGY

Approved by AICTE, New Delhi & Affiliated to JNTUK, Kakinada

An ISO 9001:2008 Certified Institution

Nandamuru, Pedana Mandal, Krishna Dt. A.P.

2014 – 2018

DECLARATION

We are here by declaring that the project entitled “**Automated Water Motor**” is the work done by us. We certify that the work contained in this report is original and has been done by us under the guidance of our project guide. The work has not been submitted to any other Institute in preparing for any degree or diploma. We have followed the guidelines provided by our Institution in preparing the report. We have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute. Whenever we have used materials (data, theoretical analysis, figures, and text) from other sources, we have given due credit to them by citing them in the text of the report and giving their details in the references. Further, we have taken permission from the copyright owners of the sources, whenever necessary.

Students Signature

K. Sankar Suresh

14MQ1A0541

G. V. N. Ramu

15MQ5A0510

G. V. V. S. T. Bharath

15MQ5A0508

B. Hosanna

15MQ5A0507

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SRI VASAVI INSTITUTE OF ENGINEERING & TECHNOLOGY

Approved by AICTE, New Delhi & Affiliated to JNTUK, Kakinada

An ISO 9001:2008 Certified Institution

Nandamuru, Pedana Mandal, Krishna Dt. A.P.

2014 – 2018



CERTIFICATE

This is to certify that the thesis entitled “Automated Water Motor” is being submitted by

K. Sankar Suresh

14MQ1A0541

G. V. N. Ramu

15MQ5A0510

G. V. V. S. T. Bharath

15MQ5A0508

B. Hosanna

15MQ5A0507

In partial fulfillment of the requirements for the award of degree of B. Tech Computer Science & Engineering from Jawaharlal Nehru Technological University Kakinada is a record of bonafide work carried out by them at Sri Vasavi Institute of Engineering & Technology.

The results embodied in this Project report have not been submitted to any other University or Institute for the award of any degree or diploma.

Project Guide
&
Head of the department
Sri S. V. C Gupta

ACKNOWLEDGEMENT

We take great pleasure to express our deep sense of gratitude to our project guide Sri. S. V. C. Gupta, Professor and HOD, for his valuable guidance during the course of our project work.

We would like to express our heart-felt thanks to Sri A.B Srinivasa Rao, Principal, Sri Vasavi Institute of Engineering & Technology, Nandamuru for providing all the facilities for our project.

Our utmost thanks to all the Faculty members and Non Teaching Staff of the Department of Computer Science & Engineering for their support throughout our project work.

Our Family Members and Friends receive our deepest gratitude and love for their support throughout our academic year.

Project Associates

K. Sankar Suresh	14MQ1A0541
G. V. N. Ramu	15MQ5A0510
G. V. V. S. T. Bharath	15MQ5A0508
B. Hosanna	15MQ5A0507

ABSTRACT

This is an abstract to switch off the motor automatically when it is not pumping water. The reason may be the water source may be drained or the source from which the motor is getting water is not in contact with the pipe of motor. In these cases the even when the motor is not pumping the water goes on working which results in power wastage and degeneration of motor. Hence this is the IOT based solution to overcome this problem.

We will place a water detecting sensor in the way in which the motor pumps water. This sensor will be controlled by a microcontroller or a microprocessor. When we switch on the motor the microcontroller will be in charge for transferring power to the motor with the help of the relay. The microcontroller also controls the water detecting sensor.

When the motor is on it checks for some time whether the motor is pumping water or not. When the check time is over and if the water is not coming then it switches off the water motor and sends alert to the prescribed person.

We will store information when it is switching off automatically due to lack of water. By this information, we can identify the deficiency of water based on the seasons and regions. We can also know the underground water levels of a particular area. Hence by this information we will take necessary steps for improving water facility by that crop production can be improved even in such bad conditions.

CONTENTS

TITLES	PAGE NO.
TITLE PAGE	i
DECLARATION	ii
CERTIFICATE	iii
ACKNOWLEDGEMENT	iv
ABSTRACT	v
LIST OF FIGURES	vi
LIST OF ABBREVIATIONS	vii
LIST OF TABLES	vii
CHAPTER 1: INTRODUCTION	(1-3)
1.1 Motivation	1
1.2 Problem Definition	1
1.3 Introduction to IOT	1
1.4 Objectives of the project	3
CHAPTER 2: LITERATURE SURVEY	(4-5)
2.1 Introduction	4
2.2 Existing system	4
2.3 Disadvantages of existing system	4
2.4 Proposed system	5
2.5 Advantages Of Proposed System	5
CHAPTER 3: SYSTEM REQUIREMENTS	(6-20)
3.1 Introduction	6
3.2 Hardware and software requirements	6
3.3 About software used	7
3.3.1 Embedded C	7
3.3.2 Arduino UNO IDE	8
3.3.3 MIT App Inventor	12

3.3.4 IFTTT	14
3.4 About hardware used	16
3.4.1 NODE MCU	16
3.4.2 Water sensor	18
3.4.5 Jumper wires	19
3.5 Thingspeak cloud	19
3.6 Procedure	20
CHAPTER 4: SYSTEM DESIGN	(21-25)
4.1 Introduction	21
4.2 Feasibility study	22
4.3 Block diagram	23
4.4 Flow chart	24
CHAPTER 5: CODING AND IMPLEMENTATION	(26-36)
5.1 Introduction	26
5.1.1 Important Header files and methods/functions used:	26
5.2 Coding	31
CHAPTER 6: SYSTEM TESTING	(37-40)
6.1 Unit testing	38
6.2 Integrate testing	38
6.3 Functional testing	38
6.4 Test cases	40
CHAPTER 7: SCREEN SHOTS	(41-45)
CHAPTER 8: CONCLUSION	46
REFERENCES	47
APPENDIX	(48-49)

LIST OF FIGURES

S NO	FIGURE	NAME	PAGE NO
1	3.3.3.1	MIT App Inventor	14
2	3.4.1.1	Node MCU	18
3	3.4.1.2	Pin diagram	19
4	3.4.2	Water sensor	20
5	3.5	Storing the sensor data in the cloud	21
6	4.3	Block diagram	26
7	4.4	Flow chart	27
8	7.1	Program code in Arduino IDE	43
9	7.2	Uploading code to NodeMCU	43
10	7.3	Checking values in serial monitor	44
11	7.4	NodeMCU in operation (motor on)	44
12	7.5	Uploading data to cloud	45
13	7.6	NodeMCU in operation (motor off)	45
14	7.7	Thingspeak login page	46
15	7.8	Thingspeak channel view	46
16	7.9	Thingspeak fields view	47
17	7.10	MIT App Inventor home	47

LIST OF ABBREVIATIONS

IOT	Internet of Things
MCU	Microcontroller Unit
IDE	Integrated Development Environment
USB	Universal Serial Bus
PCB	Printed Circuit Board
MIT	Massachusetts Institute of Technology

LIST OF TABLES

S.NO	TABLE	NAME	PAGE NO
1	6.4.1	Test Cases	40

CHAPTER-1

INTRODUCTION

1. INTRODUCTION

1.1 MOTIVATION

An **automated water motor** is a system that automates the working of the water motor in agriculture. Using this system a farmer can remotely on and off the motor and can detect the dry run of motor. Detecting the dry run is very important as it damages the functioning of the motor.

The main reason for taking up this project is, in one of my friends' farm the motor has been damaged due to dry run. This dry run happens due to the lack of water in the source and not presence of farmer at that place at that time. Hence it would be profitable for farmers if we can stop dry run. Also it is very difficult for the farmers to personally come to farm and switch on and off the motor. Hence it is very necessary to automate the motor so that the farmer from remote place switch on and off the motor using a mobile app connected to internet. This can be done using Internet of Things (IoT). The data whether the motor is in dry run or not is detected by the sensors installed and information whether the motor is on and off is sent to cloud and this information is processed by microcontroller and it takes necessary steps according to the data.

1.2 PROBLEM DEFINITION

Automated water motor system is the process of switching on and off a motor from remote place and finding if there is dry run for the motor, if so the motor is turned off automatically and alerts the user. The dry run of the motor occurs due to lack of water in the source. This type of device is important because if we do not stop the motor in dry run it leads to the damage of the motor.

1.3 INTRODUCTION TO IOT

The Internet of things (IOT) is the inter-networking of physical devices, vehicles (also referred to as "connected devices" and "smart devices"), buildings, and other items—embedded with electronics, software, sensors, actuators, and network connectivity that enable these objects to collect and exchange data. In 2013 the Global Standards Initiative on Internet of Things

AUTOMATED WATER MOTOR

(IOT-GSI) defined the IOT as "the infrastructure of the information society." The IOT allows objects to be sensed or controlled remotely across existing network infrastructure, creating opportunities for more direct integration of the physical world into computer-based systems, and resulting in improved efficiency, accuracy and economic benefit in addition to reduced human intervention. When IOT is augmented with sensors and actuators, the technology becomes an instance of the more general class of cyber-physical systems, which also encompasses technologies such as smart grids, virtual power plants, smart homes, intelligent transportation and smart cities. Each thing is uniquely identifiable through its embedded computing system but is able to interoperate within the existing Internet infrastructure. Experts estimate that the IOT will consist of almost 50 billion objects by 2020.

Typically, IOT is expected to offer advanced connectivity of devices, systems, and services that goes beyond machine-to-machine (M2M) communications and covers a variety of protocols, domains, and applications. The interconnection of these embedded devices (including smart objects), is expected to usher in automation in nearly all fields, while also enabling advanced applications like a smart grid, and expanding to areas such as smart cities.

The **Internet of things (IoT)** is the network of physical devices, vehicles, home appliances and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these objects to connect and exchange data. Each thing is uniquely identifiable through its embedded computing system but is able to inter-operate within the existing Internet infrastructure.

The figure of online capable devices increased 31% from 2016 to 8.4 billion in 2017. Experts estimate that the IoT will consist of about 30 billion objects by 2020. It is also estimated that the global market value of IoT will reach \$7.1 trillion by 2020. 0

The IoT allows objects to be sensed or controlled remotely across existing network infrastructure, creating opportunities for more direct integration of the physical world into computer-based systems, and resulting in improved efficiency, accuracy and economic benefit in addition to reduced human intervention. When IoT is augmented with sensors and actuators, the technology becomes an instance of the more general class of cyber-physical systems, which also encompasses

AUTOMATED WATER MOTOR

technologies such as smart grids, virtual power plants, smart homes, intelligent transportation and smart cities.

"Things", in the IoT sense, can refer to a wide variety of devices such as heart monitoring implants, biochip transponders on farm animals, cameras streaming live feeds of wild animals in coastal waters, automobiles with built-in sensors, DNA analysis devices for environmental/food/pathogen monitoring, or field operation devices that assist firefighters in search and rescue operations. Legal scholars suggest regarding "things" as an "inextricable mixture of hardware, software, data and service".

These devices collect useful data with the help of various existing technologies and then autonomously flow the data between other devices.

The term "the Internet of things" was coined by Kevin Ashton of Procter & Gamble, later MIT's Auto-ID Center, in 1999.

1.4 OBJECTIVES OF THE PROJECT

- To switch on and off the motor using a mobile app.
- To gather the data from the sensor.
- If the motor was in dry run the motor should be automatically turn off.
- It also alerts users that the motor was turned off.

CHAPTER-2

LITERATURE SURVEY

2. LITERATURE SURVEY

2.1 INTRODUCTION

Literature survey is the most important step in software development process. Before developing the tool, it is necessary to determine the time factor i.e., the time complexity and economy that is the budget /cost of the system. Once these things are satisfied, then next step is to determine which operating system and language can be developing the tool by satisfying the hardware and software requirements of the system.

Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers from book or from web sites. Before building the system the above consideration are taken into account for developing the disposed system.

2.2 EXISTING SYSTEM

Mostly in the today's world the farmers are using the traditional methods like going to the farm and switch on and off the motor. This causes a huge burden for them and he has to go to farm everyday even when he has any other works. This traditional method is not anymore profitable in the current day world. Also even when the motor is in dry run there is no mechanism to automatically detect it without the intervention of farmer and switch it off.

2.3 DISADVANTAGES OF EXISTING SYSTEM

- The farmer should physically present at the farm to switch on the motor.
- If we use an android app to remotely switch on and off the motor also it does not stop dry run.
- It causes damage to motor.
- Ultimately results to wastage of money to farmer.
- Power wastage also occurs.

2.4 PROPOSED SYSTEM

In order to overcome the difficulties which are present in the existing system a new technology is evolved in the IOT platform which is “**AUTOMATED WATER MOTOR SYSTEM.**” The proposed system will continuously monitor if there was dry run when the motor is on. If so it will switch off the motor and alerts the user that there is dry run. All this is done by the microcontroller we use in this system. It detects the dry run using water sensor that is placed at the outlet of the motor. If the motor was on and the water is not pumped by the motor it detects and takes necessary steps. To automate the water motor we are choosing internet but there are existing system which use GSM technology for remote ON and OFF. This is also very less in cost and makes profits to the farmer.

2.5 ADVANTAGES OF PROPOSED SYSTEM

- Keeps motor safe.
- Saves electricity.
- Notifies user about lack of water.
- Automates water pumping in agriculture.
- Remote ON and OFF.

CHAPTER-3

SYSTEM REQUIREMENTS

3 SYSTEM REQUIREMENTS

3.1 INTRODUCTION

A **System Requirements Specification** (abbreviated SRS when need to be distinct from a Software Requirements Specification SRS) is a structured collection of information that embodies the requirements of a system.

A business analyst, sometimes titled system analyst, is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the systems development life cycle domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers.

In systems engineering and software engineering, **requirements analysis** encompasses those tasks that go into determining the needs or conditions to meet for a new or altered product or project, taking account of the possibly conflicting requirements of the various stakeholders, analyzing, documenting, validating and managing software or system requirements.

Requirements analysis is critical to the success or failure of a systems or software project. The requirements should be documented, actionable, measurable, testable, traceable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design.

3.2 HARDWARE AND SOFTWARE REQUIREMENTS

HARDWARE COMPONENTS USED:

- Node MCU (ESP8266-12E 1.0 Development Board):
- Water sensor
- Relay (230V)
- Jumper wires

AUTOMATED WATER MOTOR

SOFTWARE COMPONENTS USED:

- Arduino UNO IDE
- MIT app inverter
- Thingspeak cloud

3.3 ABOUT SOFTWARE USED

3.3.1 EMBEDDED C

Embedded C is a "set of language extensions" for the C Programming language by the C Standards committee to address commonality issues that exist between C extensions for different **embedded** systems. The **Internet of Things** (IoT) is the "**NETWORK**" of physical objects—devices, vehicles, buildings and other items embedded with electronics, software, sensors, and network connectivity that enables these objects to collect and exchange data. These are the Google definitions of the subjects. One may need embedded C while working in IoT as a means to write software part. IoT involves working with Embedded systems which may need embedded C.

Embedded C is a set of language extensions for the C programming language by the C Standards Committee to address commonality issues that exist between C extensions for different embedded systems. Historically, embedded C programming requires nonstandard extensions to the C language in order to support exotic features such as fixed-point arithmetic, multiple distinct memory banks, and basic I/O operations

In 2008, the C Standards Committee extended the C language to address these issues by providing a common standard for all implementations to adhere to. It includes a number of features not available in normal C, such as, fixed-point arithmetic, named address spaces, and basic I/O hardware addressing.

Embedded C uses most of the syntax and semantics of standard C, e.g., main () function, variable definition, data type declaration, conditional statements (if, switch case), loops (while, for), functions, arrays and strings, structures and union, bit operations, macros, etc.

3.3.2 ARDUINO UNO IDE

Arduino is an open source physical computing platform based on a simple input/output (I/O) board and a development environment that implements the Processing language. Arduino can be used to develop standalone interactive objects or can be connected to software on your computer (such as Flash, Processing, or Max/MSP). The boards can be assembled by hand or purchased preassembled; the open source IDE (Integrated Development Environment) can be downloaded for free from www.arduino.cc.

The Arduino Uno is a microcontroller board based on the ATmega328 . It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

Arduino is a single board microcontroller to make using electronics in multidisciplinary projects more accessible. The software consists of a standard programming language compiler and a boot loader that executes on the microcontroller.

There are different Arduino boards available in the market. The many flavors of arduino are listed below. In our Project we are using Arduino UNO board. because Arduino UNO board Compatible futures to meet our project requirements.

- Arduino Uno
- Arduino Leonardo
- Arduino LilyPad
- Arduino Mega
- Arduino Nano
- Arduino Mini

AUTOMATED WATER MOTOR

- Arduino Mini Pro
- Arduino BT

FEATURES:

- » It is a multiplatform environment; it can run on Windows, Mac, and Linux.
- » It is based on the Processing programming IDE, an easy-to-use development environment used by artists and designers.
- » You program it via a USB cable, not a serial port. This feature is useful, because many modern computers don't have serial ports.
- » It is open source hardware and software if you wish, you can download the circuit diagram, buy all the components, and make your own, without paying anything to the makers of Arduino.
- » The hardware is cheap. The USB board costs about €20 (currently, about US\$35) and replacing a burnt-out chip on the board is easy and costs no more than €5 or US\$4. So you can afford to make mistakes.
- » There is an active community of users, so there are plenty of people who can help you.
- » The Arduino Project was developed in an educational environment and is therefore great for newcomers to get things working quickly.

A Software Serial library allows for serial communication on any of the Uno's digital pins. The ATmega328 also support I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus.

AUTOMATED WATER MOTOR

HISTORY

The Arduino project started at the Interaction Design Institute Ivrea (IDII) in Ivrea, Italy. At that time, the students used a BASIC Stamp microcontroller at a cost of \$100, a considerable expense for many students. In 2003 Hernando Barragán created the development platform Wiring as a Master's thesis project at IDII, under the supervision of Massimo Banzi and Casey Reas, who are known for work on the Processing language. The project goal was to create simple, low cost tools for creating digital projects by non-engineers. The Wiring platform consisted of a printed circuit board (PCB) with an ATmega168 microcontroller, an IDE based on Processing and library functions to easily program the microcontroller. In 2003, Massimo Banzi, with David Mellis, another IDII student, and David Cuartielles, added support for the cheaper ATmega8 microcontroller to Wiring. But instead of continuing the work on Wiring, they forked the project and renamed it Arduino.

The initial Arduino core team consisted of Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, and David Mellis, but Barragán was not invited to participate.

Following the completion of the Wiring platform, lighter and less expensive versions were distributed in the open-source community.

Adafruit Industries, a New York City supplier of Arduino boards, parts, and assemblies, estimated in mid-2011 that over 300,000 official Arduinos had been commercially produced, and in 2013 that 700,000 official boards were in users' hands.

In October 2016, Federico Musto, Arduino's former CEO, secured a 50% ownership of the company. In April 2017, Wired reported that Musto had "fabricated his academic record.... On his company's website, personal LinkedIn accounts, and even on Italian business documents, Musto was until recently listed as holding a PhD from the Massachusetts Institute of Technology. In some cases, his biography also claimed an MBA from New York University." Wired reported that neither University had any record of Musto's attendance, and Musto later admitted in an interview with Wired that he had never earned those degrees.

Around that same time, Massimo Banzi announced that the Arduino Foundation would be "a new beginning for Arduino." But a year later, the Foundation still hasn't been established, and the state of the project remains unclear.

AUTOMATED WATER MOTOR

The controversy surrounding Musto continued when, in July 2017, he reportedly pulled many Open source licenses, schematics, and code from the Arduino website, prompting scrutiny and outcry.

In October 2017, Arduino announced its partnership with ARM Holdings (ARM). The announcement said, in part, "ARM recognized independence as a core value of Arduino ... without any lock-in with the ARM architecture." Arduino intends to continue to work with all technology vendors and architectures.

How an Arduino Program Works ?

The Arduino community calls a program a sketch.

It has two main functions: **setup and loop**

```
void setup ( ){      // runs once, when the Arduino is powered on.

}

void loop ( ){      // runs continuously after the setup() has completed.
    here we check for voltage level on the inputs, and turn on/off the outputs.

}
```

3.3.3 MITAPP INVENTOR

App Inventor for Android is an open-source web application originally provided by Google, and now maintained by the Massachusetts Institute of Technology (MIT).

It allows newcomers to computer programming to create software applications for the Android operating system (OS). It uses a graphical interface, very similar to Scratch and the StarLogo TNG user interface, which allows users to drag-and-drop visual objects to create an application that can run on Android devices. In creating App Inventor, Google drew upon significant prior research in educational computing, as well as work done within Google on online development environments.

App Inventor and the projects on which it is based are informed by constructionist learning theories, which emphasizes that programming can be a vehicle for engaging powerful ideas through active learning. As such, it is part of an ongoing movement in computers and education that began with the work of Seymour Papert and the MIT Logo Group in the 1960s and has also manifested itself with Mitchel Resnick's work on Lego Mindstorms and StarLogo.

MIT App Inventor is also supported with the Firebase Database extension. This allows people to store data on Google's firebase.

The application was made available through request on July 12, 2010, and released publicly on December 15, 2010. The App Inventor team was led by Hal Abelson and Mark Friedman. In the second half of 2011, Google released the source code, terminated its server, and provided funding for the creation of The MIT Center for Mobile Learning, led by App Inventor creator Hal Abelson and fellow MIT professors Eric Klopfer and Mitchel Resnick. The MIT version was launched in March 2012.

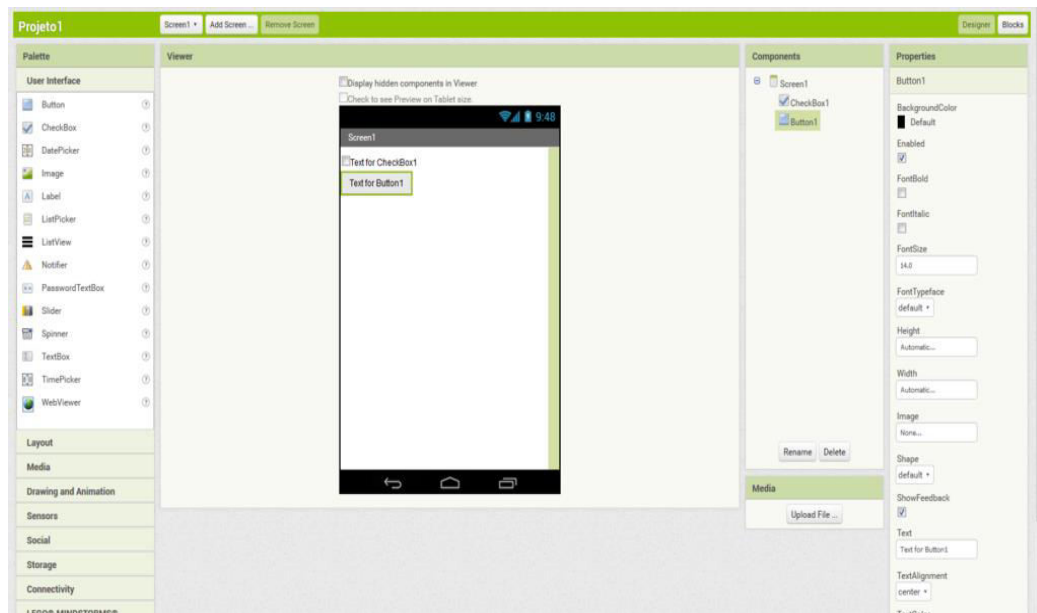


FIG:3.3.3.1 MIT APP INVERTER

HISTORY

The application was made available through request on July 12, 2010, and released publicly on December 15, 2010. The App Inventor team was led by Hal Abelson and Mark Friedman. In the second half of 2011, Google released the source code, terminated its server, and provided funding for the creation of The MIT Center for Mobile Learning, led by App Inventor creator Hal Abelson and fellow MIT professors Eric Klopfer and Mitchel Resnick. The MIT version was launched in March 2012.

On December 6, 2013 (the start of the Hour of Code), MIT released App Inventor 2, renaming the original version "App Inventor Classic" Major differences are:

- The blocks editor in the original version ran in a separate Java process, using the Open Blocks Java library for creating visual blocks programming languages and programming.

Open Blocks is distributed by the Massachusetts Institute of Technology's Scheller Teacher Education Program (STEP) and is derived from master's thesis research by Ricarose Roque. Professor Eric Klopfer and Daniel Wendel of the Scheller Program supported the distribution of

AUTOMATED WATER MOTOR

Open Blocks under an MIT License. Open Blocks visual programming is closely related to StarLogo TNG, a project of STEP, and Scratch, a project of the MIT Media Lab's Lifelong Kindergarten Group. App Inventor 2 replaced Open Blocks with Blockly, a blocks editor that runs within the browser.

- The MIT AI2 Companion app enables real-time debugging on connected devices via Wi-Fi, not just USB.

As of May 2014, were 87 thousand weekly active users of the service and 1.9 million registered users in 195 countries for a total of 4.7 million apps built.

As December 2015, had 140k weekly active users and 4 million registered users in 195 countries, run total of 12 million built applications.

3.3.4 IFTTT

If This Then That, also known as **IFTTT** (pronounced), is a free web-based service to create chains of simple conditional statements, called applets.

An applet is triggered by changes that occur within other web services such as Gmail, Facebook, Telegram, Instagram, or Pinterest.

For example, an applet may send an e-mail message if the user tweets using a hashtag, or copy a photo on Facebook to a user's archive if someone tags a user in a photo.

In addition to the web-based application, the service runs on iOS and Android. In February 2015, IFTTT renamed their original application to IF, and released a new suite of apps called Do which lets users create shortcut applications and actions. As of 2015, IFTTT users created about 20 million recipes each day. All of the functionalities of the Do suite of apps have since been integrated into a redesigned IFTTT app.

IFTTT is an initialism for If This Then That.

AUTOMATED WATER MOTOR

HISTORY

On December 14, 2010, Linden Tibbets, the co-founder of IFTTT, posted a blog post titled “ifttt the beginning...” on the IFTTT website, announcing the new project. The first IFTTT applications were designed and developed by Tibbets and co-founder Jesse Tane. On September 7, 2011, Tibbets announced the launch on the official website.

By April 30, 2012, users had created one million tasks. In June 2012, the service entered the Internet of Things space by integrating with Belkin WeMo devices, allowing recipes to interact with the physical world. On July 10, 2013, IFTTT released an iPhone app and later released a version for iPad and iPod touch. On April 24, 2014, IFTTT released a version for Android. By the end of 2014, the IFTTT business was valued at approximately USD 170 million.

On February 19, 2015, IFTTT launched three new applications. Do Button triggers an action when you press it. Do Camera automatically uploads the image to the service of your choice (Facebook, Twitter, Dropbox, etc.). Do Notes does the same as Do Camera, except with notes instead of images. As of November 2016, the four apps have been combined into one. By December 2016, the company announced a partnership with JotForm to integrate an "Applet" to create actions in other applications.

IFTTT employs the following concepts:

- Services (formerly known as channels) are the basic building blocks of IFTTT. They mainly describe a series of data from a certain web service such as YouTube or eBay. Services can also describe actions controlled with certain APIs, like SMS. Sometimes, they can represent information in terms of weather or stocks. Each service has a particular set of triggers and actions.
- Triggers are the "this" part of an applet. They are the items that trigger the action. For example, from an RSS feed, you can receive a notification based on a keyword or phrase.
- Actions are the "that" part of an applet. They are the output that results from the input of the trigger.

AUTOMATED WATER MOTOR

- Applets (formerly known as recipes) are the predicates made from Triggers and Actions. For example, if you like a picture on Instagram (trigger), an IFTTT app can send the photo to your Dropbox account (action).
- Ingredients are basic data available from a trigger—from the email trigger, for example; subject, body, attachment, received date, and sender's address.

3.4 ABOUT HARDWARE USED

3.4.1 NODE MCU

NodeMCU Dev Board is based on widely explored esp8266 System on Chip from Express if. It combined features of WIFI access point and station + microcontroller and uses simple LUA based programming language. ESP8266 Node MCU offers-

- Arduino-like hardware IO
- Event-driven API for network applications
- 10 GPIOs D0-D10, PWM functionality, IIC and SPI communication, 1-Wire and ADC A0 etc. all in one board
- WIFI networking (can be uses as access point and/or station, host a web server), connect to internet to fetch or upload data.

Node MCU is an open source IOT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. The term "NodeMCU" by default refers to the firmware rather than the dev kits. The firmware uses the Lua scripting language. It is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson, and spiffs.ESP8266 has been designed for mobile, wearable electronics and Internet of Things applications with the aim of achieving the lowest power consumption with a combination of several proprietary techniques.

ESP8266 is an impressive, low cost WiFi module suitable for adding WiFi functionality to an existing microcontroller project via a UART serial connection.

AUTOMATED WATER MOTOR

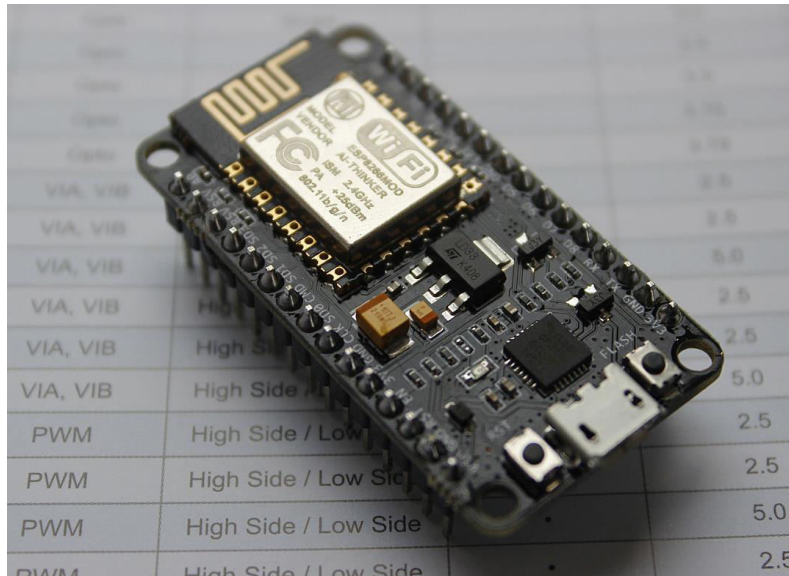


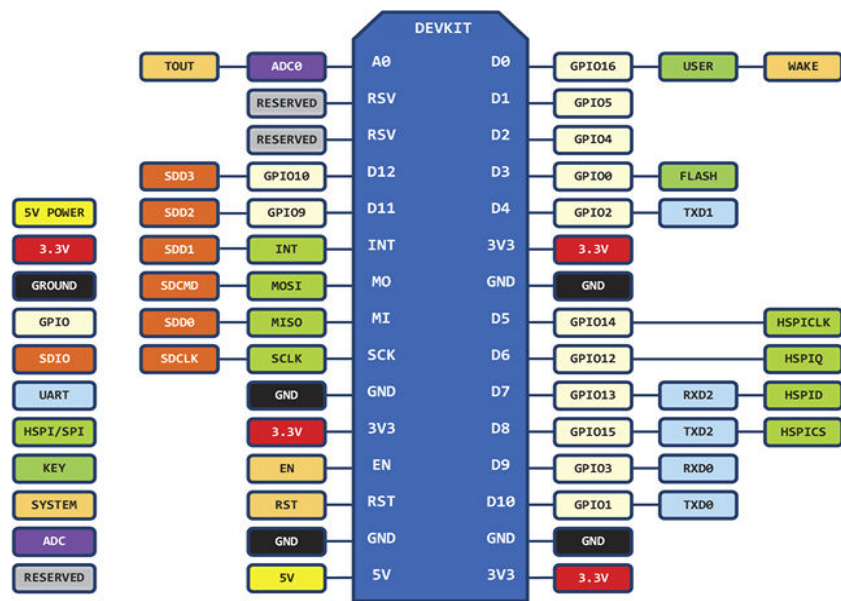
Fig:- 3.4.1.1 NODEMCU

Summary:

Microcontroller	:	ESP-8266EX
Operating Voltage	:	3.3V
Digital I/O Pins	:	11
Analog Pins	:	1(Max input: 3.2V)
Clock Speed	:	80MHZ/160MHZ
Flash	:	4MB
Length	:	34.2mm
Width	:	25.6mm
Weight	:	10gm

AUTOMATED WATER MOTOR

Pin Definition:



D0(GPIO16) can only be used as gpio read/write, no interrupt supported, no pwm/i2c/ow supported.

Fig:-3.4.1.2 PIN DIAGRAM

3.4.2 WATER SENSOR

A water sensor is a device used in the detection of the water level for various applications. Water sensors are of several types that include ultrasonic sensors, pressure transducers, bubblers, and float sensors. Water sensors find applications in nuclear power plants, automobiles for measuring the amount of gasoline left in the fuel tank, engine oil, cooling water, and brake/power steering fluid.

FEATURES:-

- Fast response and High sensitivity.
- Stable and long life.
- Simple drive circuit.

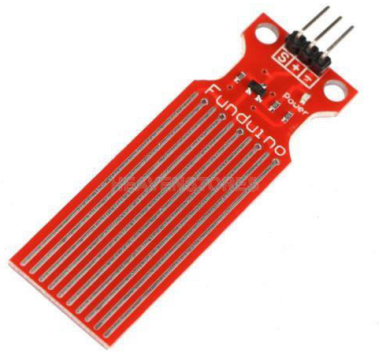


Fig:-3.4.2 WATER SENSOR

3.4.3 JUMPING WIRES

A **jump wire** (also known as jumper, jumper wire, jumper cable, DuPont wire, or DuPont cable – named for one manufacturer of them) 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.

3.5 THINGSPEAK CLOUD

According to its developers, "**ThingSpeak** is an open source Internet of Things(IOT) application and API to store and retrieve data from things using the HTTP protocol over the Internet or via a Local Area Network. ThingSpeak enables the creation of sensor logging applications, location tracking applications, and a social network of things with status updates".^[2]

ThingSpeak was originally launched by ioBridge in 2010 as a service in support of IOT applications.

ThingSpeak has integrated support from the numerical computing software MATLAB from MathWorks,^[4] allowing ThingSpeak users to analyze and visualize uploaded data using Matlab without requiring the purchase of a Matlab license from Mathworks.

ThingSpeak has a close relationship with Mathworks, Inc. In fact, all of the ThingSpeak documentation is incorporated into the Mathworks' Matlab documentation site and even enabling registered Mathworks user accounts as valid login credentials on the ThingSpeak website.^[5] The

AUTOMATED WATER MOTOR

terms of service and privacy policy of ThingSpeak.com are between the agreeing user and Mathworks, Inc.

ThingSpeak has been the subject of articles in specialized "Maker" websites like Instructables, Codeproject, and Channel9.

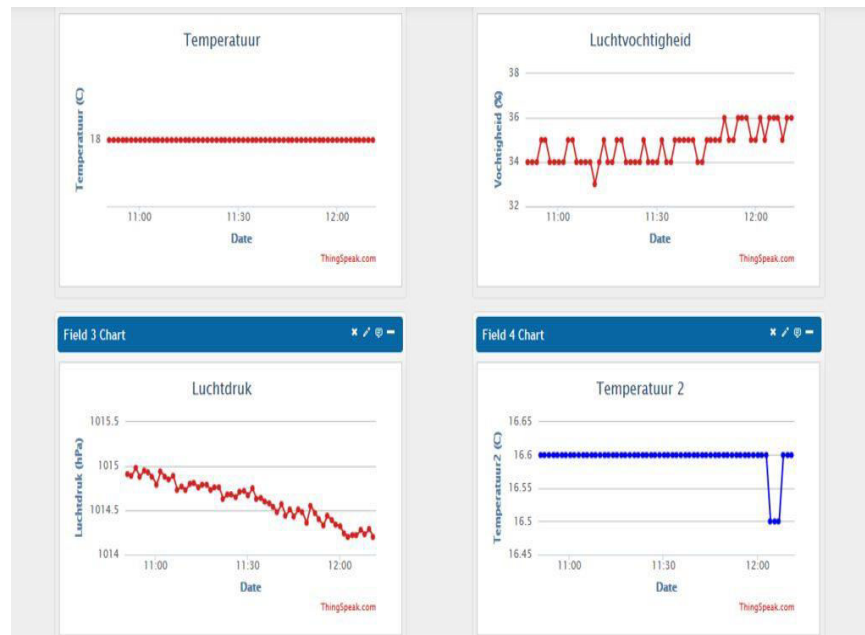


FIG:-3.5 STORING THE SENSOR DATA IN THE CLOUD

3.6 PROCEDURE

- Connect the Power supply to the Controller.
- Initialize the Arduino, Node MCU, Sensor Board.
- Arrange the connections as per block Diagram.
- Open the Arduino IDE Software and Write the Source Code, Define the pins.
- Interface the Sensor Connections According to the Pins defined in the Source code
- Load the Program into Arduino. If any error is present rectify it and Save. Again load.
- Now go to Serial Monitor and Observe the output of Sensor .
- Upload the data into Cloud through Node MCU.
- Login into Android app and take the proper action by calling to others.

CHAPTER-4

SYSTEM DESIGN

4 SYSTEM DESIGN

4.1 INTRODUCTION

Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. The design phase is when you build the plan for how you will take your project through the rest of the SDL process from implementation, to verification, to release. During the design phase you establish best practices to follow for this phase by way of functional and design specifications, and you perform risk analysis to identify threats and vulnerabilities in your software.

The design phase is consisting of two activities. The activities are:-

- A. **Logical design** - This activity involves class diagram and interaction diagram.
- B. **Interface design** - This activity includes navigation design, output design, Input design.

If the broader topic of product development "blends the perspective of marketing, design, and manufacturing into a single approach to product development," then design is the act of taking the marketing information and creating the design of the product to be manufactured. Systems design is therefore the process of defining and developing systems to satisfy specified requirements of the user.

This project can divide into 3 modules mobile App, cloud and hardware at the motor. Initially the user turns ON the motor using mobile App. This will send data to cloud about the switching ON of the motor. The microcontroller unit at the motor receives this information and switches ON the motor. The same thing happens with the switching OFF the motor. In case of dry run the microcontroller unit automatically switches the motor and sends data about this to cloud and notification to user.

The data we upload while switching ON and OFF the motor are 0 and 1. The value 0 refers that the motor currently is in resting position and the value 1 refers to that the motor currently is in

AUTOMATED WATER MOTOR

running state. If dry run occurs we will update another field in cloud with data value 1 which is cause, this cause field says that there is an abnormal OFF. The cause field can have either 0 or 1.

4.2 FEASIBILITY STUDY

Feasibility study is an assessment of the practicality of a proposed project or system. A feasibility study aims to objectively and rationally uncover the strengths and weaknesses of an existing business or proposed venture, opportunities and threats present in the natural environment, the resources required to carry through, and ultimately the prospects for success. In its simplest terms, the two criteria to judge feasibility are cost required and value to be attained.

A well-designed feasibility study should provide a historical background of the business or project, a description of the product or service, accounting statements, details of the operations and management, marketing research and policies, financial data, legal requirements and tax obligations. Generally, feasibility studies precede technical development and project implementation.

A feasibility study evaluates the project's potential for success; therefore, perceived objectivity is an important factor in the credibility of the study for potential investors and lending institutions. It must therefore be conducted with an objective, unbiased approach to provide information upon which decisions can be based.

A project feasibility study is a comprehensive report that examines in detail the five frames of analysis of a given project. It also takes into consideration its four Ps, its risks and POVs, and its constraints (calendar, costs, and norms of quality). The goal is to determine whether the project should go ahead, be redesigned, or else abandoned altogether.

The five frames of analysis are:

- The frame of definition.
- The frame of contextual risks.
- The frame of potentiality.
- The parametric frame.

AUTOMATED WATER MOTOR

- The frame of dominant and contingency strategies.

This product provides the following services

- Switching ON/OFF.
- Maintains data in the cloud.
- Detects dry run and stops the motor.
- Installation and product service will be provided freely till warranty expires.

These type of products are available in the market in GSM technology and also they are available at huge cost but this system is available at low cost that is under Rs.2000/-. The cost of the parts used in this project are NodeMCU Rs.500/-, relay Rs.100/-, jumper wires Rs.100/-, Water sensor Rs.200/- we have bought a small motor for the experimentation which does not be included actually in the project costs Rs. 300/-

4.3 BLOCK DIAGRAM

A **block diagram** is a diagram of a system in which the principal parts or functions are represented by blocks connected by lines that show the relationships of the blocks. They are heavily used in engineering in hardware design, electronic design, software design, and process flow diagrams.

Block diagrams are typically used for higher level, less detailed descriptions that are intended to clarify overall concepts without concern for the details of implementation. Contrast this with the schematic diagrams and layout diagrams used in electrical engineering, which show the implementation details of electrical components and physical construction.

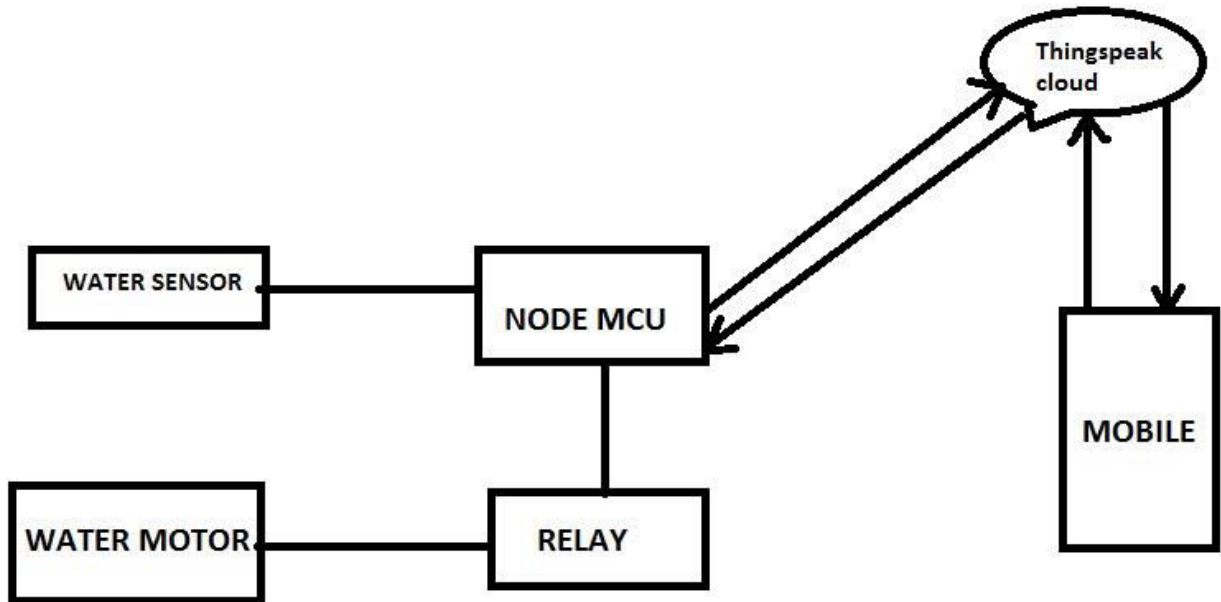


FIG:-4.3 BLOCK DIAGRAM

4.4 FLOW CHART

A **flowchart** is a type of diagram that represents an algorithm, workflow or process, showing the steps as boxes of various kinds, and their order by connecting them with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analyzing, designing, documenting or managing a process or program in various fields.

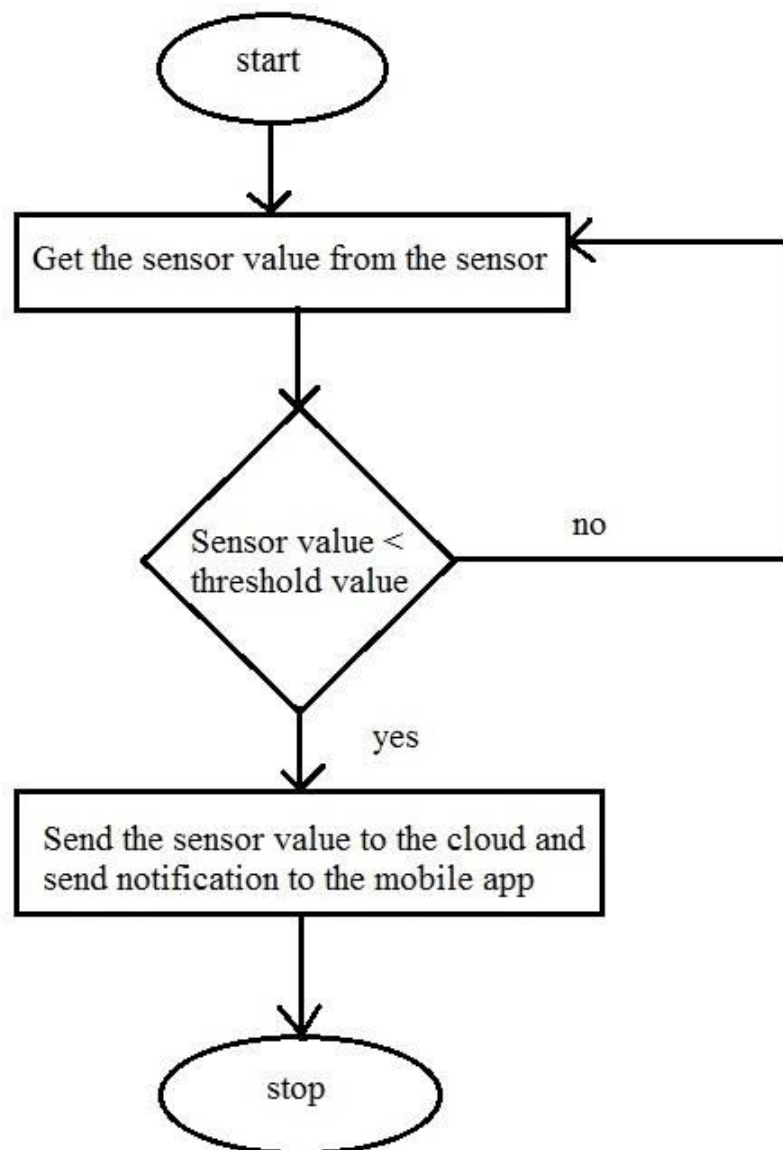


FIG:-4.4 FLOW CHART

CHAPTER-5

CODING AND IMPLEMENTATION

5 CODING AND IMPLEMENTATION

5.1 INTRODUCTION

Coding is the process of designing, writing, testing, debugging, and maintaining the source code of computer programs. This source code is written in one or more programming languages (such as C++, C#, java, Python, Smalltalk, etc..). The purpose of programming is to create a set of instructions that computers use to perform specific operations or to exhibit desired behaviors. The process of writing source code often requires expertise in many different subjects, including knowledge of the application domain, specialized algorithms and formal logic.

5.1.1 Important Header files and methods/functions used:

#include <ESP8266WiFi.h>

at the starting of Arduino code & start using the Classes associated with this library.

ESP8266 WiFi library is designed after the the standard Arduino WiFi library but has even more functionality than the standard Arduino library .The standard Arduino WiFi library is used for the WiFi shield or with the Arduino boards like YUN with inbuilt WiFi.

With the introduction of ESP8266 , the WiFi functionality has become much cheaper & easy to use.

The WiFi library has many classes that you can use. Each class has functions in it that are specific to that class. There's the WiFi class, the IP address class, the server class, the client class and the UDP class.

WiFi.begin(ssid,pass);

The possible return values are

WL_CONNECTED after successful connection is established with the Access Point

AUTOMATED WATER MOTOR

WL_IDLE_STATUS when Wi-Fi is in process of changing between statuses

WL_NO_SSID_AVAIL in case configured SSID cannot be reached

WL_CONNECT_FAILED if password is incorrect

WL_DISCONNECTED if module is not configured in station mode

The status function in the WiFi class, doesn't take any arguments but it returns stuff depending on the status of the network that you're connected to.

Usually, first, you call `WiFi.begin`, you pass the SSID and the password because you're trying to establish a connection with the network. Then, what you do is you wait in a loop until `WiFi.status` returns the value `WL_CONNECTED`.

WiFi.status()

Connection process can take couple of seconds and we are checking for this to complete in the following loop:

```
while (WiFi.status() != WL_CONNECTED)
{
    delay(500);
    Serial.print(".");
}
```

The `while()` loop will keep looping while `WiFi.status()` is other than `WL_CONNECTED`. The loop will exit only if the status changes to `WL_CONNECTED`.

Client.connect()

Description

Connects to a specified IP address and port. The return value indicates success or failure. Also supports DNS lookups when using a domain name.

Syntax

```
client.connect()  
client.connect(ip,port)  
client.connect(URL, port)
```

Parameters

IP: the IP address that the client will connect to (array of 4 bytes)

URL: the domain name the client will connect to (string, ex.: "arduino.cc")

port: the port that the client will connect to (int)

Returns

Returns an int (1,-1,-2,-3,-4) indicating connection status :

- SUCCESS 1
- TIMED_OUT -1
- INVALID_SERVER -2
- TRUNCATED -3
- INVALID_RESPONSE -4

Client.print();

Description

Print data to the server that a client is connected to. Prints numbers as a sequence of digits, each an ASCII character (e.g. the number 123 is sent as the three characters '1', '2', '3').

Syntax

```
client.print(data)  
client.print(data, BASE)
```

AUTOMATED WATER MOTOR

Parameters

data: the data to print (char, byte, int, long, or string)

BASE (optional): the base in which to print numbers: DEC for decimal (base 10), OCT for octal (base 8), HEX for hexadecimal (base 16).

Returns

byte: returns the number of bytes written, though reading that number is optional

client.stop()

Description

Disconnect from the server.

Syntax

client.stop()

Serial.begin()

Description

Sets the data rate in bits per second (baud) for serial data transmission. For communicating with the computer, use one of these rates: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, or 115200. You can, however, specify other rates - for example, to communicate over pins 0 and 1 with a component that requires a particular baud rate.

An optional second argument configures the data, parity, and stop bits. The default is 8 data bits, no parity, one stop bit.

Syntax

Serial.begin(speed)

Serial.begin(speed, config)

AUTOMATED WATER MOTOR

pinMode()

Description

Configures the specified pin to behave either as an input or an output. See the description of ([digital pins](#)) for details on the functionality of the pins.

As of Arduino 1.0.1, it is possible to enable the internal pullup resistors with the mode INPUT_PULLUP. Additionally, the INPUT mode explicitly disables the internal pullups.

Syntax

```
pinMode(pin, mode)
```

Parameters

pin: the number of the pin whose mode you wish to set

mode: INPUT, OUTPUT, or INPUT_PULLUP. (see the ([digital pins](#)) page for a more complete description of the functionality.)

digitalWrite()

Description

Write a HIGH or a LOW value to a digital pin.

If the pin has been configured as an OUTPUT with pinMode(), its voltage will be set to the corresponding value: 5V (or 3.3V on 3.3V boards) for HIGH, 0V (ground) for LOW.

If the pin is configured as an INPUT, digitalWrite() will enable (HIGH) or disable (LOW) the internal pullup on the input pin. It is recommended to set the pinMode() to INPUT_PULLUP to enable the internal pull-up resistor. See the digital pins tutorial for more information.

If you do not set the pinMode() to OUTPUT, and connect an LED to a pin, when calling digitalWrite(HIGH), the LED may appear dim. Without explicitly setting pinMode(), digitalWrite() will have enabled the internal pull-up resistor, which acts like a large current-limiting resistor.

Syntax

```
digitalWrite(pin, value)
```

5.2 CODING

```
#include <ESP8266WiFi.h>

const char* ssid    = "Not Available";
const char* password = "notavailable";
const char* host = "api.thingspeak.com";
const char* server = "api.thingspeak.com";
String apiKey = "N1UU0M33U6CMSRGP";

void setup()
{
    Serial.begin(115200);
    delay(10);
    pinMode(D0,OUTPUT);    //to the relay
    pinMode(A0,INPUT);    //from the water sensor
    digitalWrite(D0,HIGH);

    // We start by connecting to a WiFi network
    Serial.print("Connecting to ");
    Serial.println(ssid);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
}
```

AUTOMATED WATER MOTOR

```
        Serial.println();

        Serial.println("WiFi connected");

        Serial.println("IP address: ");

        Serial.println(WiFi.localIP());
    }

    int state = 0;

    int sensorvalue;

    String want;

    void loop()
    {
        delay(5000);

        Serial.print("connecting to ");

        Serial.println(host);

        // Use WiFiClient class to create TCP connections
        WiFiClient client;

        const int httpPort = 80;

        if (!client.connect(host, httpPort)) {
            Serial.println("connection failed");

            return;
        }

        // We now create a URI for the request

        String url = "/channels/409368/fields/1/last?api_key=LBQRU9ZYYATF2X91";

        Serial.print("Requesting URL: ");

        Serial.println(url);
```

```
String line="hello";

// This will send the request to the server
client.print(String("GET ") + url + " HTTP/1.0\r\n" + "Host: " + host + "\r\n" +
    "Connection: close\r\n\r\n");

delay(5000);

// Read all the lines of the reply from server and print them to Serial
while(client.available())
{
    line = client.readStringUntil('\r');
    //Serial.print(line);
}

Serial.print(line);
Serial.println();

client.stop();

int data=line.toInt();

if(data==1)
{
    if(state==1)
    {
    }
    else if(state==0)
    {
        digitalWrite(D0,LOW); //On the motor
        state=1;
    }
}
```

```
        }
    }
    else if(data==0)
    {
        if(state==1)
        {
            digitalWrite(D0,HIGH); //off the motor
            state=0;
        }
        else if(state==0)
        {
        }
    }
    //Serial.println("closing connection");
    client.stop();
    delay(5000);
    if(state==1)
    {
        sensorvalue=analogRead(A0);
        Serial.println(sensorvalue);
        if(sensorvalue<50)
        {
            digitalWrite(D0,HIGH); //off the motor
            state=0;
        }
    }
}
```



```
if (!client.connect(server, httpPort)) {  
    Serial.println("connection failed");  
    return;  
}  
String url = "/update";  
url += "?api_key=";  
url += apiKey;  
url += "&field1=";  
url += "0";  
Serial.print("Requesting URL: ");  
Serial.println(url);  
// This will send the request to the server  
client.print(String("GET ") + url + " HTTP/1.1\r\n" +  
"Host: " + server + "\r\n" + "Connection: close\r\n\r\n");  
delay(5000);  
while(client.connected() && !client.available())  
    delay(1);  
//waits for    data  
while (client.connected() || client.available())  
{  
    char charIn = client.read();  
    //Serial.print(charIn);  
}  
Serial.println();
```

AUTOMATED WATER MOTOR

```
        //Serial.println("closing connection");
        client.stop();
        //delay(10000);
    }
}
}
```

CHAPTER-6

SYSTEM TESTING

6 SYSTEM TESTING

Software testing is an investigation conducted to provide stakeholders with information about the quality of the software product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding software bugs (errors or other defects), and verifying that the software product is fit for use.

Software testing involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test

- meets the requirements that guided its design and development,
- responds correctly to all kinds of inputs,
- performs its functions within an acceptable time,
- is sufficiently usable,
- can be installed and run in its intended environments, and
- achieves the general result its stakeholders desire.

As the number of possible tests for even simple software components is practically infinite, all software testing uses some strategy to select tests that are feasible for the available time and resources. As a result, software testing typically (but not exclusively) attempts to execute a program or application with the intent of finding software bugs (errors or other defects). The job of testing is an iterative process as when one bug is fixed, it can illuminate other, deeper bugs, or can even create new ones.

Software testing can provide objective, independent information about the quality of software and risk of its failure to users or sponsors.

Software testing can be conducted as soon as executable software (even if partially complete) exists. The overall approach to software development often determines when and how testing is conducted. For example, in a phased process, most testing occurs after system requirements have

AUTOMATED WATER MOTOR

been defined and then implemented in testable programs. In contrast, under an agile approach, requirements, programming, and testing are often done concurrently.

6.1 UNIT TESTING

In the coding process of this software the unit testing conducted include

- testing of piece of code that connects to WIFI
- testing of code that gets and uploads data to the cloud
- testing of code that switches ON and OFF the motor
- testing of the code that takes input from sensor

6.2 INTEGRATE TESTING

In the project we integrated the entire code and tested whether all the modules working individually are working as a group or not. The MCU connected to the WIFI and could get data from cloud and worked according to it. It then uploaded data in case of dry run and the app in the mobile can successfully control the motor.

6.3 FUNCTIONAL TESTING

Functional testing is a quality assurance (QA) process and a type of black-box testing that bases its test cases on the specifications of the software component under test. Functions are tested by feeding them input and examining the output, and internal program structure is rarely considered (unlike white-box testing). Functional testing usually describes what the system does.

Functional testing does not imply that you are testing a function (method) of your module or class. Functional testing tests a slice of functionality of the whole system.

Functional testing differs from system testing in that functional testing "verifies a program by checking it against ... design document(s) or specification(s)", while system testing "validate[s] a program by checking it against the published user or system requirements".

Functional testing has many types:

AUTOMATED WATER MOTOR

- Smoke testing
- Sanity testing
- Regression testing
- Usability testing

SIX STEPS:-

Functional testing typically involves six steps

1. The identification of functions that the software is expected to perform
2. The creation of input data based on the function's specifications
3. The determination of output based on the function's specifications
4. The execution of the test case
5. The comparison of actual and expected outputs
6. To check whether the application works as per the customer need.

In this project we have tested for the test cases that once sending the input of ON by the android app in a mobile then the motor switched on. Also we have done the same with Switching OFF the mptor it also worked. Then we switch ON the motor and removed the water then the motor automatically switched OFF thus the project successfully worked.

6.4 TEST CASES

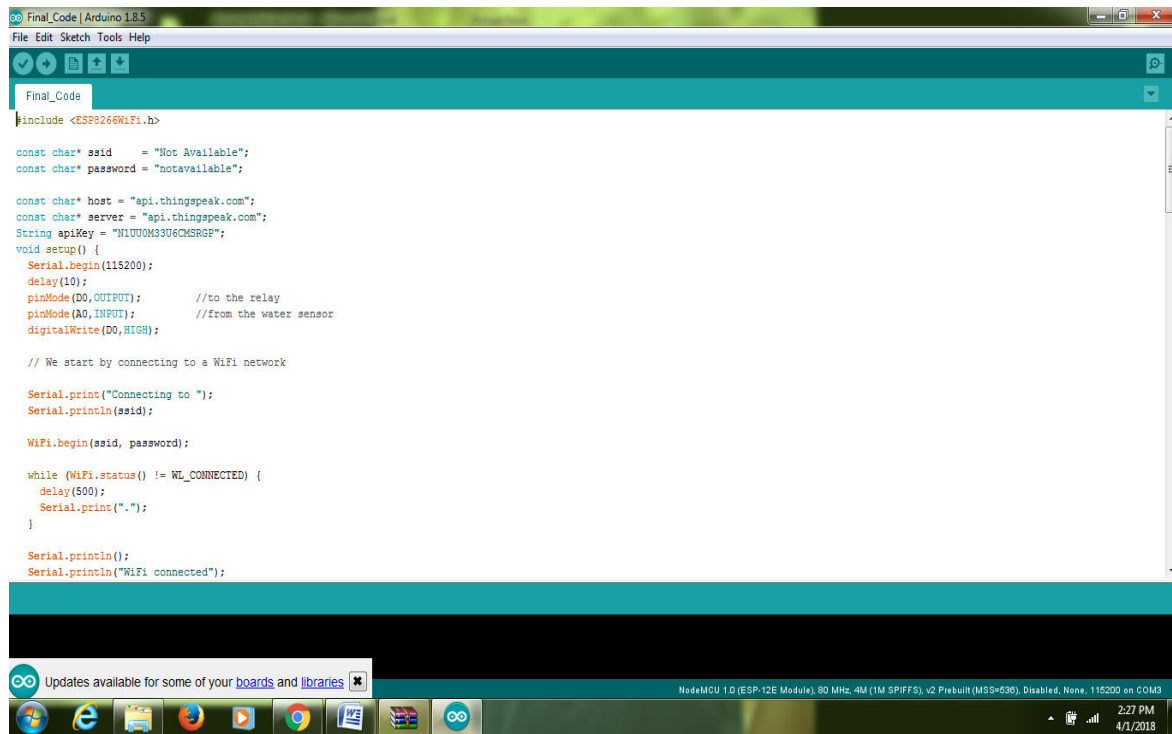
S.NO	TEST CASE	INPUT	EXPECTED OUTPUT	ACTUAL OUTPUT
1.	Connect to the Wi-Fi	Enter Wi-Fi username, password	Successfully connected to internet	Successfully connected to internet
2.	Turn OFF motor if the sensor value < threshold value	Sensor Value < threshold value	Turn OFF motor and send alert to user	Turn OFF motor and send alert to user
3.	Turn OFF motor if the sensor value < threshold value	Sensor Value > threshold value	Motor keeps Running	Motor keeps Running
4.	Connect to the Thingspeak cloud	Enter cloud username, password	Successfully connected to cloud	Successfully connected to cloud
5.	Upload to the cloud if the sensor value < threshold value	Check the sensor value with threshold value	Uploaded to the cloud successfully	Uploaded to the cloud successfully

TABLE 6.4.1 Test Cases

CHAPTER-7

SCREEN SHOTS

AUTOMATED WATER MOTOR



The screenshot shows the Arduino IDE interface with the following code:

```
Final_Code | Arduino 1.8.5
File Edit Sketch Tools Help

Final_Code

#include <ESP8266WiFi.h>

const char* ssid = "Not Available";
const char* password = "notavailable";

const char* host = "api.thingspeak.com";
const char* server = "api.thingspeak.com";
String apiKey = "H1U0M33U6CMSRGP";

void setup() {
  Serial.begin(115200);
  delay(10);
  pinMode(D0, OUTPUT); //to the relay
  pinMode(A0, INPUT); //from the water sensor
  digitalWrite(D0, HIGH);

  // We start by connecting to a WiFi network

  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

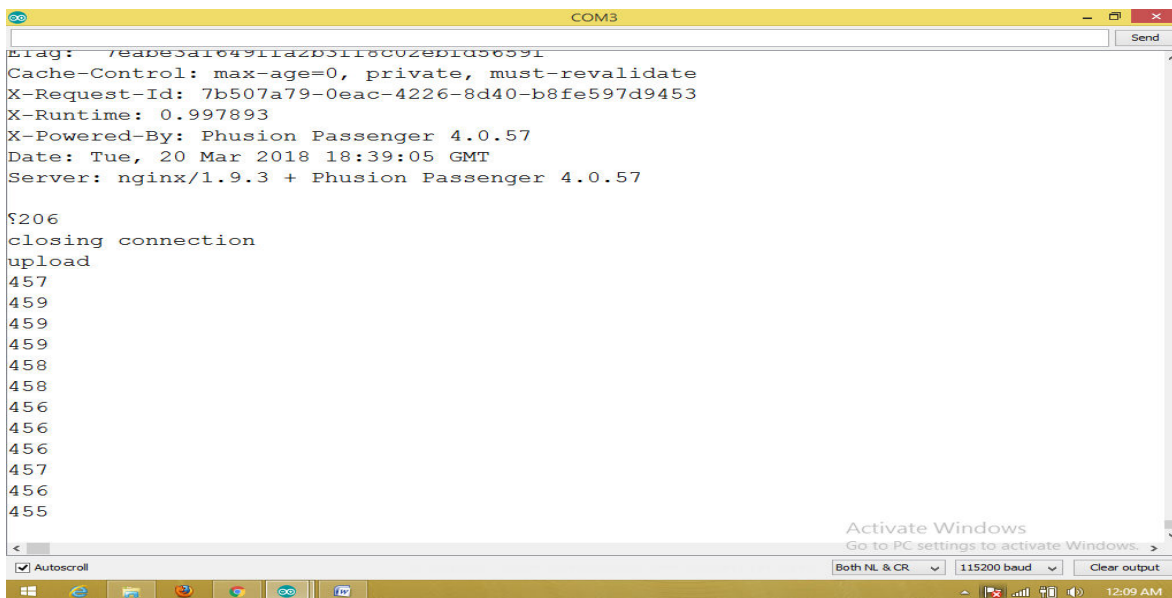
  Serial.println();
  Serial.println("WiFi connected");
}
```

Updates available for some of your boards and libraries

NodeMCU 1.0 (ESP-12E Module), 80 MHz, 4M (1M SPIFFS), v2 Prebuilt (M5S+C30), Disabled, None, 115200 on COM3

2:27 PM 4/1/2018

FIG 7.1 Program code in Arduino IDE



The screenshot shows the COM3 serial monitor with the following output:

```
COM3

tag: 7eade3a164911a2b5118c02eb1d56591
Cache-Control: max-age=0, private, must-revalidate
X-Request-Id: 7b507a79-0eac-4226-8d40-b8fe597d9453
X-Runtime: 0.997893
X-Powered-By: Phusion Passenger 4.0.57
Date: Tue, 20 Mar 2018 18:39:05 GMT
Server: nginx/1.9.3 + Phusion Passenger 4.0.57

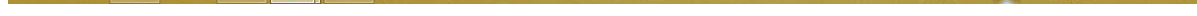
$206
closing connection
upload
457
459
459
459
458
456
456
456
457
456
455
```

Activate Windows
Go to PC settings to activate Windows.

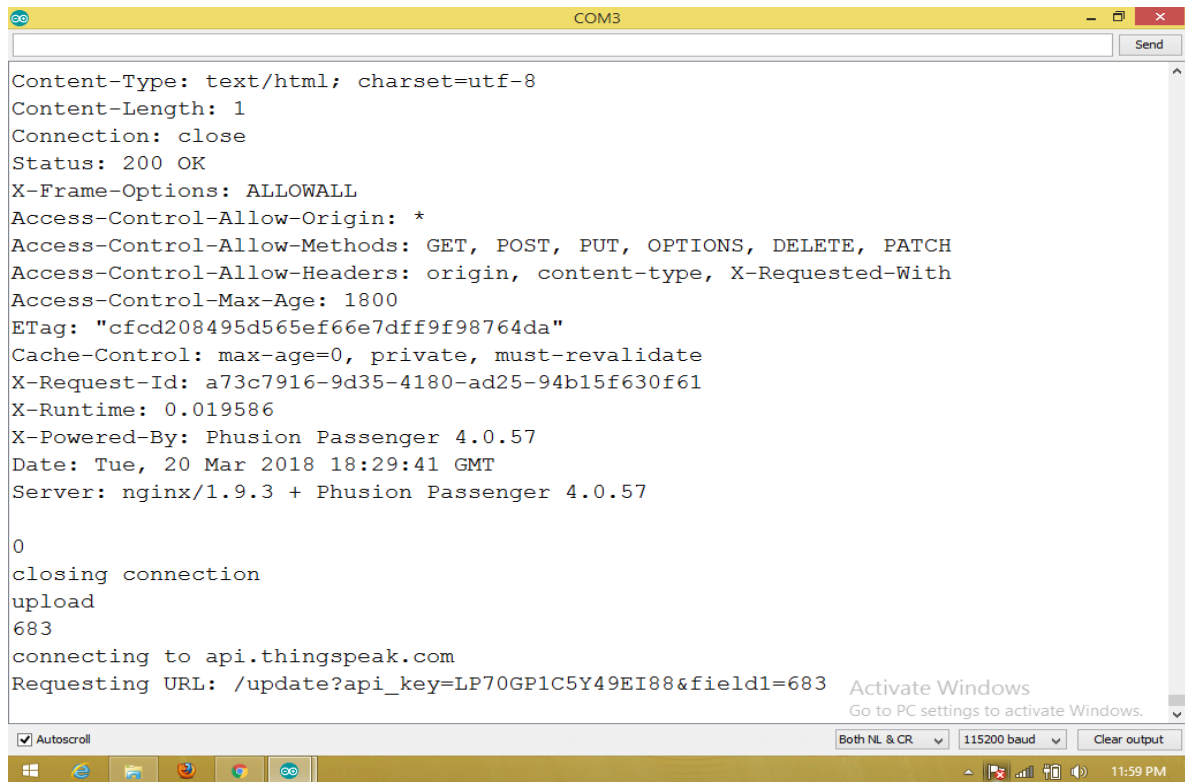
Autoscroll Both NL & CR 115200 baud Clear output

12:09 AM

FIG 7.2 Uploading code to NodeMCU



AUTOMATED WATER MOTOR



```
COM3
Content-Type: text/html; charset=utf-8
Content-Length: 1
Connection: close
Status: 200 OK
X-Frame-Options: ALLOWALL
Access-Control-Allow-Origin: *
Access-Control-Allow-Methods: GET, POST, PUT, OPTIONS, DELETE, PATCH
Access-Control-Allow-Headers: origin, content-type, X-Requested-With
Access-Control-Max-Age: 1800
ETag: "cfcd208495d565ef66e7dff9f98764da"
Cache-Control: max-age=0, private, must-revalidate
X-Request-Id: a73c7916-9d35-4180-ad25-94b15f630f61
X-Runtime: 0.019586
X-Powered-By: Phusion Passenger 4.0.57
Date: Tue, 20 Mar 2018 18:29:41 GMT
Server: nginx/1.9.3 + Phusion Passenger 4.0.57

0
closing connection
upload
683
connecting to api.thingspeak.com
Requesting URL: /update?api_key=LP70GP1C5Y49EI88&field1=683
Activate Windows
Go to PC settings to activate Windows.
```

FIG 7.5 Uploading data to cloud



FIG 7.6 NodeMCU in operation (motor off)

AUTOMATED WATER MOTOR

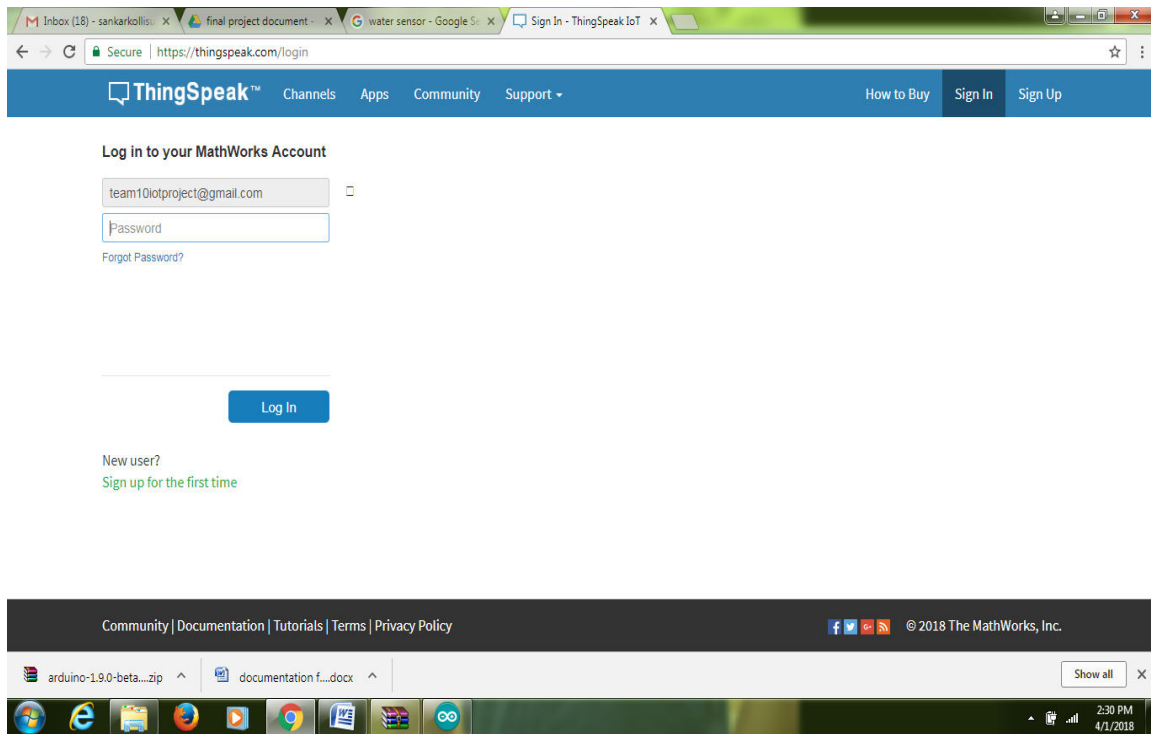


FIG 7.7 ThingSpeak login page

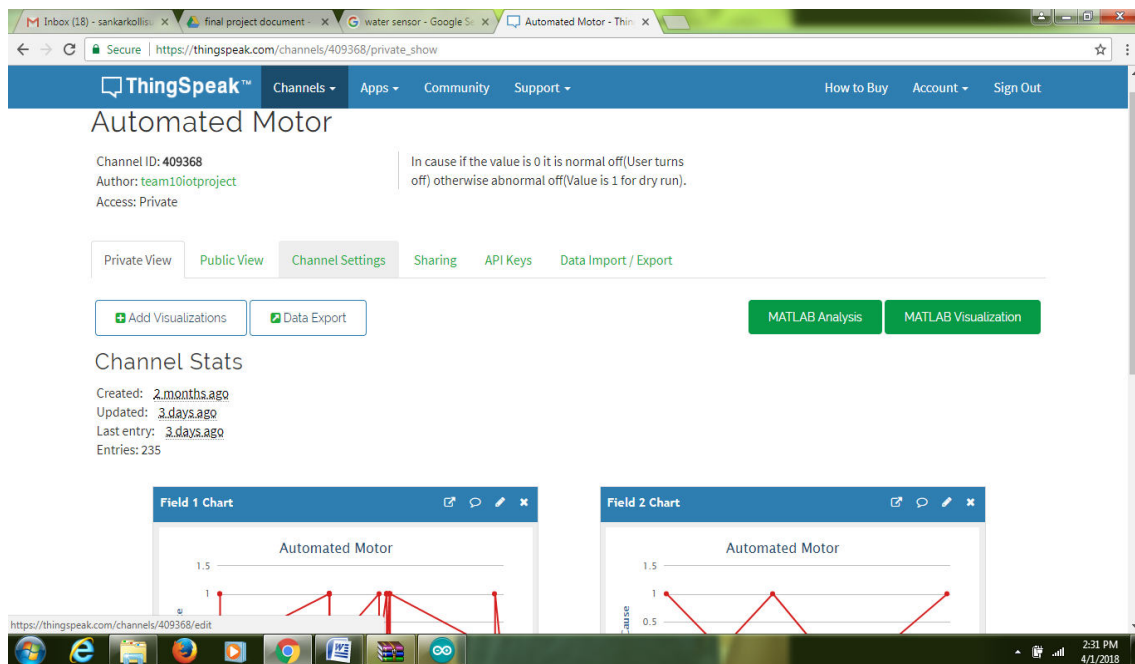


FIG 7.8 ThingSpeak channel view

AUTOMATED WATER MOTOR

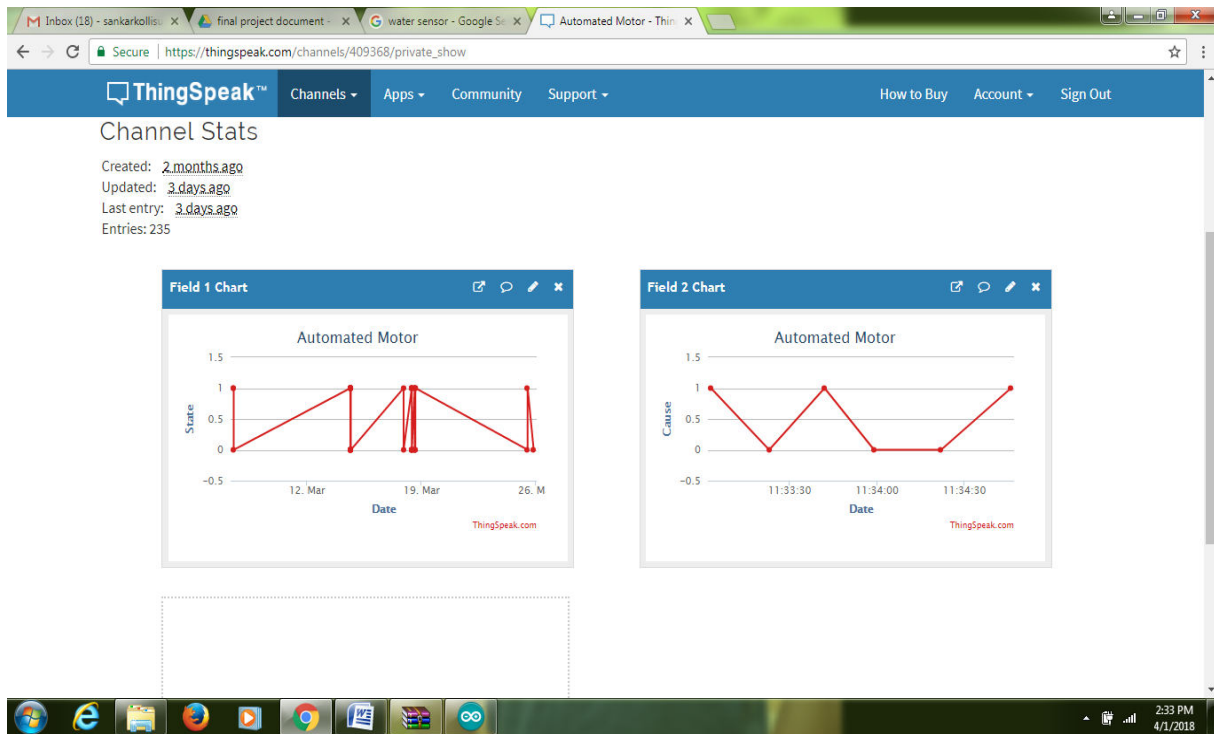


FIG 7.9 Thingspeak fields view

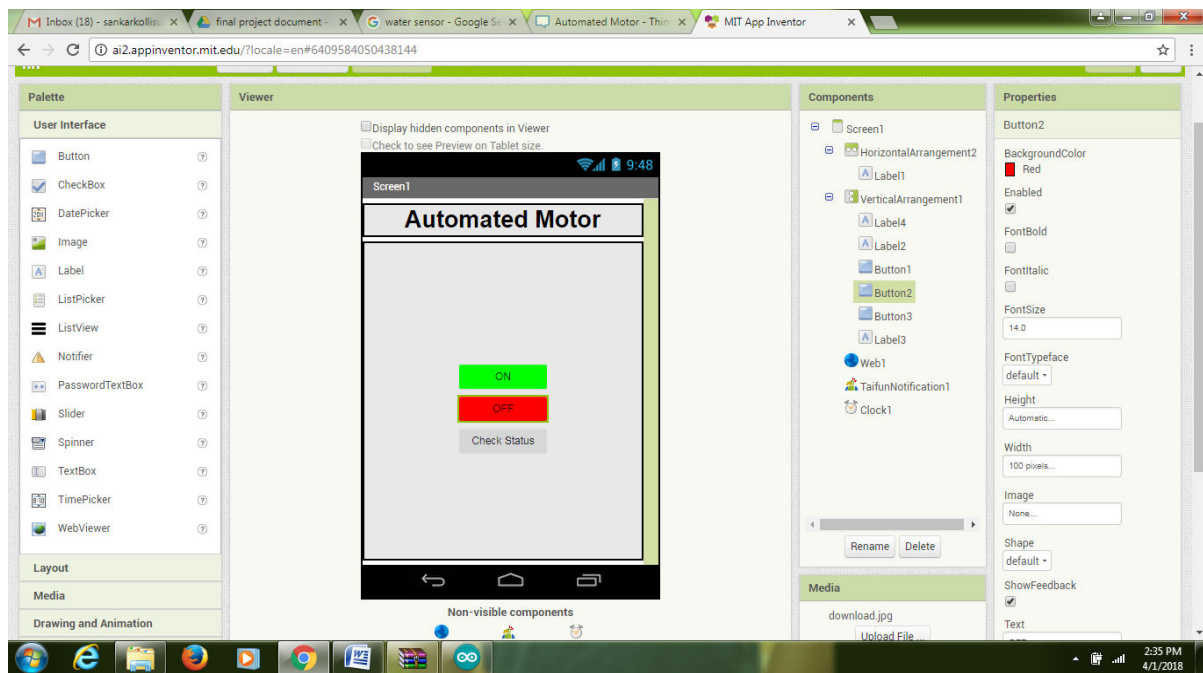


FIG 7.10 MIT App Inventor home

CHAPTER-8

CONCLUSION

8 CONCLUSION

Traditional method of using the motor causes a lot of burden to the farmers but using this approach we can remotely on and off the motor and the most interesting thing is it protects motor from dry run. The following are the advantages of this proposed system. The advantages are:-

- Keeps motor safe
- Saves electricity
- Notifies user about lack of water
- Automates water pumping in agriculture
- Remote ON and OFF

REFERENCES

9 REFERENCES

- Brown, Eric (13 September 2016). "Who Needs the Internet of Things?". Linux.com. Retrieved 23 October 2016.
- ^ :^a ^b Brown, Eric (20 September 2016). "21 Open Source Projects for IoT". Linux.com. Retrieved 23 October 2016.
- Blog post by ioBridge President of Software, Hans Scharler, retrieved 21 Mar 2016
- ^ ThingSpeak.com Terms of Service, retrieved 21 Mar 2016
- Hardesty, Larry (August 19, 2010). "The MIT roots of Google's new software". MIT News Office.
- ^ ^a ^b "On the Shoulders of Giants!". Google. Retrieved August 10, 2010.
- ^ Wolber, David; Abelson, Hal; Spertus, Ellen; Looney, Liz (May 2011), App Inventor for Android: Create Your Own Android Apps, O'Reilly, ISBN 978-1-4493-9748-7
- Tibbets, Linden. "ifttt the beginning..." IFTTT blog. Retrieved October 16, 2014.
- Information Technology — Programming languages, their environments and system software interfaces — Extensions for the programming language C to support embedded processors, ISO/IEC JTC1 SC22 WG14 N1021, Date: 2003-09-24, Reference number of document: ISO/IEC DTR 18037

APPENDIX

APPENDIX

EXECUTION STEPS

Step 1: Connect the Power supply to the Controller.

Step 2: Initialize the Arduino, Node MCU, Sensor Board.

Step 3: Arrange the connections as per block Diagram.

Step 4: Open the Arduino IDE Software and Write the Source Code , Define the pins.

Step 5: Interface the Sensor Connections According to the Pins defined in the Source code

Step 6: Load the Program into Arduino. if any error is present rectify it and Save. again load.

Step 7: Now go to Serial Monitor and Observe the output of sensor .

Step8: Upload the data into Cloud through Node MCU.

Step9: IFTTT sends email notification to the email after the data uploaded to the cloud.

AUTOMATED WATER MOTOR

STEPS TO UPLOAD THE DATA TO CLOUD

- Login to the cloud by entering the username and password.
- Create the channels in the cloud.
- Each channel has 8 fields. In each field the data is stored in the graph format.
- Take the private write API key of the particular field where we want to store the value.
- API key is used in the program to upload the value by using following URL:-

```
https://api.thingspeak.com/update?api_key=LP70GP1C5Y49EI88&field1=0
```

- Now the values are uploaded to the cloud successfully.