**Project Title**

**A PROJECT REPORT**

***Submitted***

*In the partial fulfillment of the requirements for the Mid-Term Project Assignment Evaluation of*

**BACHELOR OF TECHNOLOGY**

in

**COMPUTER SCIENCE AND ENGINEERING**

by

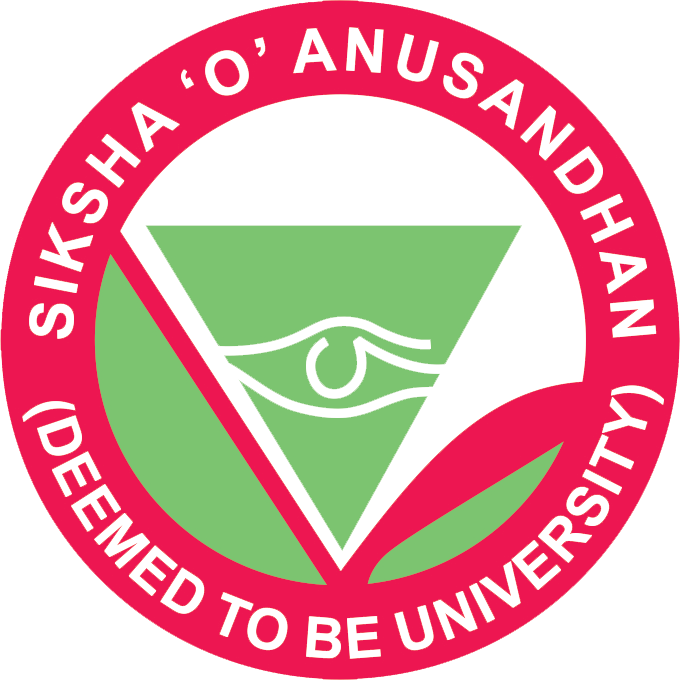
**P.SAITEJA** [13311A04I8]

**P.SAIAKHIL** [13311A04I9]

**UNDER THE GUIDANCE OF**

**Dr. Biswaranjan Swain Mrs. Subhaluxmi Sahoo**

**Associate Professor Assistant Professor**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**INSTITUTE OF TECHNICAL EDUCATION AND RESEARCH**

**SIKSHA ‘O’ ANUSANDHAN (DEEMED TO BE UNIVERSITY)**

**BHUBANESWAR-751003, ODISHA, INDIA.**

**DECEMBER-2022**

**DECLARATION**

I hereby declare that the work described in this thesis “**Project Title**” which is being submitted in partial fulfillment for the award of **Bachelor of Technology** in the Department of **Computer Science and Engineering** affiliated to Siksha ‘O’ Anusandhan (Deemed to be University), Bhubaneswar (Odisha) is the result of investigations carried out by me under the Guidance of **Dr. Biswaranjan Swain, Associate Professor and** of **Mrs. Subhaluxmi Sahoo, Assistance Professor, Centre for IoT**, **Institute of Technical Education and Research (ITER), Bhubaneswar.**

.

The work is original and has not been submitted for any Degree/Diploma of this or any other university.

**Place:** Bhubaneswr

**Date:**

**P. SAITEJA** [13311A04I8]

**P. SAIAKHIL** [13311A04I9]

**CERTIFICATE**

This is to certify that the project report entitled “**Talking Plant**” being submitted by **P. SAITEJA (13311A04I8)** and **P. SAIAKHIL (13311A04I9)** in partial fulfillment for Mid-Term Project Assignment Evaluation of **Bachelor of Technology in** the Department of **Computer Science and Engineering** affiliated to Siksha ‘O’ Anusandhan (Deemed to be University), Bhubaneswar (Odisha)**,** is a record of bonafide work carried out by them during the academic year 2022-2023 under our guidance and supervision.

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

**(Dr. Biswaranjan Swain) (Ms.Subhaluxmi Sahoo)**

**Project Guide Project Guide**

**ACKNOWLEDGEMENTS**

I would like to express my sincere gratitude to Dr. Debahuti Mishra, Professor, Head of the department, Computer Science and Engineering, Institute of Technical Education and Research (ITER), Siksha ‘O’ Anusandhan (Deemed to be University), Bhubaneswar for his continued support and valuable guidance and encouragement extended to me during my project work. I thank her for his painstaking efforts to guide me throughout my work.

I thank Dr. Biswaranjan Swain, Associate Professor of Center for Internet of Things (CIoT), Institute of Technical Education and Research (ITER), Siksha ‘O’ Anusandhan (Deemed to be University), Bhubaneswar for his valuable advice, support and help during my project work.

I thank Mrs. Subhaluxmi Sahoo, Assistance Professor of Center for Internet of Things (CIoT), Institute of Technical Education and Research (ITER), Siksha ‘O’ Anusandhan (Deemed to be University), Bhubaneswar for her valuable comments and suggestions that greatly helped in improving quality of report.

I thank Mr. Pradeep Ku. Bebrata, Mr. Bipin Ku. Karan, Ms. Mamata Sutar, Mrs Shradhanjali Nayak, Mr. Upendra Kumar Prusty, Institute of Technical Education and Research (ITER), Siksha ‘O’ Anusandhan (Deemed to be University), Bhubaneswar for their valuable comments and suggestions that greatly helped during circuit analysis and housing of the project.

I thank to all my teachers and professors for their valuable comments after reviewing my project reports.

I wish to extend my special thanks to all my colleagues and friends who helped directly or indirectly to complete my project assignment work.

I extend my thanks to my parents and all my family members for their unceasing encouragement and support who wished me a lot to complete this work.

**Signature of Students**

**ABSTRACT**

The main purpose of this project is to take care of the plants that we grow in our Houses. the main aspect in this project is that, plant talks to us about its health status. Health Status is monitored through Some sensors connected to NODEMCU; moisture sensor is used to measure the moisture of the soil that is required for the plant. Temperature and Humidity sensor is used to measure the ambient weather parameter that suits the plant health condition. whenever the plant is under healthy environment it will show a message" i am happy" On OLED Display as well as sends a notification to mobile app. If there are any abnormalities in the measured parameters it will show that "i am sad. Plant daily greets his owner and will notify about the main events like birthdays of owners friends. Another important aspect of this project is we can manually water the plant from anywhere through Mobile app communicated through WIFI communication.

**INDEX**

**TOPICS**

* **Declaration ⅰ**
* **Certificates ⅰi**
* **Acknowledgement ⅱi**
* **Abstract iv**

**CHAPTER 1: INTRODUCTION**

* 1. **Introduction of the project ………………………………………………1**
  2. **Project overview…………………………………………………… 2**

**CHAPTER 2: INTERNET OF THINGS**

**2.1 Introduction to IOT……………………………………………………… 4**

**2.2 Need of Internet of Things……………………………………………… 7**

**2.3 Applications of Internet of Things……………………………………… 10**

**CHAPTER 3: HARDWARE DESCRIPTION**

**3.1 Introduction with block diagram………………………………………… 13**

**3.2 NodeMCU………………………………………………………………… 16**

**3.3 Moisture Sensor…………………………………………………………… 18**

**3.4 Servo Motor …………………………………………………………………21**

**3.5 DHT11……………………………………………………….……….. 23**

**3.6 OLED Display…………………………………………………………….. 25**

**CHAPTER 4: SOFTWARE DESCRIPTION**

**4.1 Arduino IDE…………………………………………………………………28**

**4.3 Procedural steps for compilation, simulation and dumping……………………**

**CHAPTER 5: Project Description……………………………………………………………**

**CHAPTER 6: Advantages, Disadvantages and Applications…………………………**

**CHAPTER 7: Results, Conclusion, Future Prospects…………………………**

**CHAPTER 8: Team Work……………………………………………………**

**CHAPTER 1: INTRODUCTION**

**1.1 Introduction:**

A new generation of “Talking Plant” is, designed to increase water-delivery efficiency, often at reasonable cost.

The main purpose of this project is to take care of the plants that we grow in our Houses. the main aspect in this project is that, plant talks to us about its health status. Health Status is monitored through Some sensors connected to NODEMCU , moisture sensor is used to measure the moisture of the soil that is required for the plant. Temperature and Humidity sensor is used to measure the ambient weather parameter that suits the plant health condition.

This includes sophisticated controllers that read real-time site conditions and deliver just enough water to keep plants healthy. Sensors that deliver water based on the amount of moisture in the soil.

The most sophisticated of the "smart" systems are the so-called talking plant, these take in real-time weather data, captured on-site through weather stations from wunderground.

WIFI Technology is being used everywhere in our daily life to fulfill our requirements. We are employing different sensors for different applications sometimes we may even use same sensors differently for different applications. Whatever it may be the final output is life has increased its speed with the technology boosters. One of the ideal ways of using technology is to employ it to sense the condition of soil and weather parameters so that efficient water deliver systems can be developed .

The modules in the project are: Moisture sensor which continuously gives the information about the condition of soil (WET or DRY), humidity sensor servo is used to monitor temperature and humidity . Servo meter is used as a sprinkler and based on the sensor values this servo meter rotates . OLED display where the values of temperature and humidity are displayed which are retrieved from the wunderground .

The controlling device of the whole system is a Node MCU (ESP8266-12E). the data from the sensors is posted in weather station that we have created in wunderground. the real time values from the station ,we have created are displayed on the OLED .All this transmission is based on WIFI technology. In achieving the task the Node MCU is loaded with a program written using Embedded ‘C’ language.

**The main objectives of the project are:**

1. The soil condition and weather parameters are known wirelessly trough WiFi.
2. Maintain the health status of the plant.
3. Displaying the status of plant on OLED and also knowning the status through application.
4. Controlling the plant health and growth through the application.

5.Sending messages in the form of text to the people in the house which is displayed using OLED and android application.

**1.2 Project Overview:**

The project " **TALKING PLANT** **"** using Node MCU.

The project explains the implementation of "**TALKING PLANT"** using Node MCU The organization of the thesis is explained here with:

**Chapter 1** Presents introduction to the overall thesis and the overview of the project. In the project overview a brief introduction of Node MCU, Humidity sensor, Soil Moisture , and its applications are discussed.

**Chapter 2**  Presents the topic Internet Of Things. It explains about what is Internet Of Things, Need for Internet Of Things and its applications.

**Chapter 3**  Presents the hardware description. It deals with the block diagram of the project and explains the purpose of each block. In the same chapter the explanation of

**Chapter 4** Presents the software description. It explains the implementation of the project using Arduino IDE.

**Chapter 5** Presents the project description along with

**Chapter 6** Presents the advantages, disadvantages and applications of the project.

**Chapter 7** Presents the results, conclusion and future scope of the project.

**CHAPTER 2: INTERNET OF THINGS (IOT)**

**Introduction:**

The Internet of Things (IoT) is an important topic in technology industry, policy, and engineering circles and has become headline news in both the specialty press and the popular media. This technology is embodied in a wide spectrum of networked products, systems, and sensors, which take advantage of advancements in computing power, electronics miniaturization, and network interconnections to offer new capabilities not previously possible. An abundance of conferences, reports, and news articles discuss and debate the prospective impact of the “IoT revolution”—from new market opportunities and business models to concerns about security, privacy, and technical interoperability. It is basically connecting the electronic items to the cloud and in other words internet in order to make them more useful and worthful.

The large-scale implementation of IoT devices promises to transform many aspects of the way we live. For consumers, new IoT products like Internet-enabled appliances, home automation components,there are many uses to household people and energy management devices are moving us toward a vision of the “smart home’’, offering more security and energy efficiency. Other personal IoT devices like wearable fitness and health monitoring devices and network enabled medical devices are transforming the way healthcare services are delivered. This technology promises to be beneficial for people with disabilities and the elderly, enabling improved levels of independence and quality of life at a reasonable cost. IoT systems like networked vehicles, intelligent traffic systems, and sensors embedded in roads and bridges move us closer to the idea of “smart cities’’, which help minimize congestion and energy consumption. IoT technology offers the possibility to transform agriculture, industry, and energy production and distribution by increasing the availability of information along the value chain of production using networked sensors. There are lots of sensors in this electronic world using iot we can make them more worthful than ever.However, IoT raises many issues and challenges that need to be considered and addressed in order for potential benefits to be realized.

A number of companies and research organizations have offered a wide range of projections about the potential impact of IoT on the Internet and the economy during the next five to ten years. Cisco, for example, projects more than 24 billion Internet–connected objects by 2019; Morgan Stanley, however, projects 75 billion networked devices by 2020. Looking out further and raising the stakes higher, Huawei forecasts 100 billion IoT connections by 2025. McKinsey Global Institute suggests that the financial impact of IoT on the global economy may be as much as $3.9 to $11.1 trillion by 2025. While the variability in predictions makes any specific number questionable, collectively they paint a picture of significant growth and influence.There are enormous number of things we could see in future.

**History:**

The term “Internet of Things” (IoT) was first used in 1999 by British technology pioneer Kevin Ashton to describe a system in which objects in the physical world could be connected to the Internet by sensors. Ashton coined the term to illustrate the power of connecting Radio-Frequency Identification (RFID) tags used in corporate supply chains to the Internet in order to count and track goods without the need for human intervention. Today, the Internet of Things has become a popular term for describing scenarios in which Internet connectivity and computing capability extend to a variety of objects, devices, sensors, and everyday items.

While the term “Internet of Things” is relatively new, the concept of combining computers and networks to monitor and control devices has been around for decades. By the late 1970s, for example, systems for remotely monitoring meters on the electrical grid via telephone lines were already in commercial use. In the 1990s, advances in wireless technology allowed “machine–to–machine” (M2M) enterprise and industrial solutions for equipment monitoring and operation to become widespread. Many of these early M2M solutions, however, were based on closed purpose–built networks and proprietary or industry–specific standards, rather than on Internet Protocol (IP)–based networks and Internet standards.

Using IP to connect devices other than computers to the Internet is not a new idea. The first Internet “device”—an IP–enabled toaster that could be turned on and off over the Internet—was featured at an Internet conference in 1990. Over the next several years, other “things” were IP–enabled, including a soda machine at Carnegie Mellon University in the US and a coffee pot in the Trojan Room at the University of Cambridge in the UK (which remained Internet–connected until 2001). From these whimsical beginnings, a robust field of research and development into “smart object networking” helped create the foundation for today’s Internet of Things.

From a broad perspective, the confluence of several technology and market trends is making it possible to interconnect more and smaller devices cheaply and easily:

• **Ubiquitous Connectivity**—Low–cost, high–speed, pervasive network connectivity, especially through licensed and unlicensed wireless services and technology, makes almost everything “connectable’’.

• **Widespread adoption of IP–based networking**— IP has become the dominant global standard for networking, providing a well–defined and widely implemented platform of software and tools that can be incorporated into a broad range of devices easily and inexpensively.

• **Computing Economics**— Driven by industry investment in research, development, and manufacturing, Moore’s law continues to deliver greater computing power at lower price points and lower power consumption.

• **Miniaturization**— Manufacturing advances allow cutting-edge computing and communications technology to be incorporated into very small objects. Coupled with greater computing economics, this has fueled the advancement of small and inexpensive sensor devices, which drive many IoT applications.

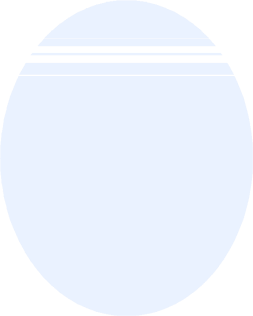
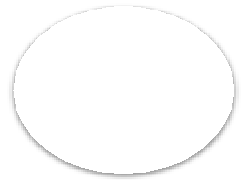
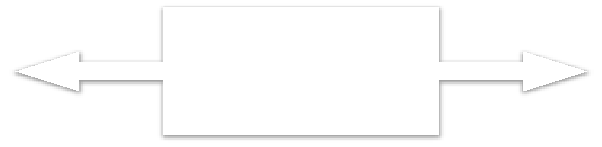
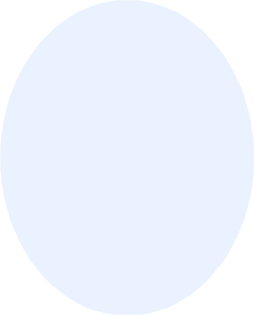
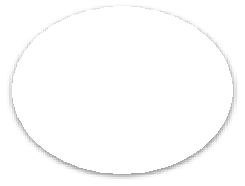
• **Advances in Data Analytics**— New algorithms and rapid increases in computing power, data storage, and cloud services enable the aggregation, correlation, and analysis of vast quantities of data; these large and dynamic datasets provide new opportunities for extracting information and knowledge.

• **Rise of Cloud Computing**– Cloud computing, which leverages remote, networked computing resources to process, manage, and store data, allows small and distributed devices to interact with powerful back-end analytic and control capabilities.

From this perspective, the IoT represents the convergence of a variety of computing and connectivity trends that have been evolving for many decades. At present, a wide range of industry sectors – including automotive, healthcare, manufacturing, home and consumer electronics, and well beyond are considering the potential for incorporating IoT technology into their products, services, and operations.

**Internet of Things Communications Models:**

The device-to-device communication model represents two or more devices that directly connect and communicate between one another, rather than through an intermediary application server. These devices communicate over many types of networks, including IP networks or the Internet. Often, however these devices use protocols like Bluetooth, Z-Wave, or ZigBee to establish direct device-to-device communications, as shown in Figure 1.



Light

Bulb

Wireless

Network

Light

Switch

Manufacturer A

**Bluetooth, Z+Wave, ZigBee**

Manufacturer B

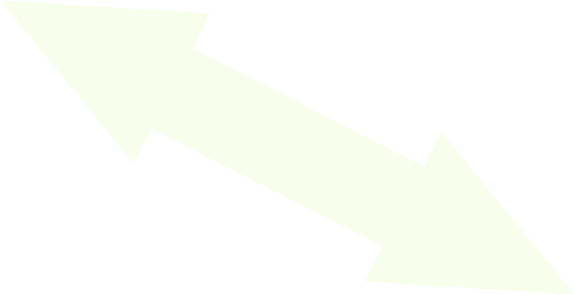
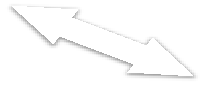
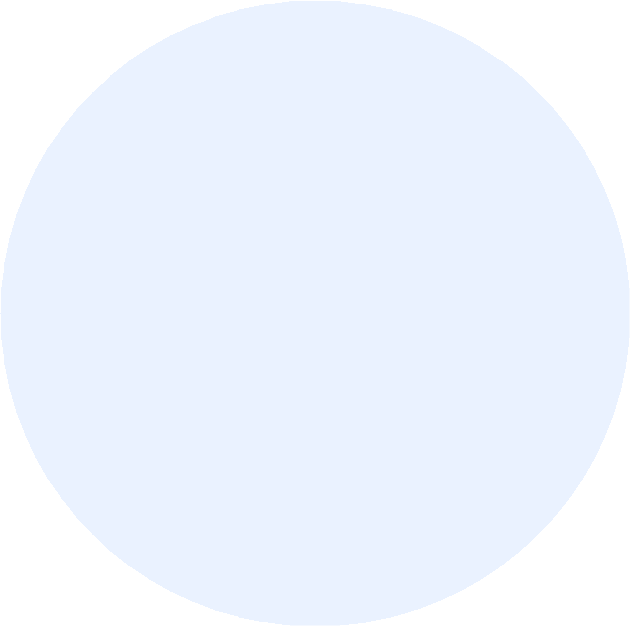
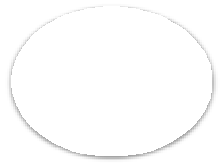
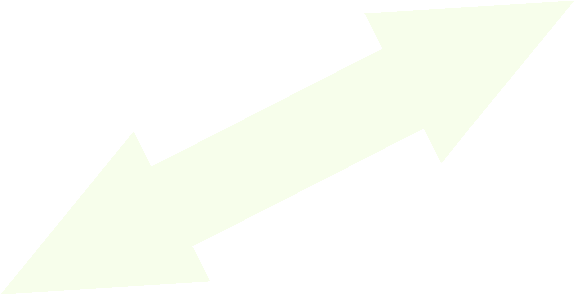
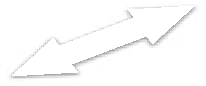
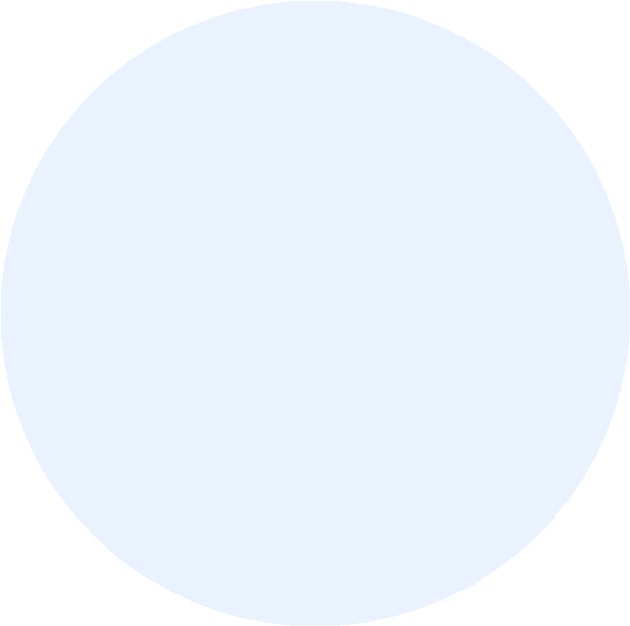
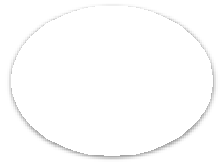
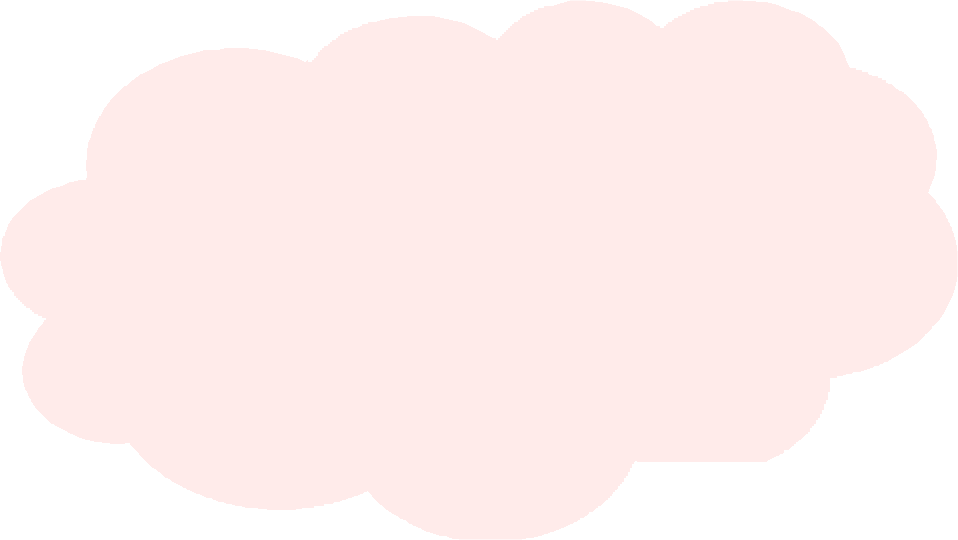
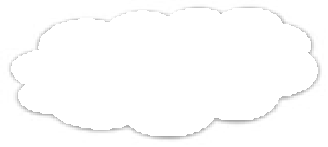
Source: Tschofenig, H., et. al., Architectural Considera/ons in Smart Object Networking. Tech. no. RFC 7452. Internet Architecture Board, Mar. 2015. Web.

<hRps://www.rfc&editor.org/rfc/rfc7452.txt>.

Figure 1. Example of device-to-device communication model.

**Device-to-Cloud Communications:**

In a device-to-cloud communication model, the IoT device connects directly to an Internet cloud service like an application service provider to exchange data and control message traffic. This approach frequently takes advantage of existing communications mechanisms like traditional wired Ethernet or Wi-Fi connections to establish a connection between the device and the IP network, which ultimately connects to the cloud service. This is shown in Figure 2.



**HTTP**

**TLS TCP IP**

Application Service

Provider

**CoAP**

**DTLS UDP IP**

Device with

Device with Carbon

Temperature Monoxide

Sensor Sensor

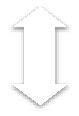
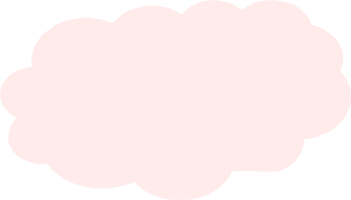
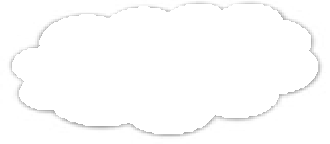
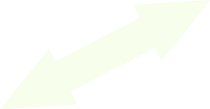
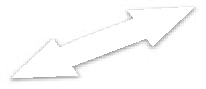
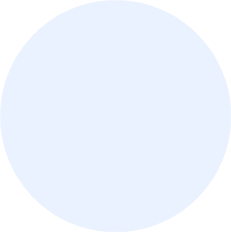
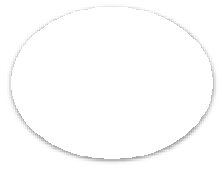
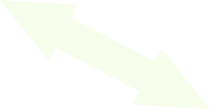
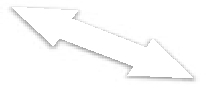
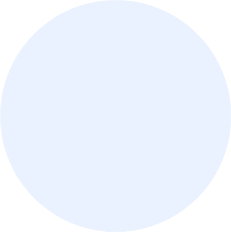
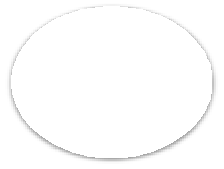
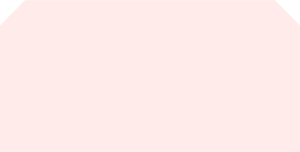
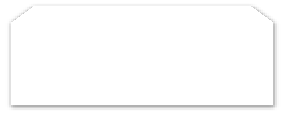
Source: Tschofenig, H., et. al., Architectural Considera9ons in Smart Object Networking. Tech. no. RFC 7452. Internet Architecture Board, Mar. 2015. Web.

<hN[ps://www.rfcQeditor.org/rfc/rfc7452.txt>.](http://www.rfcQeditor.org/rfc/rfc7452.txt)

Figure 2. Device-to-cloud communication model diagram.

**Device-to-Gateway Model:**

In the device-to-gateway model, or more typically, the device-to-application-layer gateway (ALG) model, the IoT device connects through an ALG service as a conduit to reach a cloud service. In simpler terms, this means that there is application software operating on a local gateway device, which acts as an intermediary between the device and the cloud service and provides security and other functionality such as data or protocol translation. The model is shown in Figure 3



Application Service

Provider

**IPv4/IPv6**

**Protocol**

**Stack**

**HTTP**

**TLS TCP IPv6**

Local Gateway

**CoAP**

**DTLS UDP IPv6**

Device with

Temperature Sensor

**Layer 1 Protocol**

**Bluetooth Smart IEEE 802.11 (WiDFi)**

**IEEE 802.15.4 (LRDWPAN)**

Device with

Carbon Monoxide Sensor

Source: Tschofenig, H., et. al., Architectural Considera9ons in Smart Object Networking. Tech. no. RFC 7452. Internet Architecture Board, Mar. 2015. Web.

<hN[ps://www.rfcQeditor.org/rfc/rfc7452.txt>.](http://www.rfcQeditor.org/rfc/rfc7452.txt)

Figure 3. Device-to-gateway communication model diagram.

**Back-End Data-Sharing Model:**

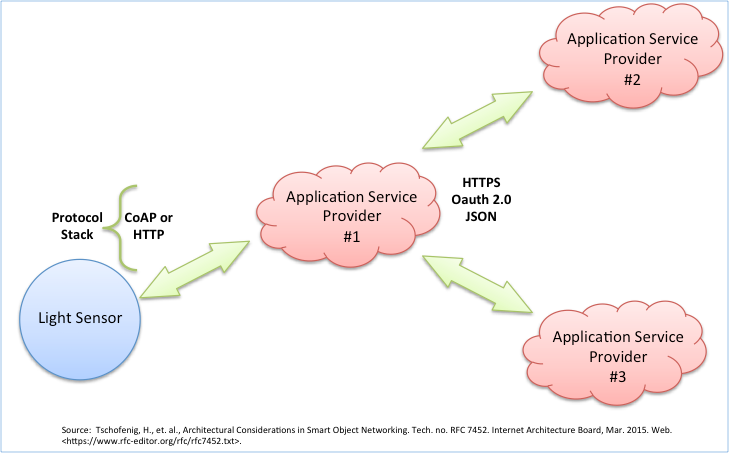
****

Figure 4. Back-end data sharing model diagram.

The back-end data-sharing model refers to a communication architecture that enables users to export and analyze smart object data from a cloud service in combination with data from other sources. This architecture supports “the [user’s] desire for granting access to the uploaded sensor data to third parties”. This approach is an extension of the single device-to-cloud communication model, which can lead to data silos where “IoT devices upload data only to a single application service provider’’. A back-end sharing architecture allows the data collected from single IoT device data streams to be aggregated and analyzed.

For example, a corporate user in charge of an office complex would be interested in consolidating and analyzing the energy consumption and utilities data produced by all the IoT sensors and Internet-enabled utility systems on the premises. Often in the single device-to-cloud model, the data each IoT sensor or system produces sits in a stand-alone data silo. An effective back-end data sharing architecture would allow the company to easily access and analyze the data in the cloud produced by the whole spectrum of devices in the building. Also, this kind of architecture facilitates data portability needs. Effective back-end data- sharing architectures allow users to move their data when they switch between IoT services, breaking down traditional data silo barriers.

The back-end data-sharing model suggests a federated cloud services approach or cloud applications programmer interfaces (APIs) are needed to achieve interoperability of smart device data hosted in the cloud. A graphical representation of this design is shown in Figure 4.

**Different Definitions, Similar Concepts:**

Despite the global buzz around the Internet of Things, there is no single, universally accepted definition for the term. Different definitions are used by various groups to describe or promote a particular view of what IoT means and its most important attributes. Some definitions specify the concept of the Internet or the Internet Protocol (IP), while others, perhaps surprisingly, do not. For example, consider the following definitions.

The Internet Architecture Board (IAB) begins RFC 7452, “Architectural Considerations in Smart Object Networking’’, with this description:

*The term "Internet of Things" (IoT) denotes a trend where a large number of embedded devices employ communication services offered by the Internet protocols. Many of these devices, often called "smart objects,’’ are not directly operated by humans, but exist as components in buildings or vehicles, or are spread out in the environment.*

Within the Internet Engineering Task Force (IETF), the term *“smart object networking”* is commonly used in reference to the Internet of Things. In this context, “smart objects” are devices that typically have significant constraints, such as limited power, memory, and processing resources, or bandwidth. Work in the IETF is organized around specific requirements to achieve network interoperability between several types of smart objects.

Published in 2012, the International Telecommunication Union (ITU) ITU–T Recommendation Y.2060, Overview of the Internet of things, discusses the concept of interconnectivity, but does not specifically tie the IoT to the Internet:

*3.2.2 Internet of things (IoT): A global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies*.

*Note 1—Through the exploitation of identification, data capture, processing and communication capabilities, the IoT makes full use of things to offer services to all kinds of applications, whilst ensuring that security and privacy requirements are fulfilled.*

*Note 2—From a broader perspective, the IoT can be perceived as a vision with technological and societal implications.*

This definition in a call for papers for a feature topic issue of IEEE Communications Magazine links the IoT back to cloud services:

*The Internet of Things (IoT) is a framework in which all things have a representation and a presence in the Internet. More specifically, the Internet of Things aims at offering new applications and services bridging the physical and virtual worlds, in which Machine-to-Machine (M2M) communications represents the baseline communication that enables the interactions between Things and applications in the cloud.*

The Oxford Dictionaries offers a concise definition that invokes the Internet as an element of the IoT:

*Internet of things (noun): The interconnection via the Internet of computing devices embedded in everyday objects, enabling them to send and receive data.*

**IoT Applications and Scenarios of Relevance:**

“The major objectives for IoT are the creation of smart environments/spaces and self-aware things (for example: smart trans- port, products, cities, buildings, rural areas, energy, health, living, etc.) for climate, food, energy, mobility, digital society and health applications”.

The outlook for the future is the emerging of a network of interconnected uniquely identifiable objects and their virtual representations in an Internet alike structure that is positioned over a network of interconnected computers allowing for the creation of a new platform for economic growth.

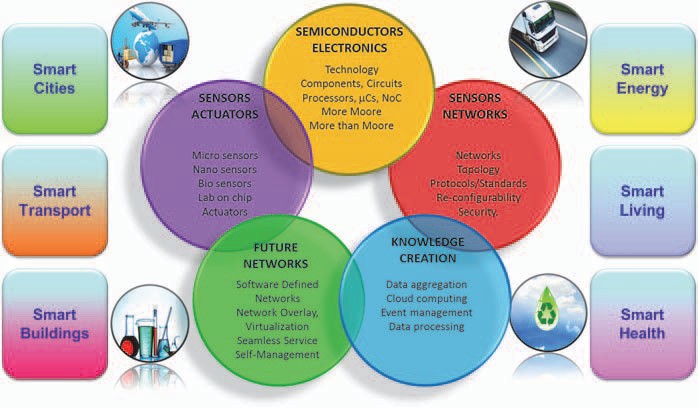


Fig. 5 Internet of Things — smart environments and smart spaces creation.

At the city level, the integration of technology and quicker data analysis will lead to a more coordinated and effective civil response to security and safety (law enforcement and blue light services); higher demand for outsourcing security capabilities.

At the building level, security technology will be integrated into systems and deliver a return on investment to the end-user through leveraging the technology in multiple applications (HR and time and attendance, customer behaviour in retail applications etc.).

There will be an increase in the development of “Smart” vehicles which have low (and possibly zero) emissions. They will also be connected to infrastructure. Additionally, auto manufacturers will adopt more use of “Smart” materials.

**IoT Functional View**

The Internet of Things concept refers to uniquely identifiable things with their virtual representations in an Internet-like structure and IoT solutions comprising a number of components such as:

* Module for interaction with local IoT devices (for example embedded in a mobile phone or located in the immediate vicinity of the user and thus contactable via a short range wireless inter- face). This module is responsible for acquisition of observations and their forwarding to remote servers for analysis and permanent storage.
* Module for local analysis and processing of observations acquired by IoT devices.
* Module for interaction with remote IoT devices, directly over the Internet or more likely via a proxy. This module is responsible for acquisition of observations and their forwarding to remote servers for analysis and permanent storage.
* Module for application specific data analysis and processing. This module is running on an application server serving all clients. It is taking requests from mobile and web clients and relevant IoT observations as input, executes appropriate data processing algorithms and generates output in terms of knowledge that is later presented to users.
* Module for integration of IoT-generated information into the business processes of an enterprise. This module will be gaining importance with the increased use of IoT data by enterprises as one of the important factors in day-to-day business or business strategy definition.
* User interface (web or mobile): visual representation of measurements in a given context (for example on a map) and interaction with the user, i.e. definition of user queries.
* It is important to highlight that one of the crucial factors for the success of IoT is stepping away from vertically-oriented, closed systems towards open systems, based on open APIs and standardized protocols at various system levels.

**Data Management:**

Data management is a crucial aspect in the Internet of Things. When considering a world of objects interconnected and constantly exchanging all types of information, the volume of the generated data and the processes involved in the handling of those data become critical.

A long-term opportunity for wireless communications chip makers is the rise of Machine-to-Machine (M2M) computing, which one of the enabling technologies for Internet of Things. This technology spans abroad range of applications. While there is consensus that M2M is a promising pocket of growth, analyst estimates on the size of the opportunity diverge by a factor of four . Conservative estimates assume roughly 80 million to 90 million M2M units will be sold in 2014, whereas more optimistic projections forecast sales of 300 million units. Based on historical analyses of adoption curves for similar disruptive technologies, such as portable MP3 players and antilock braking systems for cars, it is believed that unit sales in M2M could rise by as much as a factor of ten over the next five years.

There are many technologies and factors involved in the “data management” within the IoT context. Some of the most relevant concepts which enable us to understand the challenges and opportunities of data management are:

• Data Collection and Analysis

• Big Data

• Semantic Sensor Networking

• Virtual Sensors

• Complex Event Processing.

**Data Collection and Analysis (DCA):**

Data Collection and Analysis modules or capabilities are the essential components of any IoT platform or system, and they are constantly evolving in order to support more features and provide more capacity to external components (either higher layer applications leveraging on the data stored by the DCA module or other external systems exchanging information for analysis or processing).

The DCA module is part of the core layer of any IoT platform. Some of the main functions of a DCA module are:

**User/customer data storing:** Provides storage of the customer’s information collected by sensors

**User data & operation modelling:** Allows the customer to create new sensor data models to accommodate collected information and the modelling of the supported operations

**On demand data access:** Provides APIs to access the collected data

**Device event publish/subscribe/forwarding/notification:** Provides APIs to access the collected data in real time conditions

**Customer rules/filtering:** Allows the customer to establish its own filters and rules to correlate events

**Customer task automation:** Provides the customer with the ability to manage his automatic processes.

Example: scheduled platform originated data collection,…

**Customer workflows:** Allows the customer to create his own workflow to process the incoming eventsfrom a device

**Multitenant structure:** Provides the structure to support multiple organizations and reseller schemes.

In the coming years, the main research efforts should be targeted to some features that should be included in any Data Collection and Analysis platform:

• **Multi-protocol.** DCA platforms should be capable of handling or understanding different input (and output) protocols and formats. Different standards and wrappings for the submission of observations should be supported.

• **De-centralization.** Sensors and measurements/observations captured by them should be stored in systems that can be de-centralized from a single platform. It is essential that different components, geographically distributed in different locations may cooperate and exchange data. Related with this concept, federation among different systems will make possible the global integration of IoT architectures.

• **Security.** DCA platforms should increase the level of data protection and security, from the transmission of messages from devices (sensors, actuators, etc.) to the data stored in the platform.

• **Data mining features.** Ideally, DCA systems should also integrate capacities for the processing of the stored info, making it easier to extract useful data from the huge amount of contents that may be recorded.

**CHAPTER 3: HARDWARE DESCRIPTION**

**3.1 Introduction**:

In this chapter the block diagram of the project and design aspect of independent modules are considered. Block diagram is shown in fig: 3.1

Servo Motor

Cloud

Moisture Sensor

Humidity

NodeMCU

OLED Display

Android app

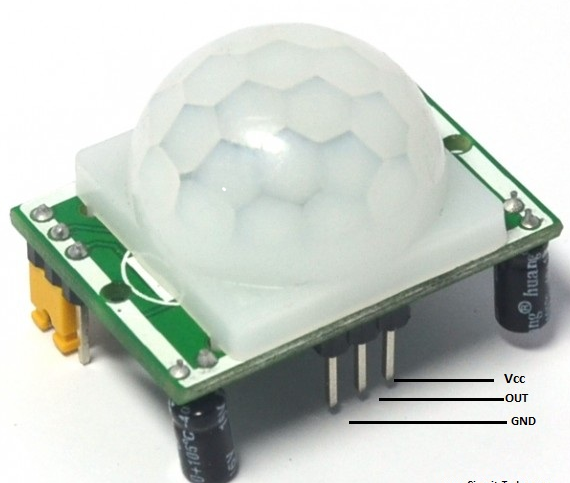
**FIG 3.1: Block diagram of TALKING PLANT**

**The main blocks of this project are:**

* Node MCU
* Moisture sensor
* DHT sensor
* servo Motor
* OLED display

**PIR SENSOR:**

PIR Sensor (#555-28027) The PIR (Passive Infra-Red) Sensor is a pyroelectric device that detects motion by sensing changes in the infrared (radiant heat) levels emitted by surrounding objects. This motion can be detected by checking for a sudden change in the surrounding IR pattern. When motion is detected the PIR sensor outputs a high signal on its output pin. This logic signal can be read by a microcontroller or used to drive an external load; see the source current limits in the features list below.

**Features**: 

Detect a person up to approximately 30 ft away, or up to 15 ft away in reduced sensitivity mode

Jumper selects normal operation or reduced sensitivity Source current up to 12 mA @ 3 V, 23 mA @ 5 V

Onboard LEDs light up the lens for fast visual feedback when movement is detected Mounting holes for #2 sized screws

3-pin SIP header ready for breadboard or through-hole projects Small size makes it easy to conceal

Easy interface to any microcontroller

**Key Specifications**:

Power Requirements: 3 to 6 VDC; 130 µA idle, 3 mA active (no load)

Communication: Single bit high/low output

Operating temperature: 32 to 122 °F (0 to 50 °C)

Dimensions: 1.41 x 1.0 x 0.8 in (35.8 x 25.4 x 20.3 cm)

**Theory of Operation:**

Pyroelectric devices, such as the PIR sensor, have elements made of a crystalline material that generates an electric charge when exposed to infrared energy. The changes in the amount of infrared energy striking the element change the voltages generated, which are measured by an on-board amplifier. The device contains a Fresnel lens, which focuses the infrared signals onto the element. As the ambient infrared signals change rapidly, the on-board amplifier trips the output to indicate motion. The onboard jumper allows the user to select between normal operation and reduced sensitivity. The sensitivity of the PIR Sensor varies with temperature and other environmental conditions. Generally, when in reduced sensitivity mode, the PIR sensor will detect an object at up to half the distance it would in normal operating mode

**3.2 ESP8266-12E:**

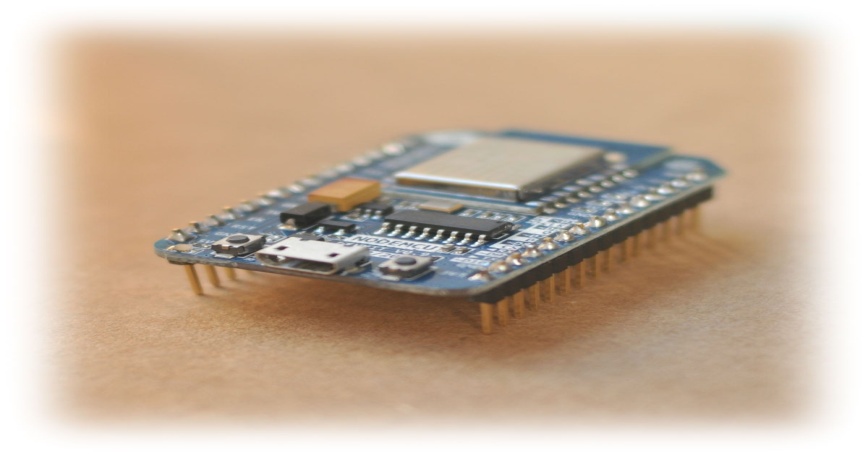
****

Fig. 8 ESP8266-12E

Your ESP8266 is an impressive, low cost WiFi module suitable for adding WiFi functionality to an existing microcontroller project via a UART serial connection. The module can even be reprogrammed to act as a standalone WiFi connected device–just add power! The feature list is impressive and includes: 802.11 b/g/n protocol Wi-Fi Direct (P2P), soft-AP Integrated TCP/IP protocol stack.

**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

**Board:**

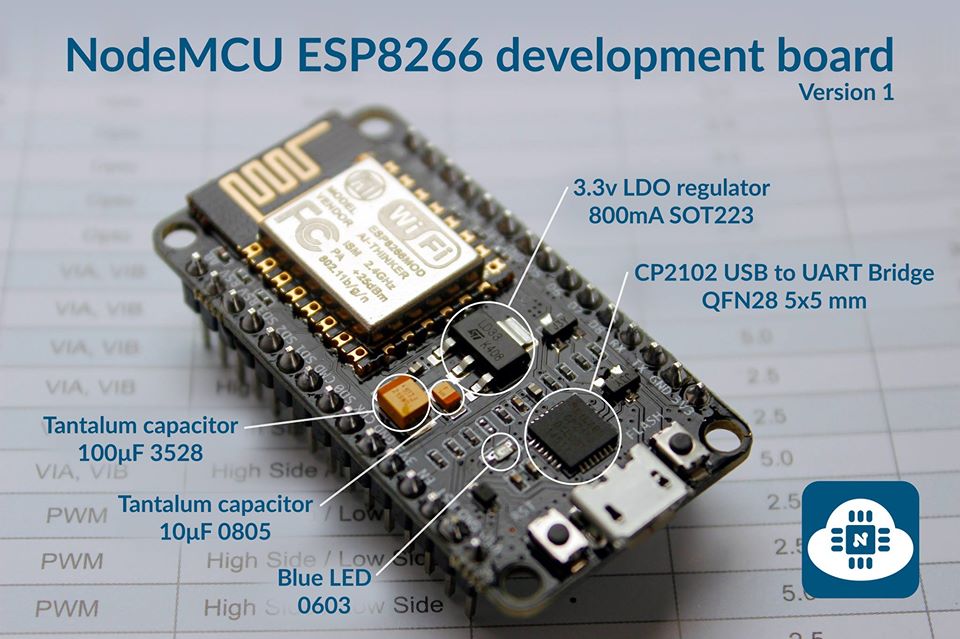
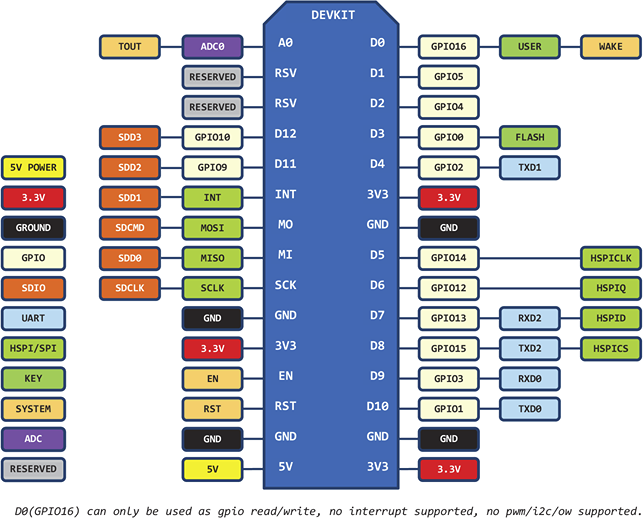
****

Fig. 8 ESP8266-12E board Description

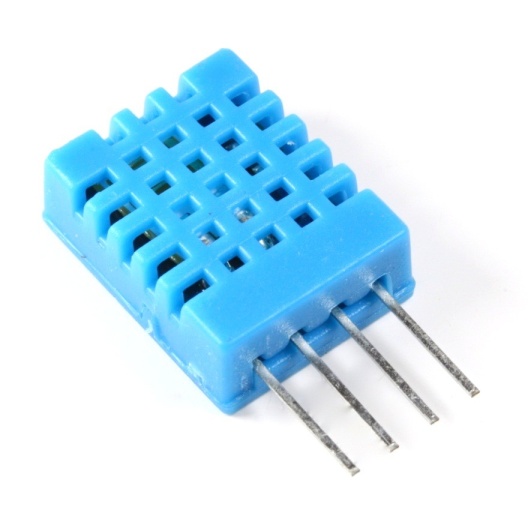
ESP8266EX has been designed for mobile, wearable electronics and Internet of Things applications with the aim of achieving the lowest power consumption with a combination of several proprietary techniques. The power saving architecture operates mainly in 3 modes: active mode, sleep mode and deep sleep mode. By using advance power management techniques and logic to power-down functions not required and to control switching between sleep and active modes, ESP8266EX consumes about than 60uA in deep sleep mode (with RTC clock still running) and less than 1.0mA (DTIM=3) or less than 0.5mA (DTIM=10) to stay connected to the access point. When in sleep mode, only the calibrated real-time clock and watchdog remains active. The real-time clock can be programmed to wake up the ESP8266EX at any required interval. The ESP8266EX can be programmed to wake up when a specified condition is detected. This minimal wake-up time feature of the ESP8266EX can be utilized by mobile device SOCs, allowing them to remain in the low-power standby mode until WiFi is needed. In order to satisfy the power demand of mobile and wearable electronics, ESP8266EX can be programmed to reduce the output power of the PA to fit various application profiles, by trading off range for power consumption.

**Pin Definition:**

****

**DHT11:(Humidity Sensor)**

DHT11 digital temperature and humidity sensor is a composite Sensor contains a calibrated digital signal output of the temperature and humidity. Application of a dedicated digital modules collection technology and the temperature and humidity sensing technology, to ensure that the product has high reliability and excellent long-term stability. The sensor includes a resistive sense of wet components and an NTC temperature measurement devices, and connected with a high-performance 8-bit microcontroller.



**Applications :**

HVAC, dehumidifier, testing and inspection equipment, consumer goods, automotive, automatic control, data loggers, weather stations, home appliances, humidity regulator, medical and other humidity measurement and control.

**Features:**

Low cost, long-term stability, relative humidity and temperature measurement, excellent quality, fast response, strong anti-interference ability, long distance signal transmission, digital signal output, and precise calibration.

**Product parameters**:

Relative humidity Resolution : 16Bit

Repeatability : ±1% RH

Accuracy : At 25 º C ±5% RH

Interchangeability : fully interchangeable

Response time : 1 / e (63%) of 25 º C 6s 1m / s air 6s

Temperature Resolution : 16Bit

Repeatability : ±0.2

Range : At 25º C±2 º C

Response time : 1 / e (63%) 10S

**Electrical Characteristics**

Power supply : DC 3.5～5.5V

Supply Current : measurement 0.3mA standby 60μ A

Sampling period : more than 2 seconds

**Pin Description**

1, the VDD power supply 3.5～5.5V DC

2 DATA serial data, a single bus

3, NC, empty pin

4, GND ground, the negative power

**Attentions of application**

**(1) Operating conditions**

Applying the DHT11 sensor beyond its working range stated in this datasheet can result in 3%RH signal shift/discrepancy. The DHT11 sensor can recover to the calibrated status gradually when it gets back to the normal operating condition and works within its range. Please refer to (3) of this section to accelerate its recovery. Please be aware that operating the DHT11 sensor in the non-normal working conditions will accelerate sensor’s aging process.

**(2) Attention to chemical materials**

Vapor from chemical materials may interfere with DHT’s sensitive-elements and debase its sensitivity. A high degree of chemical contamination can permanently damage the sensor.

**(3) Restoration process when (1) & (2) happen**

Step one: Keep the DHT sensor at the condition of Temperature 50~60Celsius, humidity <10%RH for 2 hours;

Step two:K keep the DHT sensor at the condition of Temperature 20~30Celsius, humidity >70%RH for 5 hours.

**(4) Temperature Affect**

Relative humidity largely depends on temperature. Although temperature compensation technology is used to ensure accurate measurement of RH, it is still strongly advised to keep the humidity and temperature sensors working under the same temperature. DHT11 should be mounted at the place as far as possible from parts that may generate heat.

**(5) Light Affect**

Long time exposure to strong sunlight and ultraviolet may debase DHT’s performance.

**(6) Connection wires**

The quality of connection wires will affect the quality and distance of communication and high quality shielding-wire is recommended.

**(7) Other attentions**

\* Welding temperature should be bellow 260Celsius and contact should take less than 10 seconds.

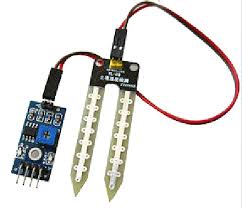
\* Avoid using the sensor under dew condition.

\* Do not use this product in safety or emergency stop devices or any other occasion that failure of DHT11 may cause personal injury.

\* Storage: Keep the sensor at temperature 10-40℃, humidity <60%RH.

**Soil Moisture Sensor:**

This sensor can be used to test the moisture of soil, when the soil is having water shortage, the module output is at high level, else the output is at low level. By using this sensor one can automatically water the flower plant, or any other plants requiring automatic watering technique. Module triple output mode, digital output is simple, analog output more accurate, serial output with exact readings.



**Features:**

* Sensitivity adjustable.
* Has fixed bolt hole, convenient installation.
* Threshold level can be configured.
* Module triple output mode, digital output is simple, analog output more accurate, serial output with exact readings.

**Applications:**

* Agriculture
* Landscape irrigation

**Specifications:**

* Operating Voltage +5v dc regulated
* Soil moisture Digital value is indicated by out pin

**Servo Meter:**

Tiny and lightweight with high output power. Servo can rotate approximately 180 degrees (90 in each direction), and works just like the standard kinds but smaller. You can use any servo code, hardware or library to control these servos. Good for beginners who want to make stuff move without building a motor controller with nfeedback & gear box, especially since it will fit in small places. It comes with a 3 horns (arms) and hardware.

****

**Specifications:**

* Weight: 9 g
* Dimension: 22.2 x 11.8 x 31 mm approx.
* Stall torque: 1.8 kgf·cm
* Operating speed: 0.1 s/60 degree
* Operating voltage: 4.8 V (~5V)
* Dead band width: 10 µs
* Temperature range: 0 ºC – 55 ºC

**OLED display:**

Transparent OLED is a breakthrough transparent display technology that displays dynamic or interactive information on a transparent surface glass. This revolutionary display allows users to view what is shown on a glass video screen while still being able to see through it. Designers can overlay text, digital images, and video content onto physical objects or scenes that sit behind the glass.

Transparent OLED displays are self-emitting and utilize cutting-edge Organic Light Emitting Diode (OLED) technology to eliminate the need for a backlight or enclosure, making it possible to create truly see-through installations in a virtually frameless glass design.



**Working principle of OLED:**

OLEDs work in a similar way to conventional diodes and LEDs, but instead of using layers of n-type and p-type semiconductors, they use organic molecules to produce their electrons and holes. A simple OLED is made up of six different layers. On the top and bottom there are layers of protective glass or plastic. The top layer is called the seal and the bottom layer the substrate. In between those layers, there's a negative terminal (sometimes called the cathode) and a positive terminal (called the anode). Finally, in between the anode and cathode are two layers made from organic molecules called the emissive layer (where the light is produced, which is next to the cathode) and the conductive layer (next to the anode)

**CHAPTER 4: SOFTWARE DESCRIPTION**

**Arduino IDE:**

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

### **Writing Sketches**

Programs written using Arduino Software (IDE) are called **sketches**. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom righthand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

**NB: Versions of the Arduino Software (IDE) prior to 1.0 saved sketches with the extension .pde. It is possible to open these files with version 1.0, you will be prompted to save the sketch with the .ino extension on save.**

|  |  |
| --- | --- |
| https://www.arduino.cc/en/uploads/Guide/play.png | Verify  Checks your code for errors compiling it. |
| https://www.arduino.cc/en/uploads/Guide/export.png | Upload  Compiles your code and uploads it to the configured board. See [uploading](https://www.arduino.cc/en/Guide/Environment" \l "uploading) below for details.  Note: If you are using an external programmer with your board, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer" |
| https://www.arduino.cc/en/uploads/Guide/new.png | New  Creates a new sketch. |
| https://www.arduino.cc/en/uploads/Guide/open.png | Open  Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.  Note: due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the **File | Sketchbook**menu instead. |
| https://www.arduino.cc/en/uploads/Guide/save.png | Save  Saves your sketch. |
| https://www.arduino.cc/en/uploads/Guide/serial_monitor.png | SerialMonitor  Opens the [serial monitor](https://www.arduino.cc/en/Guide/Environment" \l "serialmonitor). |

Additional commands are found within the five menus: **File**, **Edit**, **Sketch**, **Tools**, **Help**. The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.

#### File:

* New  
  Creates a new instance of the editor, with the bare minimum structure of a sketch already in place.
* Open  
  Allows to load a sketch file browsing through the computer drives and folders.
* OpenRecent   
  Provides a short list of the most recent sketches, ready to be opened.
* Sketchbook   
  Shows the current sketches within the sketchbook folder structure; clicking on any name opens the corresponding sketch in a new editor instance.
* Examples  
  Any example provided by the Arduino Software (IDE) or library shows up in this menu item. All the examples are structured in a tree that allows easy access by topic or library.
* Close  
  Closes the instance of the Arduino Software from which it is clicked.
* Save  
  Saves the sketch with the current name. If the file hasn't been named before, a name will be provided in a "Save as.." window.
* Saveas...   
  Allows to save the current sketch with a different name.
* PageSetup  
  It shows the Page Setup window for printing.
* Print  
  Sends the current sketch to the printer according to the settings defined in Page Setup.
* Preferences  
  Opens the Preferences window where some settings of the IDE may be customized, as the language of the IDE interface.
* Quit  
  Closes all IDE windows. The same sketches open when Quit was chosen will be automatically reopened the next time you start the IDE.

#### Edit:

* Undo/Redo   
  Goes back of one or more steps you did while editing; when you go back, you may go forward with Redo.
* Cut  
  Removes the selected text from the editor and places it into the clipboard.
* Copy  
  Duplicates the selected text in the editor and places it into the clipboard.
* Copy for ForumCopies the code of your sketch to the clipboard in a form suitable for posting to the forum, complete with syntax coloring.
* Copy as HTML   
  Copies the code of your sketch to the clipboard as HTML, suitable for embedding in web pages.
* Paste  
  Puts the contents of the clipboard at the cursor position, in the editor.
* SelectAll   
  Selects and highlights the whole content of the editor.
* Comment/Uncomment   
  Puts or removes the // comment marker at the beginning of each selected line.
* Increase/DecreaseIndent  
  Adds or subtracts a space at the beginning of each selected line, moving the text one space on the right or eliminating a space at the beginning.
* Find  
  Opens the Find and Replace window where you can specify text to search inside the current sketch according to several options.
* Find Next   
  Highlights the next occurrence - if any - of the string specified as the search item in the Find window, relative to the cursor position.
* Find Previous  
  Highlights the previous occurrence - if any - of the string specified as the search item in the Find window relative to the cursor position.

#### Sketch:

* Verify/Compile   
  Checks your sketch for errors compiling it; it will report memory usage for code and variables in the console area.
* Upload  
  Compiles and loads the binary file onto the configured board through the configured Port.
* Upload Using Programmer   
  This will overwrite the bootloader on the board; you will need to use Tools > Burn Bootloader to restore it and be able to Upload to USB serial port again. However, it allows you to use the full capacity of the Flash memory for your sketch. Please note that this command will NOT burn the fuses. To do so a Tools -> Burn Bootloader command must be executed.
* Export Compiled Binary   
  Saves a .hex file that may be kept as archive or sent to the board using other tools.
* Show Sketch Folder   
  Opens the current sketch folder.
* IncludeLibrary   
  Adds a library to your sketch by inserting #include statements at the start of your code. For more details, see[libraries](https://www.arduino.cc/en/Guide/Environment" \l "libraries) below. Additionally, from this menu item you can access the Library Manager and import new libraries from .zip files.
* AddFile...   
  Adds a source file to the sketch (it will be copied from its current location). The new file appears in a new tab in the sketch window. Files can be removed from the sketch using the tab menu accessible clicking on the small triangle icon below the serial monitor one on the right side o the toolbar.

#### 

#### Tools:

* Auto Format   
  This formats your code nicely: i.e. indents it so that opening and closing curly braces line up, and that the statements inside curly braces are indented more.
* Archive Sketch   
  Archives a copy of the current sketch in .zip format. The archive is placed in the same directory as the sketch.
* Fix Encoding & Reload   
  Fixes possible discrepancies between the editor char map encoding and other operating systems char maps.
* Serial Monitor   
  Opens the serial monitor window and initiates the exchange of data with any connected board on the currently selected Port. This usually resets the board, if the board supports Reset over serial port opening.
* Board  
  Select the board that you're using. See below for [descriptions of the various boards](https://www.arduino.cc/en/Guide/Environment" \l "boards).
* Port  
  This menu contains all the serial devices (real or virtual) on your machine. It should automatically refresh every time you open the top-level tools menu.
* Programmer  
  For selecting a harware programmer when programming a board or chip and not using the onboard USB-serial connection. Normally you won't need this, but if you're [burning a bootloader](https://www.arduino.cc/en/Tutorial/Bootloader) to a new microcontroller, you will use this.
* Burn Bootloader   
  The items in this menu allow you to burn a [bootloader](https://www.arduino.cc/en/Hacking/Bootloader) onto the microcontroller on an Arduino board. This is not required for normal use of an Arduino or Genuino board but is useful if you purchase a new ATmega microcontroller (which normally come without a bootloader). Ensure that you've selected the correct board from the **Boards** menu before burning the bootloader on the target board. This command also set the right fuses.

### 

### **Sketchbook**

The Arduino Software (IDE) uses the concept of a sketchbook: a standard place to store your programs (or sketches). The sketches in your sketchbook can be opened from the **File > Sketchbook** menu or from the **Open** button on the toolbar. The first time you run the Arduino software, it will automatically create a directory for your sketchbook. You can view or change the location of the sketchbook location from with the **Preferences** dialog.

**Beginning with version 1.0, files are saved with a .ino file extension. Previous versions use the .pde extension. You may still open .pde named files in version 1.0 and later, the software will automatically rename the extension to .ino**.

### **Tabs, Multiple Files, and Compilation**

Allows you to manage sketches with more than one file (each of which appears in its own tab). These can be normal Arduino code files (no visible extension), C files (.c extension), C++ files (.cpp), or header files (.h).

### **Uploading**

Before uploading your sketch, you need to select the correct items from the **Tools > Board** and **Tools > Port** menus. The [boards](https://www.arduino.cc/en/Guide/Environment" \l "boards) are described below. On the Mac, the serial port is probably something like **/dev/tty.usbmodem241** (for an Uno or Mega2560 or Leonardo) or **/dev/tty.usbserial-1B1** (for a Duemilanove or earlier USB board), or**/dev/**tty.USA19QW1b1P1**.1** (for a serial board connected with a Keyspan USB-to-Serial adapter). On Windows, it's probably COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to find out, you look for USB serial device in the ports section of the Windows Device Manager. On Linux, it should be **/dev/ttyACMx** ,**/dev/ttyUSBx** or similar. Once you've selected the correct serial port and board, press the upload button in the toolbar or select the **Upload** item from the **File** menu. Current Arduino boards will reset automatically and begin the upload. With older boards (pre-Diecimila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino Software (IDE) will display a message when the upload is complete, or show an error.

When you upload a sketch, you're using the Arduino **boot-loader**, a small program that has been loaded on to the microcontroller on your board. It allows you to upload code without using any additional hardware. The boot-loader is active for a few seconds when the board resets; then it starts whichever sketch was most recently uploaded to the microcontroller. The boot-loader will blink the on-board (pin 13) LED when it starts (i.e. when the board resets).

### **Libraries**

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from the **Sketch > Import Library** menu. This will insert one or more **#include** statements at the top of the sketch and compile the library with your sketch. Because libraries are uploaded to the board with your sketch, they increase the amount of space it takes up. If a sketch no longer needs a library, simply delete its **#include** statements from the top of your code. Some libraries are included with the Arduino software. Others can be downloaded from a variety of sources or through the Library Manager. Starting with version 1.0.5 of the IDE, you do can import a library from a zip file and use it in an open sketch

### **Third-Party Hardware**

Support for third-party hardware can be added to the **hardware** directory of your sketchbook directory. Platforms installed there may include board definitions (which appear in the board menu), core libraries, bootloaders, and programmer definitions. To install, create the **hardware** directory, then unzip the third-party platform into its own sub-directory. (Don't use "arduino" as the sub-directory name or you'll override the built-in Arduino platform.) To uninstall, simply delete its directory.

### **Serial Monitor**

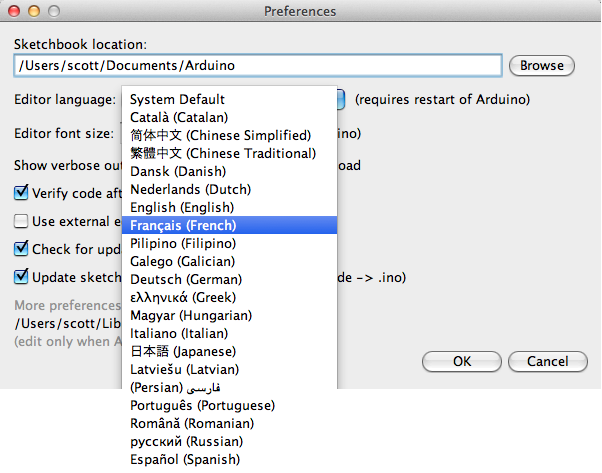
Displays serial data being sent from the Arduino or Genuino board (USB or serial board). To send data to the board, enter text and click on the "send" button or press enter. Choose the baud rate from the drop-down that matches the rate passed to **Serial.begin** in your sketch. Note that on Windows, Mac or Linux, the Arduino or Genuino board will reset (rerun your sketch execution to the beginning) when you connect with the serial monitor.

You can also talk to the board from Processing, Flash, MaxMSP, etc (see the [interfacing page](http://www.arduino.cc/playground/Main/Interfacing) for details).

### **Preferences**

Some preferences can be set in the preferences dialog (found under the **Arduino** menu on the Mac, or **File** on Windows and Linux). The rest can be found in the preferences file, whose location is shown in the preference dialog.

### **Language Support**



Since version 1.0.1 , the Arduino Software (IDE) has been translated into 30+ different languages. By default, the IDE loads in the language selected by your operating system. (Note: on Windows and possibly Linux, this is determined by the locale setting which controls currency and date formats, not by the language the operating system is displayed in.)

If you would like to change the language manually, start the Arduino Software (IDE) and open the **Preferences** window. Next to the **Editor Language** there is a dropdown menu of currently supported languages. Select your preferred language from the menu, and restart the software to use the selected language. If your operating system language is not supported, the Arduino Software (IDE) will default to English.

You can return the software to its default setting of selecting its language based on your operating system by selecting **System Default** from the **Editor Language** drop-down. This setting will take effect when you restart the Arduino Software (IDE). Similarly, after changing your operating system's settings, you must restart the Arduino Software (IDE) to update it to the new default language.

### **Boards**

The board selection has two effects: it sets the parameters (e.g. CPU speed and baud rate) used when compiling and uploading sketches; and sets and the file and fuse settings used by the burn boot-loader command. Some of the board definitions differ only in the latter, so even if you've been uploading successfully with a particular selection you'll want to check it before burning the bootloader. You can find a comparison table between the various boards [here](https://www.arduino.cc/en/Products/Compare).

Arduino Software (IDE) includes the built in support for the boards in the following list, all based on the AVR Core. The [Boards Manager](https://www.arduino.cc/en/Guide/Cores) included in the standard installation allows to add support for the growing number of new boards based on different cores like Arduino Due, Arduino Zero, Edison, Galileo and so on.

* Arduino Yùn   
  An ATmega32u4 running at 16 MHz with auto-reset, 12 Analog In, 20 Digital I/O and 7 PWM.
* Arduino/Genuino Uno   
  An ATmega328 running at 16 MHz with auto-reset, 6 Analog In, 14 Digital I/O and 6 PWM.
* Arduino Diecimila or Duemilanove w/*ATmega168*   
  An ATmega168 running at 16 MHz with auto-reset.
* Arduino Nano w/*ATmega328*   
  An ATmega328 running at 16 MHz with auto-reset. Has eight analog inputs.
* Arduino/Genuino Mega 2560   
  An ATmega2560 running at 16 MHz with auto-reset, 16 Analog In, 54 Digital I/O and 15 PWM.
* Arduino Mega   
  An ATmega1280 running at 16 MHz with auto-reset, 16 Analog In, 54 Digital I/O and 15 PWM.
* Arduino Mega ADK   
  An ATmega2560 running at 16 MHz with auto-reset, 16 Analog In, 54 Digital I/O and 15 PWM.
* Arduino Leonardo   
  An ATmega32u4 running at 16 MHz with auto-reset, 12 Analog In, 20 Digital I/O and 7 PWM.
* Arduino Micro   
  An ATmega32u4 running at 16 MHz with auto-reset, 12 Analog In, 20 Digital I/O and 7 PWM.
* Arduino Esplora   
  An ATmega32u4 running at 16 MHz with auto-reset.
* Arduino Mini w/*ATmega328*  
  An ATmega328 running at 16 MHz with auto-reset, 8 Analog In, 14 Digital I/O and 6 PWM.
* Arduino Ethernet   
  Equivalent to Arduino UNO with an Ethernet shield: An ATmega328 running at 16 MHz with auto-reset, 6 Analog In, 14 Digital I/O and 6 PWM.
* Arduino Fio   
  An ATmega328 running at 8 MHz with auto-reset. Equivalent to Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ATmega328, 6 Analog In, 14 Digital I/O and 6 PWM.
* Arduino BT w/*ATmega328*   
  ATmega328 running at 16 MHz. The bootloader burned (4 KB) includes codes to initialize the on-board bluetooth module, 6 Analog In, 14 Digital I/O and 6 PWM..
* LilyPad Arduino USB   
  An ATmega32u4 running at 8 MHz with auto-reset, 4 Analog In, 9 Digital I/O and 4 PWM.
* LilyPad Arduino   
  An ATmega168 or ATmega132 running at 8 MHz with auto-reset, 6 Analog In, 14 Digital I/O and 6 PWM.
* Arduino Pro or Pro Mini (5V, 16*MHz*) w/*ATmega328*   
  An ATmega328 running at 16 MHz with auto-reset. Equivalent to Arduino Duemilanove or Nano w/ ATmega328; 6 Analog In, 14 Digital I/O and 6 PWM.
* Arduino NG or older w/*ATmega168*   
  An ATmega168 running at 16 MHz without auto-reset. Compilation and upload is equivalent to Arduino Diecimila or Duemilanove w/ ATmega168, but the bootloader burned has a slower timeout (and blinks the pin 13 LED three times on reset); 6 Analog In, 14 Digital I/O and 6 PWM.
* Arduino Robot Control   
  An ATmega328 running at 16 MHz with auto-reset.Arduino Robot Motor   
  An ATmega328 running at 16 MHz with auto-reset.

**CHAPTER 6: ADVANTAGES AND DISADVANTAGES**

The hallmark of Talking plant is the system’s ability to monitor the growth of a plant even though we are far from it. Efficient and effective monitoring of plant, OLED display used to display any message which is sent by the person through application in the form of text regarding anything and also health status of plant is displayed on the OLED display.

**Easy Installation**

**Water Conservation**

Only required amount of water is used with respect to the humidity and moisture levels of the plant and soil.

**DISADVANTAGES:**

**High-Maintenance**

Primarily, these systems require more regular maintenance than conventional plantation.

**CHAPTER 7: RESULTS**

**7.1 Results:**

The project “**TALKING PLANT**” was designed a system which gives the weather and soil monitoring status using humidity sensor, soil moister sensor. Health of a plant can be maintained and monitored from far distances digitally.

**All figures in sequence step by step**

**7.2 Conclusion:**

This review is proposed to supports the growth of a plant which can be grown and monitored from far distances without any difficulty. Microcontroller in the system promises about increase in systems life by reducing the power consumption resulting in lower power consumption. It is very useful for people who are very busy nowadays in this mechanical life. Talking Plant is effective and efficient working to monitor the life of a plant from far places without any difficulty.

* 1. **Future Scope:**

The performance of the system can be further improved in terms of the operating speed, more sensors can be interfaced by using advanced versions of controllers. This system can be widely used anywhere regarding the growth of plants and monitoring it digitally in this digital world.

**REFERENCES**

The sites which were used while doing this project:

1. [Research](http://www.arduino.cc) Paper(Journal, conference etc)

2. [www.allaboutcircuits.com](http://www.allaboutcircuits.com)

3. [www.microchip.com](http://www.microchip.com" \t "_blank)

4. [www.howstuffworks.com](http://www.howstuffworks.com" \t "_blank)

5. [www.arduino.cc](http://www.arduino.cc)

6. [www.allaboutcircuits.com](http://www.allaboutcircuits.com)

# **CHAPTER 8: Teamwork**

### 8.1. Summary of team work

### 8.1.1 Attributes

|  |  |
| --- | --- |
| 1 | Attends group meetings regularly and arrives on time. |
| 2 | Contributes meaningfully to group discussions. |
| 3 | Completes group assignments on time. |
| 4 | Prepares work in a quality manner. |
| 5 | Demonstrates a cooperative and supportive attitude. |
| 6 | Contributes significantly to the success of the project. |

### 8.1.2 Score

**1**=strongly disagree; **2**=disagree; **3**=agree; **4**=strongly agree

**Student 1: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Student 2: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |  |
| --- | --- | --- |
| Student 1 | Evaluated by | |
| Attributes | Student 2 |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
|  | **Grand Total** |  |

|  |  |  |
| --- | --- | --- |
| Student 2 | Evaluated by | |
| Attributes | Student 1 |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
|  | **Grand Total** |  |

|  |  |  |
| --- | --- | --- |
| **Signature of**  **Student 1** |  | **Signature of**  **Student 2** |

**Appendix**

**Paste your code here**