الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'enseignement Supérieur et de la Recherche scientifique

Université Mohamed Khider Biskra
Faculté des Sciences et de la Technologie
Département de Génie Electrique
Filière : Automatique
Option : Automatique et Informatique industrielle

Réf:............

Mémoire de Fin d'Etudes
En vue de l'obtention du diplôme :

MASTER

Thème

Internet of Things based on ESP8266

Présenté par :
LALOUANI Abdelhak

Soutenu le : 23 Juin 2018

Devant le jury composé de :

TOUBA Moustafa	M.C.A	Président
BOUMEHRAZ Mouhamed	Prof	Encadreur
ABDOU Latifa	M.C.A	Examinateur

Année universitaire : 2017 / 2018

الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement Supérieur et de la recherche scientifique

Université Mohamed Khider Biskra
Faculté des Sciences et de la Technologie
Département de Génie Electrique
Filière : Automatique
Option : Automatique et Informatique industrielle

Mémoire de Fin d'Etudes
En vue de l'obtention du diplôme:

MASTER

Thème

Internet of Things based on .ESP8266
.Application to a green house

Présenté par :
LALOUANI Abdelhak

Avis favorable de l'encadreur :
BOUMEHRAZ Mouhamed

Avis favorable du Président du Jury
TOUBA Moustafa

Cachet et signature

الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement Supérieur et de la Recherche scientifique

Université Mohamed Khider Biskra
Faculté des Sciences et de la Technologie
Département de Génie Electrique
Filière : Automatique
Option : Automatique et Informatique industrielle

Thème :

Internet of Things based on .ESP8266

.Application to a green house

Proposé par : BOUMEHRAZ Mouhamed
Dirigé par : BOUMEHRAZ Mouhamed

RESUMES (Anglais et Arabe)

This end of studies' project wanted to give an overview on the Internet of Things and its projects using the ESP8266 modules. We tried in this work to get closer to the Internet of Things and its world, and we investigated the origin of its term, how it works, its architecture and its most protocols. We also identified the wireless modules used in IoT projects, where we focused more on the ESP8266 modules.

Finally, we built an IoT project using the ESP8266 modules represented in a smart and wireless greenhouse.

أردنا من خلال هذه المذكرة أن نعطي نضرة عامة عن انترنت الأشياء ومشاريعها التي تستعمل ESP8266. حاولنا في هادا العمل أن نقترب أكثر من انترنت الأشياء وعالمها، حققنا في أصل مصطلحها، كيف تعمل، هندستها وأهم بروتوكولاتها. كما عرفنا أهم الأجهزة اللاسلكية في مشاريع انترنت الأشياء، أين ركزنا أكثر على أجهزة ESP8266.
في الأخير، قمنا ببناء مشروع في انترنت الأشياء باستعمال أجهزة ESP8266 يتمثل في بيت بلاستيكي ذكي.

Dedication

I dedicate this modest work to my dear parents, who showed me their support in my choices, their attentions and their encouragement. With, unconditional love and supplications that have always accompanied me. Without missing my brothers and my dear sister, Which I always find them beside me when needed.

Special thanks to my colleagues and friends Mohamed and Massinissa, which helped me provide the hardware requirements, and my friends Islam and Sedik, which helped me print the work, as well as all those who contributed from near or far. I dedicate this work to them.

Dedication

Acknowledgment

First and foremost, I thank «**Allah** » that helped me and gave me health, courage, luck and patience during this long year of study.

Secondly, the first person I would like to present my thanks is my supervisor «Mr. BOUMEHRAZ Mohamed», for guidance, and confidence to carry out this work at the highest level.

Then, my heartfelt thanks also go to the members of the jury « Mrs. ABDOU Latifa» and « Mr. TOUBA Moustafa» for their interest in my project by agreeing to examine my work.

Acknowledgment

I would like to express my gratitude to the friends and colleagues who gave me their moral and intellectual support throughout my journey.

Finally, I would like to express my gratitude to all those who have contributed directly or indirectly to this work.

Tables List

Figures List

Figures List

Acronyms

ADC	Analog-to-Digital Conversion
AC	Alternating Current
ARP	Address Resolution Protocol
Auto-ID	Automatic identification
BLE	Bluetooth Low Energy
Bluetooth SIG	Bluetooth Special Interest Group
cm	Centimeter
DARPA	Defense Advanced Research Project Agency
DAS	Data Acquisition Systems
DevKit	Development Kit
DHT11	Digital Humidity and Temperature sensor
DIL	Dual In Line
DIL ICs	Dual In-Line packaged Integrated Circuits
DNS	Domain Naming System
EPC	Electronic Product Code
ETS	Electron Transport System
EV	Electric Vehicle
FCC	Federal Communications Commission
FDDI	Fiber Distributed Data Interface
FreeRTOS	Free Real Time Operating System
FTP	File Transfer Protocol
GAS ID	Google Apps Scripts Identification
GHz	Gigahertz
G-mail	Google mail service
GND	The ground
GPIO	General-Purpose Input/Output
GPS	Global Positioning System
HDLC	High-Level Data Link Control
hPa	hectopascal
HTTP	Hypertext Transfer Protocol
I/O	Input Output
I^2C	Inter Integrated Circuit
IAB	Internet Architecture Board
ICMP	Internet Control Message Protocol
IDE	Integrated Development Environment
IEEE	The Institute of Electrical and Electronics Engineers
IGMP	Internet Group Management Protocol
IoT	Internet of Things
IP	Internet Protocol
ISM band	Industrial, Scientific and Medical band
ISO	International Organization for Standardization
IT	Information Technology
ITU	International Telecommunications Union
LAN	Local Area Network

Acronyms

LDR	Light Dependent Resistor
LED	Light-Emitting Diode
LLC	Logical Link Control
M2M	Machine to Machine
M2P	Machine to Person
MAC	Media Access Control
mBar	millibar
Mbps	Megabits per second
MIT	Massachusetts Institute of Technology
NodeMCU	Node Micro-Controller Unit
OSI	Open Systems Interconnection
Pa	the Pascal
PAN	Personal Area Network
PC	Personal Computer
PCB	Printed Circuit Board
PDAs	Personal Digital Assistant devices
PDU	Protocol Data Unit
PN	Positive Negative
PPP	Point-to-Point Protocol
PWM	Pulse-Width Modulation
RAM	Random Access Memory
RDP	Remote Desktop Protocol
RF	Radio Frequency
RFI	Radio Frequency Interference
RFID	Radio Frequency Identification
RISC	Reduced Instruction Set Computer
ROM	Read Only Memory
RX	Receive data
SCL	Serial Clock Line
SDA	Serial Data Access
SDK	Software Development Kit
SI	International System of Units
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SoC	System on Chip
SPI	Serial Peripheral Interface
SSID	Service Set Identifier
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol / Internet Protocol
TFTP	Trivial File Transfer Protocol
TX	Transmit data
UART	Universal Asynchronous Receiver-Transmitter
UDP	User Datagram Protocol
URL	Uniform Resource Locator
USA	United States of America
USB	Universal Serial Bus
WAN	Wide Area Network
WEP	Wired Equivalent Privacy
WLAN	Wireless Local Area Network
WMM	Wi-Fi Multi Media

WMM-PS	Wi-Fi Multi Media Power Save
WPA	Wi-Fi Protected Access

Content

Content

Introduction

Our way of life, our jobs, our needs and everything is changing and evolving quickly in this modern world.

The world is in a race to reach the greatest possible evolutions that provide the welfare of man, including the automation system, which itself is evolving to achieve this goal.

The Automation of systems became requiring more and more a direct communication between the things without any human intervention, which led, in addition to the need to create databases for some products through the Internet, to the emergence of the Internet of Things.

The Internet of Things is expanding at a rapid rate, and it is becoming increasingly important for professionals to understand what it is, how it works, and how to harness its power to improve business.

We can define the Internet of Things as advanced automation and analytics system that exploits networking, sensing, big data, and artificial intelligence technology to deliver complete systems for a product or service.

Our end-of-study project will be concerned about the subject of the Internet of things (IoT), where we will see, in four chapters, an overview of it and the most devices used in its projects, then we will try to make an Internet of Things project using the wifi module “ESP8266”.   We will try to build a smart greenhouse where we can read and store the reads of its physical parameters such as the temperature, the humidity , the pressure and the amount of light inside the greenhouse, in addition to the automatic irrigation, air conditioning, and fertilizing.

Introduction

In the first chapter, we will take a general idea about the Internet of Things, its story and the origin of its term, in addition to a technical view of how it works, its architecture and its most protocols.

The second chapter will be dedicated to recognizing the wireless modules used in IoT projects, and their most important characteristics. Moreover, we will focus on the ESP8266 modules, their types and their most providers.

Chapter 3 will be devoted to the requirements that we need in building of our smart greenhouse project, where we classified the needs into two parts, hardware needs and software.

In the fourth chapter, we will take a look at the details of building our smart greenhouse. We did a hardware discussion where we discussed the wiring and the hardware setup, and a software discussion where we discussed the software steps followed in this project.

We will finish this end-of study project by a retrospective conclusion of the different phases of our work.

1.1. Introduction

After the World Wide Web in the 90’s and the mobile internet in the beginning of the new millennium, we are trending toward the third phase of the internet revolution[1].

In the early 2000’s, at MIT’s Auto-ID labs, one of British technology pioneers his name is Kevin Ashton was laying the groundwork for what would become the IoT(Internet of Things) [3,4].

This term meansthe network of physical things (devices) embedded with electronics (sensors, actuators …) and network connectivity. It allows devices to be sensed or controlled remotely without changing network infrastructure[3,4].

The term "Things", in IoT sense, can refer to a wide variety of devices, for example, vehicle sensors linked with Wi-Fi shield,cameras streaming live feeds of wild animalsorfield operation devices that assist firefighters in their search and rescue operations.It can be anything we can link with internet shield. The Legal scholars suggest regarding "things" like an "inextricable mixture of software, hardware, service and data "[3].

Simply, the internet of things links the things or the objects of the real world with the virtual world; it can be explained in form of an equation stating:

Physical Object + Controllers, Sensors and Actuators + Internet = Internet of Things.

1.2. What is the origin of the term "Internet of Things"

The phrase "Internet of Things" was coined some 20 years ago by Kevin Ashton one of the founders of the original MIT Auto-ID Center, in 1999 and David L Brock in 2001.Ashton conceived this notion when he did a research for methods that Procter &Gamble could develop its business by linking Radio Frequency Identification(RFID)information to the internet[4].

Kevin Ashton wrote in RFID Journal in June 2009 "*I could be wrong, but I'm fairly sure the phrase "Internet of Things" started life as the title of a presentation I made at Procter & Gamble (P&G) in 1999*"[6].

Auto-ID, this term may mean a large class of identification technologies used in industry, its function is to reduce errors, automate, and increase efficiency. These identification technologies include biometrics, sensors, and smart cards. However, the Auto-ID technology on the main stage has been (RFID) Radio Frequency Identification since 2003[1].

Auto-ID Labs network is a group of research specialized in the field of networked radio-frequency identification (RFID) and emerging sensing technologies, these labs contain seven research universities from seven different countries in four continents. These institutions met together to design the architecture for the Internet of Things with EPCglobal.

The topics of the research in the labs firstly were about RFID technology and now include sensor networks and new emerging sensing technology[1].

The internet of things concept came officially into the light when the ITU (International Telecommunications Union) published its first report on the subject in 2005. At that time, the head of the ITU's Strategy and Policy Unit, Lara Srivastava, said:"*It's safe to say that technology today is more pervasive than we would ever have imagined possible 10 years ago. Similarly, 10 years from now things will continue in this general direction. That's what these new technologies are telling us*"[1].

1.3. The internet in the Internet of Things

The Internet of Things consists of two important parts, part of the things, which is the hardware part, includes the physical material used in the IoT such as the sensors, the actuators, and the objects that we need to link them to the internet. Moreover, the second part which is the internet that includes the software part, such as the protocols of communication, the interfaces, and the computing systems.

Therefore, the internet occupies an important and large place in the Internet of Thingsfield.In the large-scale networks, efficient and reliable communication can be one of the most complex tasks. All data networks are based on the OSI (Open Systems Interconnection). The OSI model was introduced in1984, by ISO (the International Organization for Standardization) to address this compound problem. The model is a conceptual model purposed to standardize and describe the main communication functions of any computing system or any telecommunication, regardless of their underlying internal structure and technology. Its objective is how interoperate diverse communication systems with standard protocols[4].

1.3.1. The Open Systems Interconnection Model

The OSI model uses the concept “divide & conquer concept” which virtually breaks down network communication responsibilities to smaller functions, their name is layers, to make them easier to control.

	Name of layer	**Data formats for each layer**
Layer 7	**APPLICATION**	Data
Layer 6	**PRESENTATION**	Data
Layer 5	**SESSION**	Data
Layer 4	**TRANSPORT**	Segment
Layer 3	**NETWORK**	Packets
Layer 2	**DATA LINK**	Frames
Layer 1	**PHYSICAL**	Bits

Table 1.1 OSI layers.

As shown in (**Table 1.1**), the OSI model consists of seven layers; each layer provides some well-defined services to the adjacent layer (upper or lower layer).

Seventh layer ⇒ application layer: is the closest layer to the end user, it is an abstraction layer, which specifies the interface methods and shared protocols. Both the user and the application layer interact directly with the software application using the higher-level protocols such as HTTP (Hypertext Transfer Protocol), FTP (File Transfer Protocol), TFTP (Trivial File Transfer Protocol), SMTP (Simple Mail Transfer Protocol), SNMP (Simple Network Management Protocol), RDP (Remote Desktop Protocol) and DNS (Domain Naming System).

In brief, the application layer Provides access to available resources.

Sixth layer ⇒ presentation layer:it is under the application layer, the presentation layer is where operating system services (Windows, Mac OS, Linux, Unix … etc.) reside. This layer establishes context between application layers entities that can use different semantics and syntax if there is a mapping between them provided by presentation service. If a mapping is available, the presentation service encapsulates its data into session protocol data units and passes them down the protocol stack. As an example for the work of presentation layer, if you want to access your bank account via the internet, the presentation layer provides to you a secure connection for using.Furthermore, the presentation layer will encrypt the login information of your account before transmission. Finally, at the Internet server of your financial institution, the presentation layer will decrypt the login information of your account to make it available for processing.

In brief, the presentation layer translates, encrypts and compresses the data.

Fifth layer ⇒ session layer: Underneath the presentation layer, there is the session layer that its responsibility is session check-pointing and recovery. It deals with the communication to open a session between two network elements and it properly combines or synchronizes information of different streams, or originating from different sources.

In brief, the session layer establishes, manages and terminates communicative sessions.

Fourth layer ⇒ transport layer: it delivers data to the opportune application process on host computers and it process end-to-end communication between two endpoints.In this layer, to decide the amount of the information that should be sent at time between two endpoints, it is favorable to use the windowing concept; this technique is used as a method of controlling the flow of packets between computers or network hosts. It is called also sending windows method.

Third layer ⇒ network layer: this layer is where the routers operate, in this layer, data are packaged into packets or 'IP datagrams', these IP datagrams contain destination and source IP address information, which is used to forward the datagrams between hosts , the third layer is responsible for providing, the switching, the forwarding and routing technologies.A routing protocol define the communication method of the routers with each other, exchanging information, which allows them to choose routes between any two nodes on a computer network.It should be noted that while network and transport layers (third and fourth layers) are separate in theory, they are exceedingly related to each other in practice in a typical way.The famous internet protocol, which is TCP/IP (Transmission Control Protocol / Internet Protocol), comes from the network layer protocol, which is IP, and the transport layer, which is TCP.

Second layer ⇒ data link layer: the data link layer is the protocol layer; its function is transferring data between the nodes in neighboring network in WAN (Wide Area Network) or between nodes that should be in the same LAN (Local Area Network) segment.

This layer is concerned with delivery of frames between devices on the same LAN. Data-Link frames don't cross the limits of a local network.Inter-network routing and global addressing are handled by network

Layer,allowing the frames to focus on local delivery, addressing, and media arbitration. In this case, the second layer is like a neighborhood traffic police officer, it essays toarbitrate between parties contending to find a compromise,without concern for their ultimate destination.Two sublayers divide the data link layers, which are:

- MAC (Media Access Control) layer, this sublayer controls how a computer on the network gains access to the data and permission to transmit it.
- LLC (Logical Link Control) layer, this sublayer controls frame synchronization, flow control and error checking.

Examples of data link protocols are PPP (Point-to-Point Protocol), HDLC (High-Level Data Link Control) and Ethernet multimode for local area networks.

First layer ⇒physical layer:the physical layer ensures the physical transports of data. It manages the transmission of data in optical, electrical or radio frequency (depending on the type of media).The specifications of this layer manage the characteristics of the components (cables, pinouts and connectors) and data encoding method (physical data representation).The physical layer services are provided by network equipment that content modem, hub, fiber media converter, repeater and network interface controller.

1.3.2. How devices communicate in OSI model

To understand how devices communicate in OSI model, we will take an example of two computers communicating with each other, here the OSI model will tell each computer what it is supposed to do when data needs to be sent or when data is received. An application (G-mail for example on user A’s computer) produces data oriented toward the other device (user B’s computer). Each layer encapsulates its data to the front of the data it receives from the layer above it. Data encapsulation is the process that each layer putts its own data information (called headers) around the data that are received from the previous layer. Encapsulated data is transmitted in PDUs (Protocol Data Units) and each layer provides its compatible unit, the compatible unit of each layer is explained in (**Table 1.1)**, the PDU is a unit of measurement for the exchanged information in computer network. PDUs are passed down through the stack of layers until they can be transmitted over the physical layer. In the physical layer, the PDUs are sliced into bits, these bits are transmitted over the physical connection. The physical layer is the responsible of the physical connectivity between hosts over which all communication occurs, so it is the link between the computers. It is also responsible of the language spoken by the two computers by ensuring that both computers speak the same language on the same layer. After data arrived at the user B's computer, it passes upward and each layer decapsulate its own headers and passes data further up the stack.

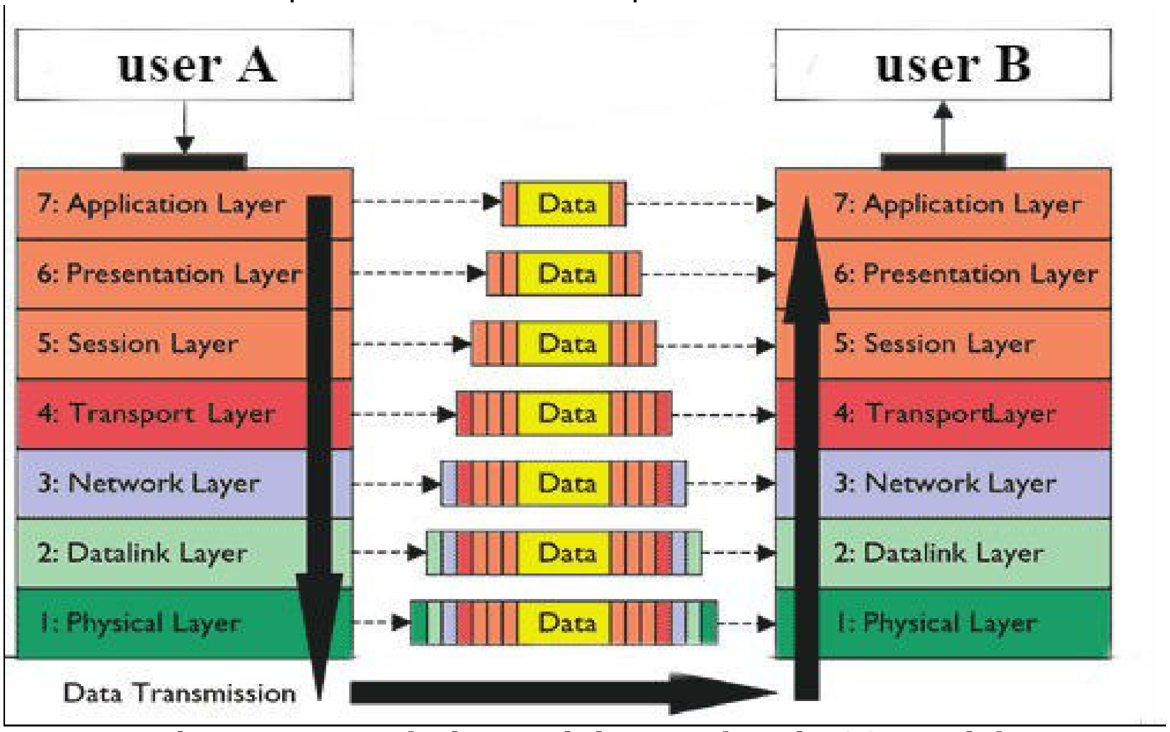


Fig 1.1Encapsulation and decapsulate in OSI model.

1.3.3. TCP/IP suite

TCP/IP (Transmission Control Protocol/Internet Protocol) or The Internet protocol suite is came from the DARPA project (Defense Advanced Research Project Agency) in USA at the ends of 60's. The protocols developed in 70’s to pass in 1980 for the universities researches using in USA. Now, the IAB (Internet Architecture Board) is the responsible of developing the TCP/IP model.

This model provides end-to-end data communication, this principle is the design framework in computer networking that specifies the data packetization way, and how the data packets should be addressed, transmitted, routed, and received. Four abstraction layers organize this functionality, and classify all related protocols according to the scope of networking involved.These layers are application layer, transport layer, internet layer and network access layer whom can be divided to data link layer, and physical layer**.** Thus, they become five layers. The (**Table 1.2**)shows the relationship between OSI model and Internet protocol suite layers.

layer 7: **APPLICATION**	layer 4: **APPLICATION**
layer 6: **PRESENTATION**	
layer 5: **SESSION**	
layer 4: **TRANSPORT**	layer 3: **TRANSPORT**
layer 3: **NETWORK**	layer 2: **INTERNET**
layer 2: **DATA LINK**	layer 1: **NETWORK ACCESS**
layer 1: **PHYSICAL**	

Table 1.2 Relationship between OSI reference model and TCP/IP.

Fourth layer ⇒ application layer: the application layer is the topmost layer in the TCP/IP model as with the OSI model. The application, presentation, session layers of the OSI layer are combined in this layer to define TCP/IP application protocols and the way to host programs interface with transport layer services to use the network.This layer contains all high-level protocols like TFTP (Trivial File Transfer Protocol), SMTP (Simple Mail Transfer Protocol), HTTP (Hyper Text Transfer Protocol).

Third layer ⇒ transport layer: this layer, which called also host-to-host transport layer, is similar to the transport layer in the OSI model, and it have the same main role. The transport protocols include TCP, which provides a one-to-one, connection-oriented, reliable communications service. TCP take care of the establishment of a TCP connection, the recovery of packets lost during transmission and the sequencing and acknowledgment of packets sent.The transport protocols include also UDP (User Datagram Protocol), which is sometimes used instead of TCP for special purposes, UDP provides a one-to-one or one-to-many, connectionless, unreliable communications service.

Second layer ⇒ internet layer: the internet layer or the networking layer is analogous to the network layer in OSI model; this layer is responsible for addressing, packaging, and routing functions. The internet protocols include IP (Internet Protocol), ARP (Address Resolution Protocol), IGMP (Internet Group Management Protocol) and ICMP (Internet Control Message Protocol).

First layer ⇒ network access layer: it has many names: link layer, network interface layer, network access layer and physical layer. This layer combines the lowest two layers of the OSI model, data link and physical layers. This layer defines the details of sending data physically through the networks, and how hardware devices signal the bits electrically or optically; it defines also how these data interface directly with a network medium, such as optical fiber, twisted pair copper wire or coaxial cable.

The protocols included in this layer are FDDI (Fiber Distributed Data Interface), Ethernet, Token Ring and ARP (Address Resolution Protocol).

1.4. The architecture of the Internet of Things

We can number about four stages in Internet of Thing architecture, each one of them with specific functionality, productivity and utilizations, the (**Table 1.3**) illustratesthe architecture of IoT[2]:

SENSORS/ACTUATORS: *wired ,wireless*
INTERNET GATEWAY: *data aggregation, measurement, control*
EDGE IT: *analytics, preprocessing*
DATA CENTER/ CLOUD: *analytics, management , archive*

Table 1.3The architecture of IoT.

STAGE 1: sensors/actuators: the sensors or the actuators form the first stage of IoT architecture.

sensors: which have another term is transducers, are devices, modules or subsystems which detect events or changes in their environmentand send the data to other electronics, which may be as simple as light or as complex as computer,however the sensors are always used with other electronics.

actuators are the devices which are responsible for moving and controlling a mechanism or system, shutting off a power supply, adjusting an air flow valve or opening a valve simply, the most actuators requirements are a control signal and a source of energy, the control signal is generally low energy may be electric current or voltage.

The data is the heart of IoT architecture and it has several types at deferent levels, there should be sensors for sensing each type of data. However, the rising of the technology help us to solve this problem using the power over Ethernet and the low power wireless sensor network technologies, etc.

STAGE 2: internet gateway: The sensors give the data in analog form, but to process the data it should be in digital form, therefore, the data should be converted from analog form to digital form for further processing. The DAS (Data Acquisition Systems) is the responsible of the task of data aggregation and other conversion functionalities; it connects to the sensor network and executes the analog to digital conversation. These aggregated and digitalized will be received by the internet gateway, then the internet gateway routes the data over wired LANs, Wi-Fi, internet, to the third stage for more processing.

STAGE 3: edge IT: After routing the digitalized data over the internet by the internet gateway, the data gets ready to cross in the IT realm. However, the data is not completely ready yet and it need more processing before send it to the data center,so, this is the role of edge IT.IT (Information Technology)is the application of computers to store,transmit,retrieveand manipulate data; the edge Information Technology (edge IT) performs more analyses to make the data ready for the data center. The edge IT processing offices are generally situated at distant locations, they reside closer to the sensors. The IoT data consumption of the network bandwidth is large, so it is prone to have systems at the edge, which can perform analytics to reduce the burden of IT infrastructurecore.

STAGE 4: data center / cloud: We can define a data center as physical site used to house computer systems and their associated components. Such as servers, routers, storage systems and telecommunications.it is responsible for storing and distributing data through an internal network or via Internet access. The data center is the last destination of our data after the three stages before. Here the data can be stored and several users will use it later. These data can be public and every one can use it, or it can be private also and there is a specific number, whichcan access to it, because of the confidentiality of information, and that raises the problem of security.

1.5. Conclusion

The Internet of Things promises to deliver a step change in individuals' quality of life and companies' productivity. In this chapter, we have made a great idea about the Internet of Things, its definition, its history and the origin of the IoT term, we have subsequently known its relationship with the internet and the models used in the internet generally and the Internet of Things particularly. In addition to mentioning the architecture of the Internet of Things.

The next chapter will be dedicated to know the Wi-Fi module used to apply the IoT, its types, its configurations and its development environments.

2.1. Introduction

Wireless technology is expanding beyond its traditional boundaries into everything from consumer electronics to industrial controls and medical devices as well as the growing movement toward the Internet of Things.

There are many internet modules, wired or wireless, but the most modules used in the internet of things are the wireless shields. The wireless modules are used to connect any electronic systems with the internet. They emerged as a link between the microcontrollers (especially the Arduino) and the internet, but now, most wireless modules integrate a self-microcontroller, thus they don’t need an external microcontroller to work.

2.2. The most wireless modules in wireless projects

Getting projects connected to the internet is relatively easy these days because of the evolution of the technology especially the wireless technologies. To choose a wireless module adapted to our project, we should know what is available in the markets these days, what is most important to our project, and what do we need from the wireless module, do we need something low power for a battery, what about distance. These questions will help us to consider which wireless is the best for our project [7].

Here is a sample of the wireless modules used in wireless projects:

- **Bluetooth LE or BLE:**

The Bluetooth low energy (BLE, Bluetooth LE or Bluetooth smart) is a wireless PAN (Personal Area Network) technology designed by the standards organization “Bluetooth SIG” (Bluetooth Special

Interest Group) which is responsible of the development of bluetooth standards. This chip is designed for the use by internet-connected machines and appliances; it was introduced as an alternative to Bluetooth classic in the Bluetooth 4.0 specification. It is designed to reduce power consumption and costs while maintaining a similar communication range. The BLE uses 2.4 GHz radio frequencies, which allow dual mode devices to share just one antenna. It is natively supported by PC operating systems including Windows8, Windows10, Linux, MacOS and mobiles operating systems including Android, Windows phone, Blackberry and iOS. It is used in many applications such as health care, sports and fitness, internet connectivity and generic sensors [7-9].

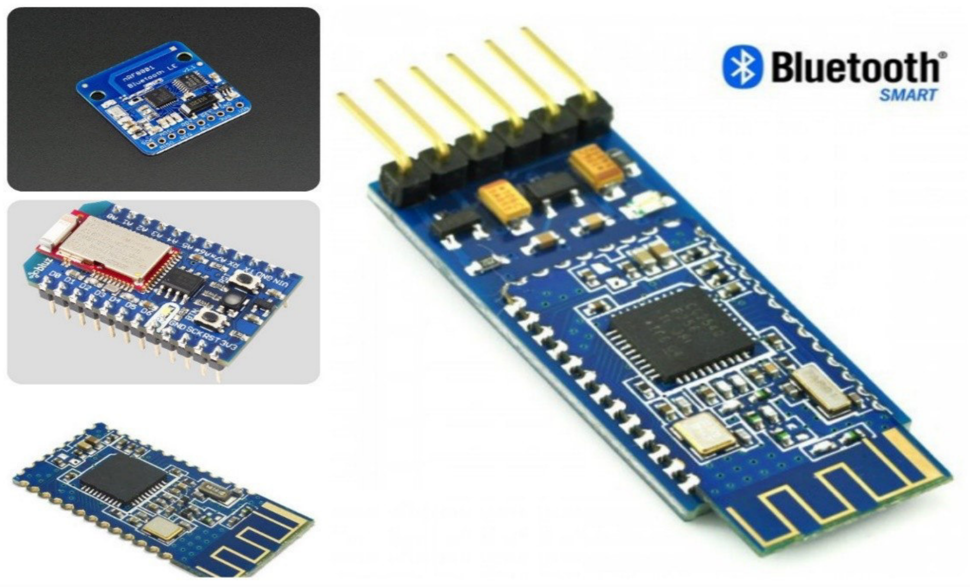


Fig 2.1 Bluetooth LE modules.

- **<u>nRF24L01 wireless module</u>:**

Serial or UART is a method for two devices to talk to each other, this method mostly needs three wires at least: TX, RX and GND. nRF24L01 module are created to remove the wires between the two devices [7].

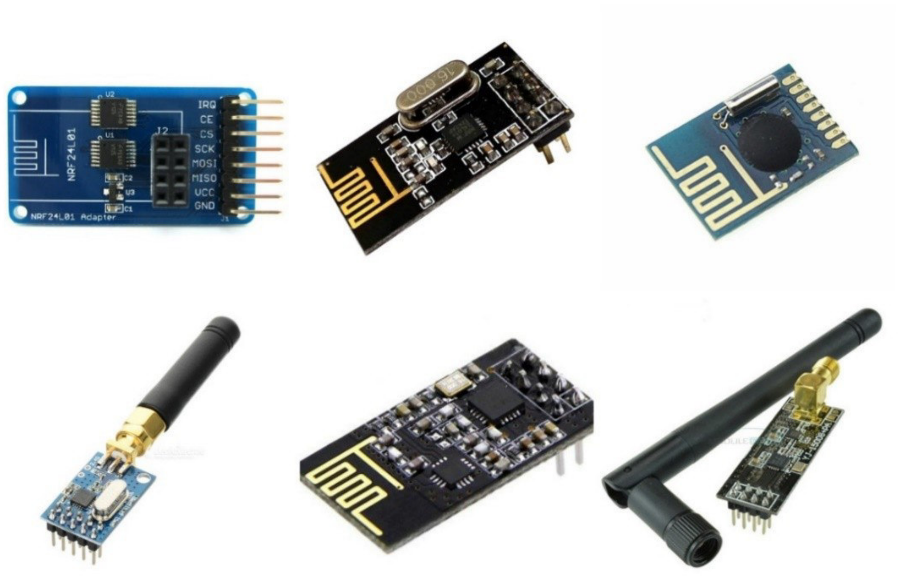


Fig 2.2 nRF24L01 modules.

The nRF24L01 module is a highly integrated, ultra-low power radio frequency transceiver for the 2.4GHz ISM band (Industrial, Scientific and Medical band), it is used in many application such as Wireless PC Peripherals, Game controllers, RF remote controls for consumer electronics and Sports watches and sensors [10,11].

- **Arduino Wi-Fi shield:**

Because of the multiple types of heavy encryption and huge number of layers to the wireless and Ethernet aspects, the 8-bit microcontrollers like Arduino or even 16-bit microcontrollers like a MSP430 can't handle the adding of Wi-Fi protocols. That is why the engineers of Arduino thought about the invention of a Wi-Fi shield [7].

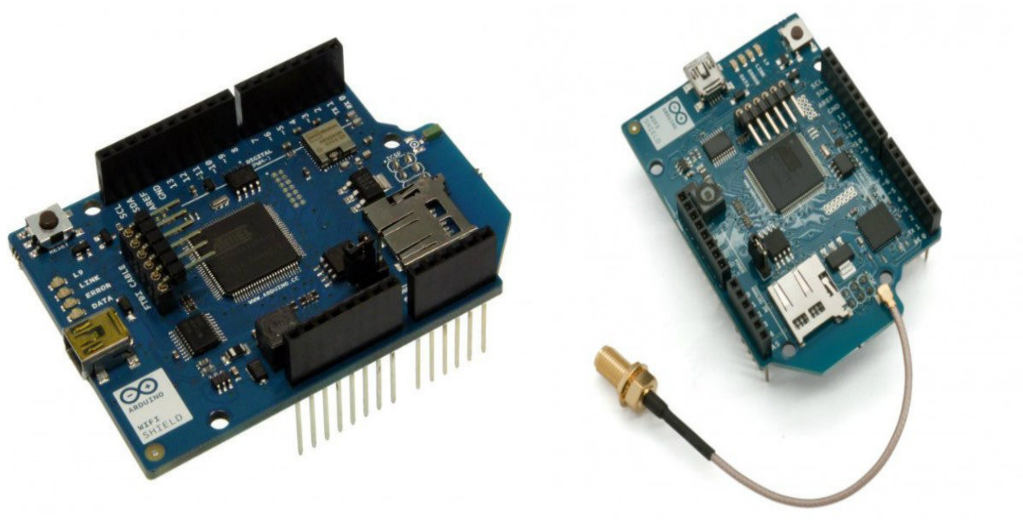


Fig 2.3 Arduino Wi-Fi shields.

The Arduino Wi-Fi shield allows an Arduino board to connect to the internet using the Wi-Fi (called also 802.11 wireless specification). It is based on the HDG204 Wireless LAN 802.11b/g System in-Package. It can connect to wireless networks that operate according to the 802.11g and 802.11b specifications [12].

An IP (network stack) capable of both UDP and TCP is provided by AT32UC3 microcontroller, which uses the Wi-Fi library to write the sketches, these sketches connect to the internet using the shield that can connect to the Arduino board using long wire-wrap headers that extend through the shield. This keeps the pin layout intact and allows another shield to be stacked on top [12].

- **ESP8266:**

The ESP8266 is a microcontroller integrated circuit with Wi-Fi connection developed by the Chinese manufacturer “Espressif” [13].

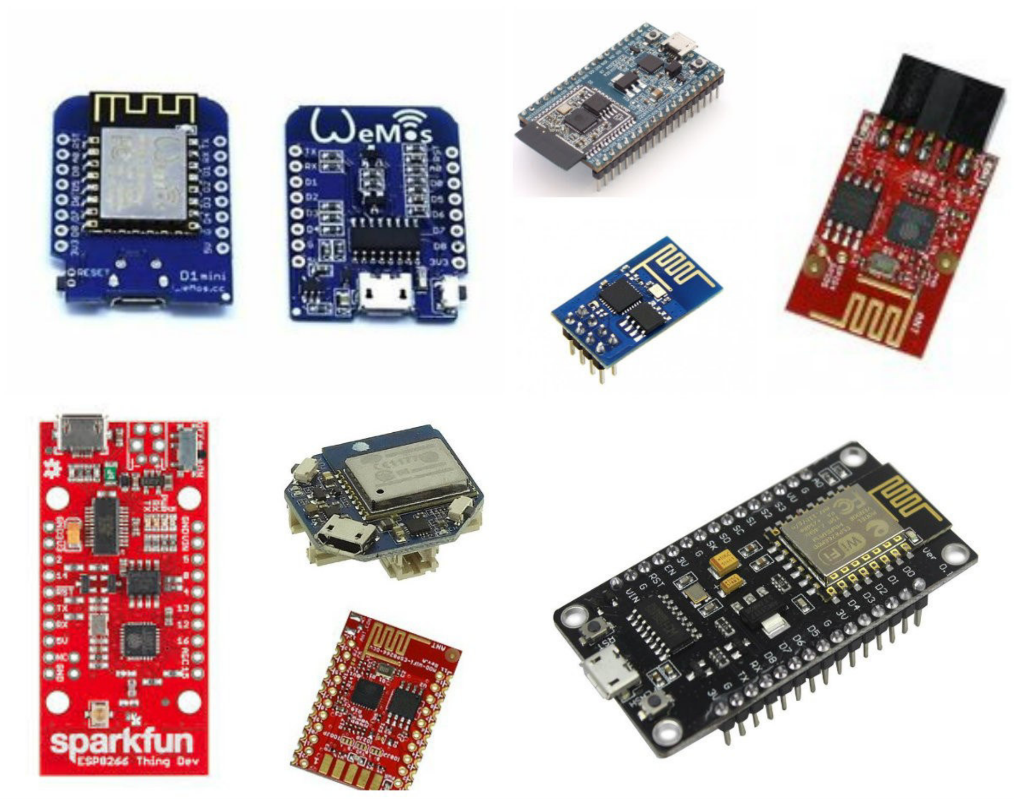


Fig 2.4 ESP8266 modules.

2.3. How to select a Wi-Fi module for IoT projects

To convoy the growing movement toward the Internet of Things, more and more engineers working on embedded systems are being asked to design interoperability systems that also use the Wi-Fi. These designs should obey one or more of the IEEE 802.11 standards. This led to the appearing of many products in this field. That is why we should know how to select our Wi-Fi chipsets for our wireless projects [16].

Many devices use Wi-Fi to connect to the network resource via a wireless network access point, these devices can be divided by their applications [16].

The most application where we can use Wi-Fi devices are:

- Consumer electronics: remote control, internet radio, home security …
- Medical fitness and healthcare: fitness equipment, patient monitor …

- Industrial controls: chemical sensors, M2M communications, wireless controller …
- Utility and smart energy: thermostat, EV chargers, lighting controls …

2.3.1. The parametric filters for our module search

When we do our search, we will find many different kinds of Wi-Fi modules and solutions, and there are many parameters that classify the modules including the data rate, range, radio frequency band, etc. the parametric filter will allow us to refine the search results according to the required parameter [16].

This is a list of sixteen parameter, we will demonstrate some of these parameters:

- Protocol / standard supported.
- Operating frequency band.
- Transmit range.
- Transmit power output, operating supply current or voltage.
- Data rate (max throughput).
- Microcontroller/microprocessor.
- Operating system (driver support).
- Antenna and connector.
- Secure Wi-Fi authentication schemes.
- Shape and size.
- Operating temperature range.
- Packaging type.
- PCB layout.
- Hardware interfaces.
- Other features (real-time clock, auto-sleep / wake-up, etc.).
- Certification/compliance.

Protocol / standard supported: we will find a number of different wireless standards on the market these days. The benefits of wireless networking depend on the standard employed. Today, the modems use WLAN including 801.11a/b/g/j/n/p/ac/ad…, each specification had a different specification requirement [16].

It is important to consider the tradeoff, like the power consumption, operating frequency band and the data rate, depend on the applications.

Operating Frequency Band: a frequency band is an interval in frequency domain, delimited by an upper frequency and a lower frequency. A radio communications signal must occupy range of frequencies carrying most of its energy, called its bandwidth [15,16].

There are three ranges used in Wi-Fi modules [16]:

- 2.4 Gigahertz (2.4 ⇒ 2.483 GHz) - 802.11b/g/n
- 5 Gigahertz (5.15 ⇒ 5.725 GHz) - 802.11a/h/j/n/ac
- 5.9 Gigahertz (5.85 ⇒ 5.9 GHz) - 802.11p

Each range is divided into a number of smaller bands or channels, similar to television channels. Countries apply their own regulations to the allowable channels, maximum power levels and allowed users within these frequency ranges [15,16].

Some Wi-Fi modules are Dual-Band Support, Dual-Band equipment is capable of transmitting in either of two different standard frequency ranges. For example, 802.11g solutions operate only in the 2.4 GHz frequency band, but 802.11n solution supports both the 2.4 and 5 GHz bands, maximizing flexibility in how mobile devices are deployed and managed [15,16].

Transmit Power Output, Operating Supply Current or Voltage: for the Wi-Fi module, higher transmit power output is required for longer transmission distanceand higher data rate. However, the major bugbear is battery life in mobile devices because of the high power consumption of Wi-Fi [16].

Secure Wi-Fi Authentication Schemes: Before allowing the devices to connect to the networks using Wi-Fi, those responsible for information security must be confident that the Wi-Fi networks and the devices that use them will protect sensitive information that are transmitted over Wi-Fi or stored on the networks. WEP (Wired Equivalent Privacy), WPA (Wi-Fi Protected Access), WPA2, WPA2 Enterprise, WPS (Wi-Fi Protected Setup), WMM (Wi-Fi MultiMedia), WMM-PS (Wi-Fi MultiMedia Power Save) are typical Wi-Fi security types; each one has advantages and disadvantages [16].

Other Features and Consideration: User programmable (I/O), Receive Sensitivity, Manufacturer, On board TCP/IP stack, Over the air firmware upgrade, Real-time clock for time-stamping, auto-sleep, and auto-wakeup modes, Price. These things can be crucial in our choice [16].

2.3.2. The tradeoff when we are selecting the module

When evaluating wireless technologies, we should consider three key factors: Power requirements, range, and data rate. If we did a comparison between different Wi-Fi protocols, we will find that 802.11b/g has the advantage in power requirements and compatibility, while 802.11n and 802.11ac have the advantage of the higher data throughput, which is the rate of successful message delivery over a communication channel [16].

There are many operational advantages provided by the 802.11n specification, higher data throughput, robust link quality and greater range. 802.11ac offers the double of the bandwidth for the 600 Mbps via 802.11n, thus 802.11ac builds on high bandwidths over extended ranges. It enables Wi-Fi solution to meet the increasing demand for high quality and high capacity real-time applications like voice and video. Using multiple antenna to increase the range and the data rate. However, these features comes at the expense of power consumption and the simplicity of the design

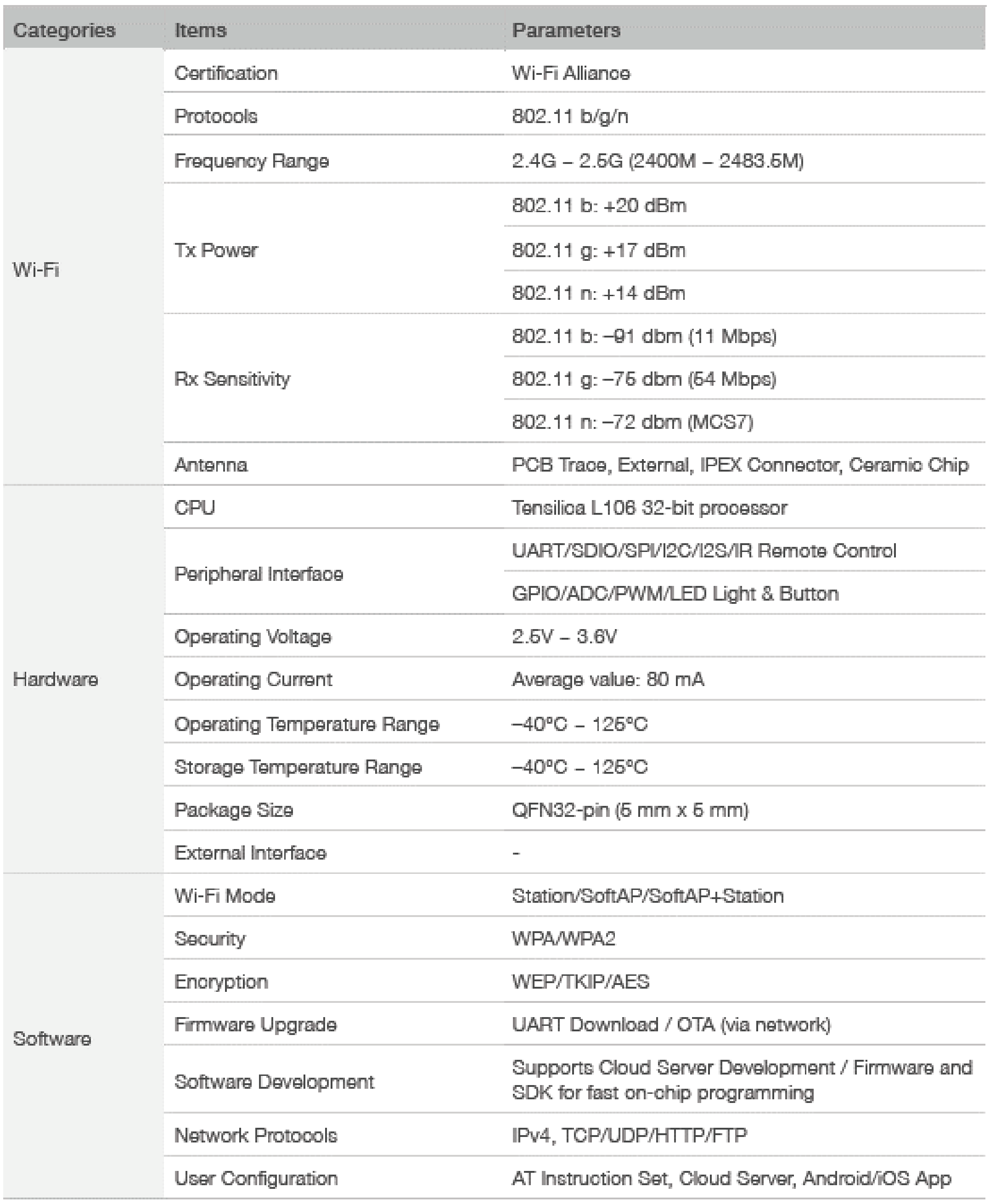
If we will use Wi-Fi devices to connect clients such as sensors, industrial and home automation products, buildings, M2M (Machine to Machine) and M2P (Machine to Person) control devices, we should know that these things are not concerned with bandwidth, but with compatibility and with battery life. Therefore, 802.11b/g supports data rates sufficient for most M2M and M2P applications. The 802.11b/g modules are also ideally suited for the embedded space as the b/g protocol is still best for fully Wi-Fi compatible, low power applications [16].

2.4. The ESP8266

With a simple research on google, we will find number of Wi-Fi modules available these days, but the famous are ESP8266 modules, which we will take as a sample.

ESP8266 series, or family, of Wi-Fi chips is provided by the fabless semiconductor company operating out of Shanghai, China, ”Espressif Systems”. Now, the ESP8266 series includes ESP8266EX and ESP8285 chips [13,14].

ESP8266EX: (simply called ESP8266) these series are the most famous and bestselling in the markets [13].the (**Table 2.1**) gives some specifications of ESP8266EX modules:



Categories	Items	Parameters
Wi-Fi	Certification	Wi-Fi Alliance
	Protocols	802.11 b/g/n
	Frequency Range	2.4G – 2.5G (2400M – 2483.5M)
	Tx Power	802.11 b: +20 dBm
		802.11 g: +17 dBm
		802.11 n: +14 dBm
	Rx Sensitivity	802.11 b: –91 dbm (11 Mbps)
		802.11 g: –75 dbm (54 Mbps)
		802.11 n: –72 dbm (MCS7)
	Antenna	PCB Trace, External, IPEX Connector, Ceramic Chip
Hardware	CPU	Tensilica L106 32-bit processor
	Peripheral Interface	UART/SDIO/SPI/I2C/I2S/IR Remote Control
		GPIO/ADC/PWM/LED Light & Button
	Operating Voltage	2.5V – 3.6V
	Operating Current	Average value: 80 mA
	Operating Temperature Range	–40°C – 125°C
	Storage Temperature Range	–40°C – 125°C
	Package Size	QFN32-pin (5 mm x 5 mm)
	External Interface	-
Software	Wi-Fi Mode	Station/SoftAP/SoftAP+Station
	Security	WPA/WPA2
	Encryption	WEP/TKIP/AES
	Firmware Upgrade	UART Download / OTA (via network)
	Software Development	Supports Cloud Server Development / Firmware and SDK for fast on-chip programming
	Network Protocols	IPv4, TCP/UDP/HTTP/FTP
	User Configuration	AT Instruction Set, Cloud Server, Android/iOS App

Table 2.1 ESP8266EX Specifications.

ESP8285: is a variation of ESP8266 with 1-Megabyte of embedded flash memory [13].

2.4.1. ESP8266 modules

There are many vendors provided their ESP8266 modules such as Espressif, AI-Thinker and other vendor**s.**

"Espressif", the Chinese manufacturer have provided four series of ESP8266-based modules until now: ESP-WROOM-02, ESP-WROOM-02D, ESP-WROOM-02U and ESP-WROOM-S2.

"AI-Thinker", the third-party manufacturer have provided more than twenty series of ESP8266-based modules until now because of the wide fame of its products, where they are considered the most widely available in the markets. We should note that the ESP8266 chip first came to attention of western makers in August 2014 with the ESP-01 module that is made by "AI-Thinker" [13]. The AI-Thinker ESP8266 collective mostly referred to as "ESP-xx modules". Most of the other boards have a popularity because of the inclusion of USB-to-UART bridge and a Micro-USB connector coupled with a 3.3-volt regulator [13].

2.4.2. How to program an ESP8266

The computer scientists refer to the process of conceiving, specifying, designing, programing, documenting, testing, and bug fixing involved in creating and maintaining application, frameworks, or other software components, as Software Development [17].

The SDK (Software Development Kit) is typically a set of software development tools, this set allows the creation of applications for a certain software package, software framework, hardware platform, computer system, video game console, operating system, or similar development platform [18]. In late October 2014, "Espressif Systems" released a SDK, which allowed the chip to be programmed, without the need for a separate microcontroller. The SDK of "Espressif Systems" is available in two versions: Espressif's official SDK based on FreeRTOS (Free Real Time Operating System), and Espressif's official SDK based on callbacks [13,14].

The ESP8266 can be programmed in other ways: [13,14]

- By the open source IoT platform NodeMCU based on Lua scripting language.
- By Arduino IDE (Arduino Integrated Development Environment) based on C++ firmware.
- By Espruino firmware based on Java Script language.
- By MicroPython firmware.
- By the open source ESP-Open-SDK based on the GCC toolchain with C language.

2.5. Conclusion

In this chapter, we have mentioned the wireless modules that can be used in wireless projects, the basis of choice between them and the tradeoff while selecting the appropriate wireless module. Then we have spotlighted the ESP8266 modules, the most manufacturers of these products and the products of each manufacturer. Finally, we have talked about the programing of these chips.

The next chapter will be dedicated to see the hardware and software requirements that we need for our project.

3.1. Introduction

To make any electronic project, always there are some requirements that we need. We can classify these needs into two classes: hardware requirements, which include everything that is tangible such as motors, wires, electronics…etc. And software requirements, which deal with defining software resource requirements and prerequisites that need to be installed on a computer to provide optimal functioning of the applications, such as the IDEs (Integrated Development Environments), the host clouds …etc. Our project needs are varying between software and hardware, each one has its own importance.

3.2. Hardware requirements

3.2.1. ESP8266 modules

In our project, the ESP8266 modules are considered the most important requirements. It can be confirmed that they are the brain of the system. For this project, we chose two different kinds of ESP8266 boards, NodeMCU board and Wemos D1 mini board. These two standalone boards do not require us to use another microcontroller such as the Arduino board. It can even be considered that they are better than the Arduino UNO thanks to the benefits that they have compared to the Arduino UNO such as: more RAM space, more flash (ROM) space (more codes can be stored),… .

NodeMCU:

Generally, the word NodeMCU refers to an open source hardware and software development environment, this environment is built around a very cheap SoC (System-on-Chip) called the ESP8266.

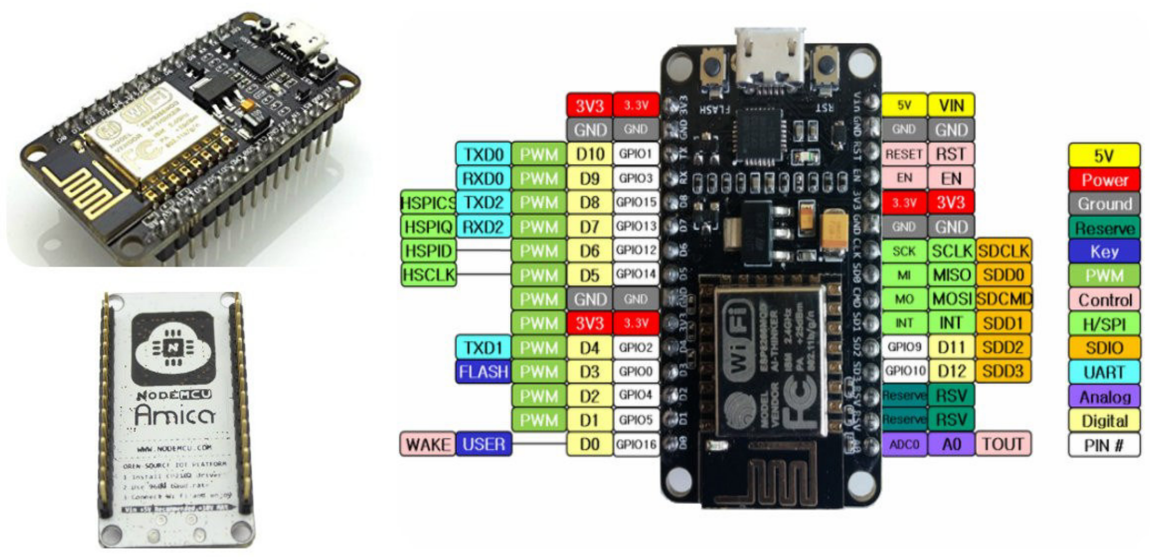


Fig 3.1 NodeMCU DevKit.

As a chip, the ESP8266 is hard to access, use and program. To use this chip alone you have to solder wires, with the appropriate analog voltage, to its pins for the simplest tasks such as powering it on or sending a keystroke to the "computer" on the chip. The program should be in low-level machine instructions, which can be interpreted by chip. This level of integration is not problem if the chip is used as an embedded controller in electronics with Wholesale production. However, for hackers, student, or hobbyists who want to experiment with it in their IoT projects, it is a huge burden and a big problem.

The NodeMCU (Node Micro-Controller Unit) project aims to simplify ESP8266 development on two levels the hardware level and the software. It has two key components: An open source ESP8266 firmware that is built on top of the chip manufacturer's proprietary SDK. And a DevKit (Development Kit) board that incorporates the ESP8266 chip on a standard circuit board. Until now, there are two generations of the NodeMCU DevKit, V0.9 (called also V1) and V1.0 (called also V2).

For this project, we have acquired the second version V1.0. These are some specification:

- Based on ESP12-E module.
- USB-to-UART Bridge.

   - Micro-USB connector coupled with a 3.3-volt power regulator.
   - Compatible with the breadboard.

- Compatible with Arduino.
- Compatible with NodeMCU Lua
- Compatible with Micro Python
- 1 analog input(one volt max input)

   - Thirteen digital input/output pins Named as follows:

ESP8266 Pin	NodeMCU DevKit	ESP8266 Pin	NodeMCU DevKit
GPIO16	D0	GPIO13	D7
GPIO5	D1	GPIO15	D8
GPIO4	D2	GPIO3	D9
GPIO0	D3	GPIO1	D10
GPIO2	D4	GPIO9	D11
GPIO14	D5	GPIO10	D12
GPIO12	D6		

Table 3.1 NodeMCU DevKit digital input/output pins.

WeMos D1 mini:

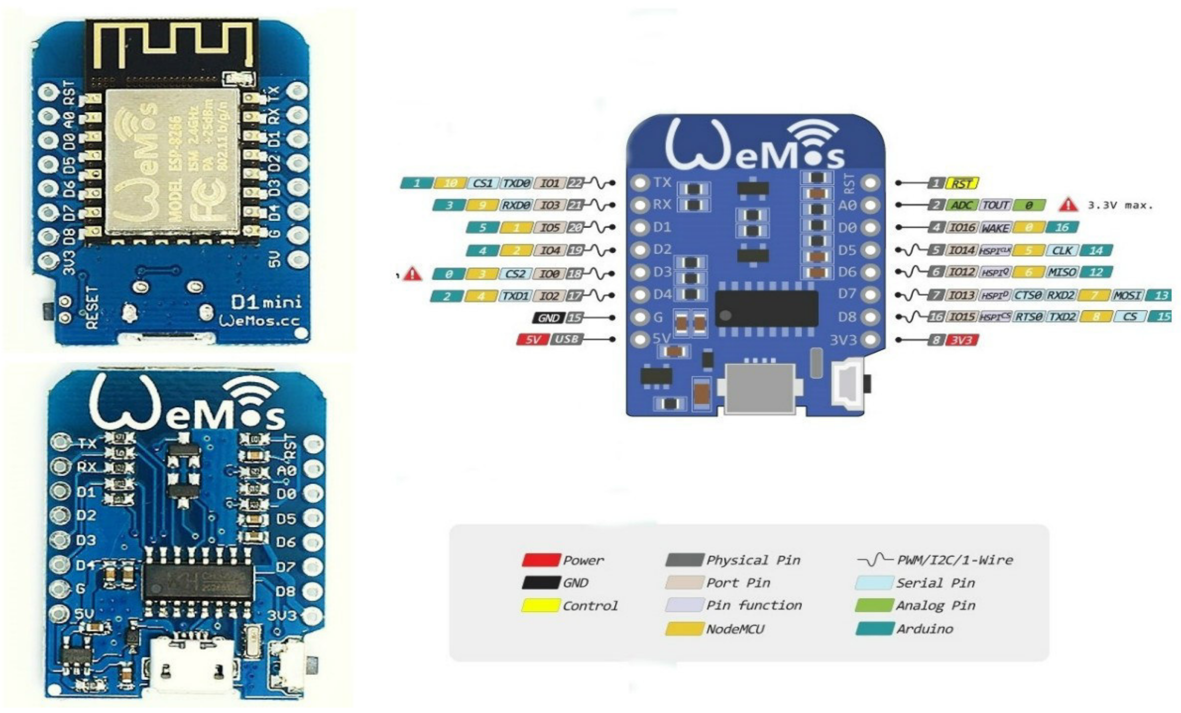


Fig 3.2 WeMos D1 mini DevKit.

WeMos D1 mini is one of the collection of development boards based on ESP8266 modules, designed by the young Chinese company “WeMos”.

This board designed to make the use of ESP8266 easy and simple. These are some of its features:

- Based on ESP-8266EX.
- Compatible with Arduino.
- Compatible with NodeMCU Lua
- Compatible with MicroPython.
- A Micro USB connection.
- One analog input (3.2V max input).
- Eleven digital input/output pins, all pins have interrupt /PWM/I^{2}C/ one-wire supported (except D0).

The digital pins are named as follow:

ESP8266 Pin	WeMos D1 mini DevKit	ESP8266 Pin	WeMos D1 mini DevKit
GPIO16	D0	GPIO14	D5
GPIO5	D1	GPIO12	D6
GPIO4	D2	GPIO13	D7
GPIO0	D3	GPIO15	D8
GPIO2	D4		

Table 3.2 WeMos D1 mini DevKit digital input/output pins.

3.2.2. The sensors

The sensors are defined as devices that transform the physical quantities into normed quantities, usually electrical quantities, which can be interpreted by a control device.

This definition gives us a vision about the place of the sensors in our project. The sensors are the links between the things that we want to detect their events and the control devices, which are in our case the ESP8266 modules.

We need for our project a number of sensors are all related to the agronomy field.

DHT11 module:

DHT11 (Digital Humidity and Temperature sensor), it is a basic, low-cost digital sensor senses, measures and reports the relative humidity in the air. It therefore measures both moisture and air temperature. The DHT11 uses a capacitive humidity sensor that relies on electrical capacitance, and a thermistor to measure the surrounding air temperature.

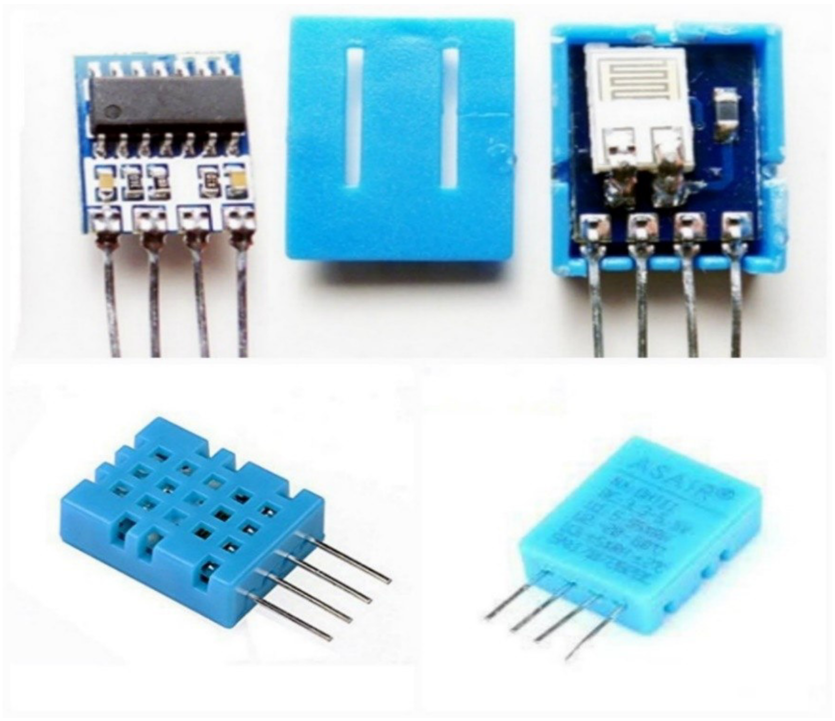


Fig 3.3 DHT11 module.

Relative humidity is the ratio of the existing moisture in the air to the greatest amount of moisture that can be maintained at that air temperature. The rise of the air temperature increases the amount of moisture can be held.

Two metal plates form the sensor; contain a non-conductive polymer film between them is responsible of collecting the moisture from the surrounding air, which causes the voltage between the two plates to change. The voltage changes are converted into a calibrated digital signal able to read by a control device.

The DHT11 module is clearly simple to use, but requires accurate and careful timing to grab data. The only real disadvantage of this sensor is that you can only obtain new data once every two seconds that is why we should use a special library for the DHT11. In brief, the features of the DHT11 are Low cost, excellent quality, relative humidity and temperature measurement, digital signal output, long distance signal transmission, strong anti-interference ability, long-term stability, fast response, and precise calibration.

The DHT11 parameters:

Electrical Characteristics	○ Power supply: DC 3~5.5V ○ Supply Current: measurement 0.3milli-ampir standby 60 micro-ampir. ○ 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, .not connected pin. 4. GND ground, the negative power
Relative humidity	○ Resolution: 16Bit. ○ Repeatability: ±1percent RH. ○ Accuracy: At 25℃ ±5percent RH. ○ Interchangeability: fully interchangeable. ○ Hysteresis: <± 0.3 percent RH. ○ Long-term stability: <± 0.5 percent RH / yr in.
Temperature	○ Resolution: 16 Bit ○ Repeatability: ±0.2℃ ○ Range: At 25℃ ±2℃ ○ Response time: 1 / e (63%) 10S

Table 3.3 DHT11 parameters.

Soil moisture sensor:

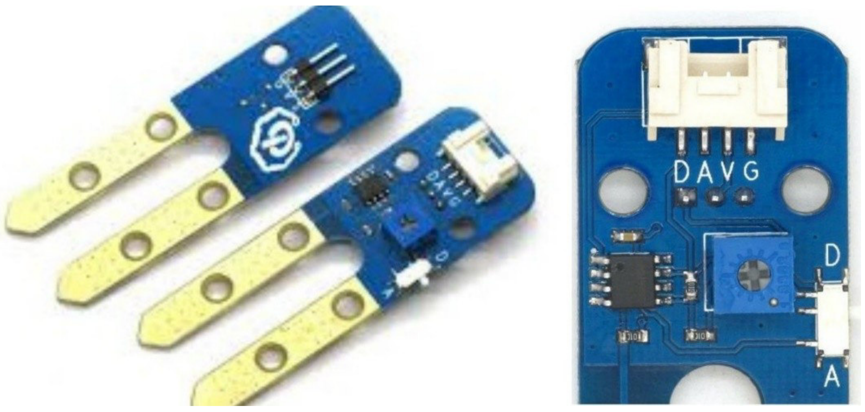


Fig 3.4 Soil moisture sensor.

The soil moisture sensor is used generally to detect the moisture content in the soil or judge if there is water around the sensor. It is based on soil resistivity measurement. A soil resistivity is a measure of how much the soil resists the flow of electricity. The soil resistivity value is subject to great variation, due to moisture, temperature and chemical content.

parameter	Min	Typical	Max	Unit
Working voltage	*2.1*	*5*	*5.5*	*VDC*
Analog output voltage (for VCC=5 volts)	*0*	*V out*	*5*	*V*
Digital output voltage (for VCC=5 volts)	*0*	*-*	*5*	*V*
Working current (for VCC=5 volts)	*-*	*5*	*-*	*mA*
Threshold hysteresis ΔU_{th}	*-*	*VCC*0.09*	*-*	*V*

For our project, we have contained an electronic brick of soil moisture sensor from "Itead Studio" company. Electronic brick of soil moisture sensor is mainly used to detect the moisture content in the soil. The control board can get the moisture value or threshold in the soil via analog or digital pins. The next table illustrates the electrical characteristics of the product.

Table 3.4 Electrical characteristics of soil moisture sensor.

Light sensor:

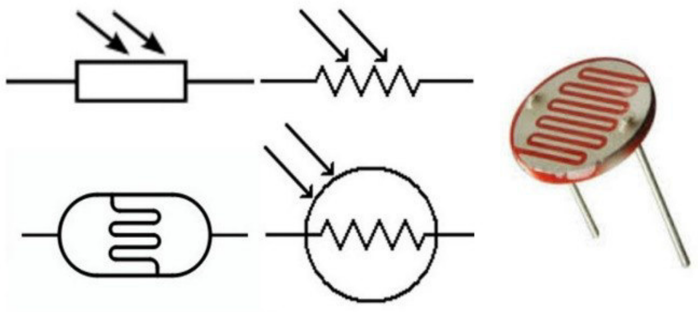


Fig 3.5 LDR.

We can call the light sensor, the photo sensor or the photoelectric device, any electronic device used to detect light. It is a passive device that convert the “light energy” whether visible or in the infrared parts of the spectrum into an electrical signal output. There are several types of light sensors; we can group them into two principal categories, those that generate electricity when exposed to light, such as Photo-emissive or Photovoltaic… etc., and those which change their electrical properties in one way or another when exposed to light such as Photo-conductors or photo-resistors.

In our project, we will use the photo resistor as a light sensor. A photo resistor or an LDR (Light Dependent Resistor) is a component that has a variable resistance that changes with the change of the intensity of the light that falls on it. The resistance of a photo resistor decreases with increasing incident light intensity; in other words, it exhibits photoconductivity.

It should be noted that photoresistors are devices that are less sensitive to light than photodiodes or phototransistors: the latter two components are true semiconductor devices, while a photo resistor is a passive component and has no junction PN. In addition, the photoresistance of any photoresistor can vary widely depending on the ambient temperature, rendering them inappropriate for applications that require accurate measurement or sensitivity to light photons.

Pressure sensor: it has many other names: pressure transducer, pressure indicator, pressure sender, pressure transmitter, piezometer and manometer, among other names.

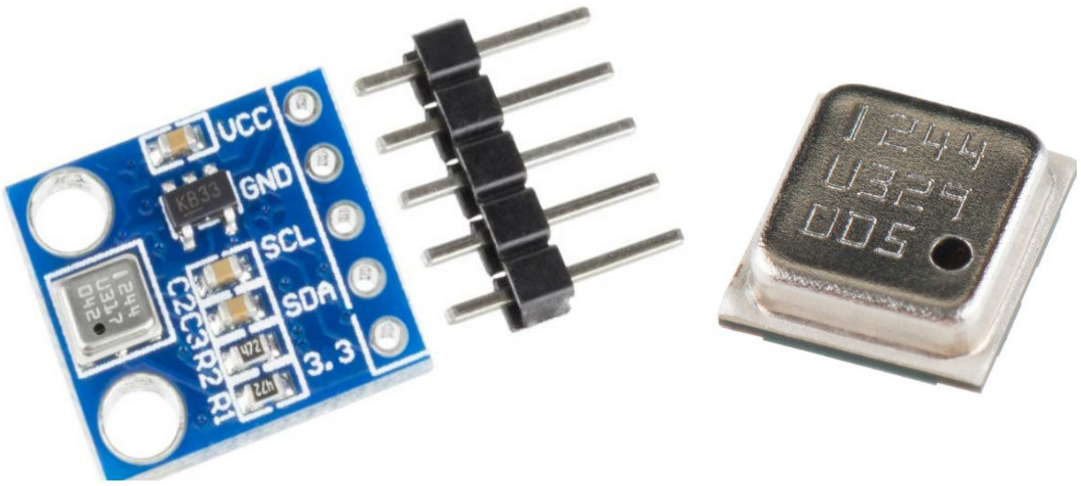


Fig 3.6 BMP180 sensor.

We can generally define the pressure as the force applied perpendicularly to the surface of an object per unit area over which that force is distributed. It is simply an expression of the force required to stop a fluid from expanding.

The SI (International System of Units) unit for pressure is Pa (the Pascal), is equal to one newton per square meter. But there are other units used to express pressure such as pounds per square inch and bar. Some meteorologists prefer the hPa (hectopascal) for atmospheric air pressure, which is equivalent to the older unit mBar (millibar).

Pressure sensor is a device specialized in the pressure measurement of liquids or gases. This device is designed to convert pressure variations into electrical voltage variations.

In our project, we will use the BMP180 sensor, based on piezo-resistive technology for EMC robustness, high accuracy and linearity as well as long-term stability.

The ultra-low power, low voltage electronics of the BMP180 is optimized for use in PDAs (Personal Digital Assistant devices), GPS (Global Positioning System) navigation devices and outdoor equipment. The I2C interface allows for easy system integration with a microcontroller.

3.2.3. The actuators

The actuators are defined as components of a machine that are responsible for moving and controlling a mechanism or system. Simply, they are the movers.

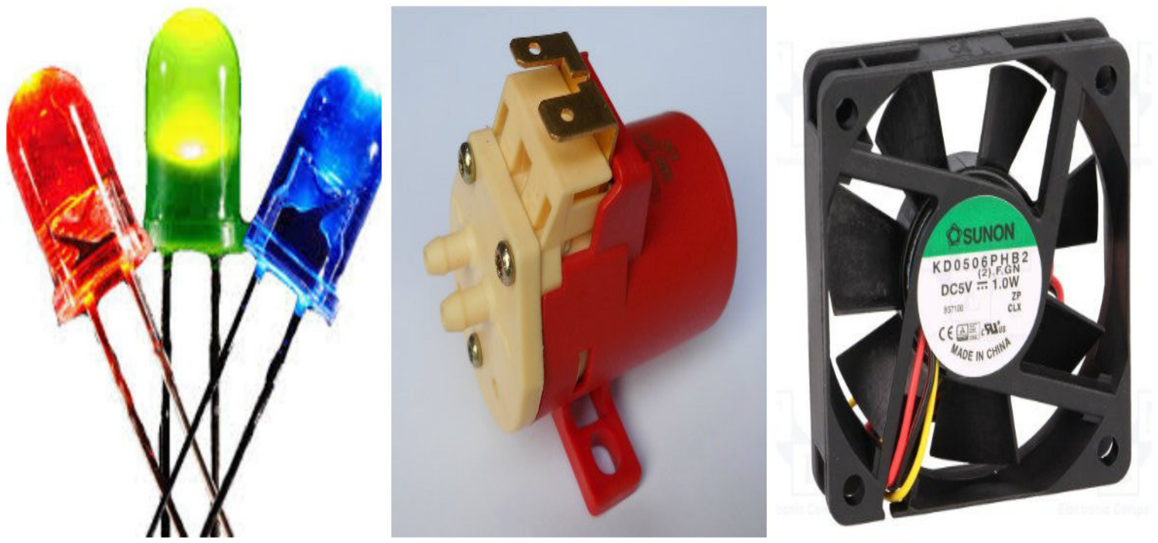


Fig 3.7 The actuators.

In our project, we need some actuators as follows:

- **A water pump:** we need a water pump to use it in irrigation. the water pump that we use is 12v supplied, it is a wiper water pump, but we will change its use to the irrigation and watering of our plants in the greenhouse.
- **A ventilator:** the ventilator is an actuator used for the ventilation and cooling, we will use a 5v ventilator that is used generally for cooling the central unit of the desktop PC in our project to ventilate the greenhouse when there is some unsuitable heat.
- **LEDs:** we will use some LEDs as light signals.
- **AC Actuators:** in the reality, we can change our DC actuators in our prototype to AC actuators such as AC lamp, AC ventilator and AC water pump. We just need to change the transistor, the optocoupler and motor driver to AC relays.

3.2.4. Other hardware requirements

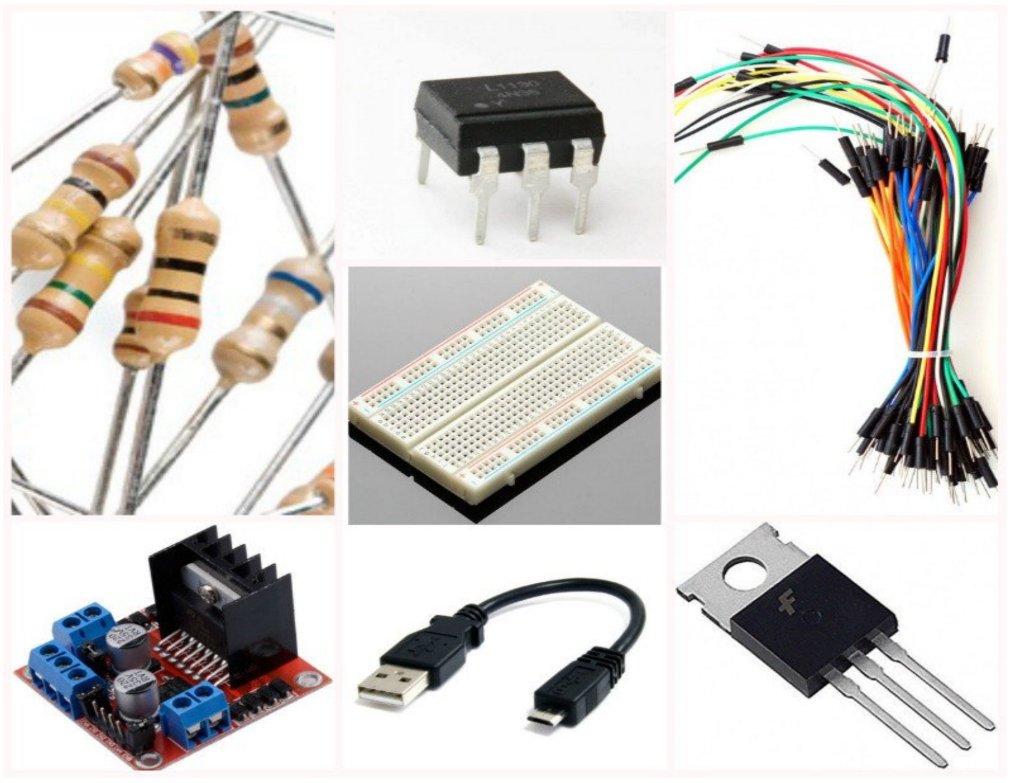


Fig 3.8 Other hardware requirements.

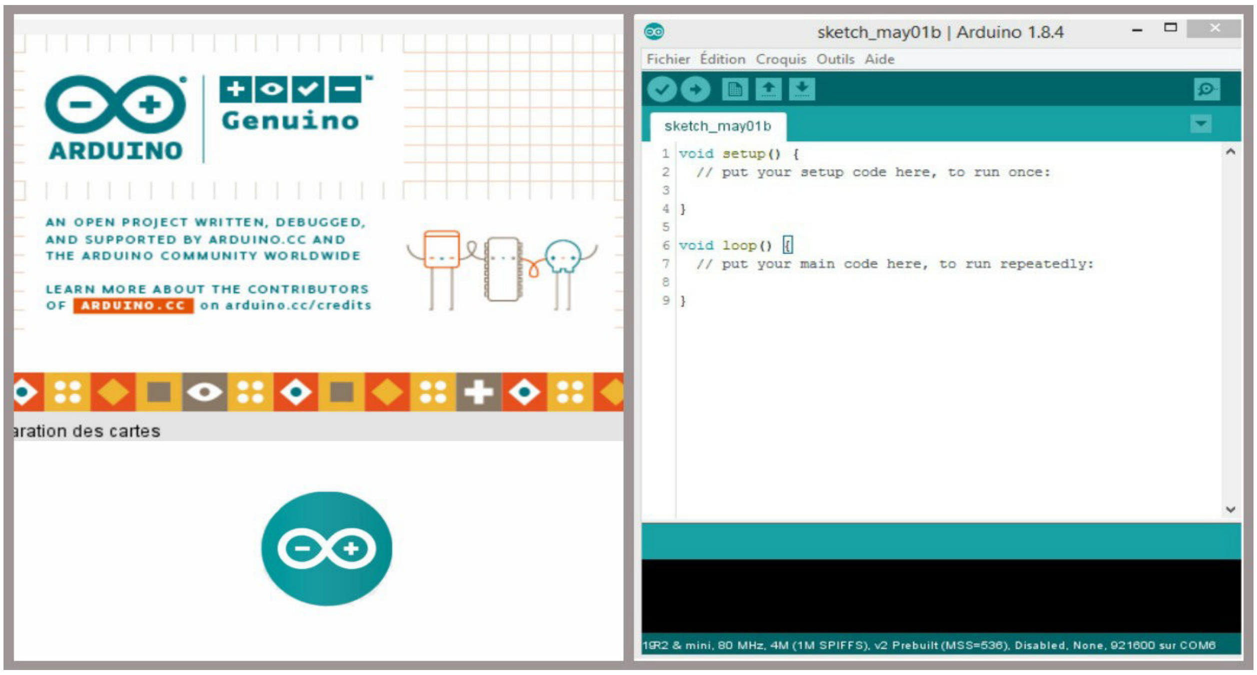
- **Breadboards:** we need two breadboards, one for each ESP8266 module, to give each module an independence of the other, in the professional and commercial projects the breadboards and the jumper wires will be changed by PCBs.
- **Jumper wires:** the jumper wires are considerables in our project; they are the links between different electronic devices.
- **The resistors:** we need the resistors for many applications in our project, to make a voltage divider, to read the amount of light from the LDR, pull up resistor for the DHT11 … etc. we need multiple resistor values, each value having a specific color coding.
- **USB cables:** we need two USB 2.0 - Micro-USB to USB Cables, they are perfect to connect the ESP8266 modules to the PC or power banks.
- **L298N:** motor driver, motor controller or motor shield is a little current amplifier; the function of motor controllers is to take a low-

current control signal and then turn it into a higher-current signal that can drive a motor, the L298N H-bridge module is a DC motor shield allows our ESP8266 module to drive two channel DC motors. We will use it to drive the ventilator.

- **BD911:** the BD911 is a silicon Epitaxial-Base NPN power transistor; it is intended for use in power linear and switching applications. We will use it to control the water pump.
- **4n35**: it is an optocoupler or photocoupler, phototransistor output, with base connection. It is a component that transfers electrical signals between two isolated circuits by using light, and it consists of gallium arsenide infrared LED and a silicon NPN phototransistor. We will use it with the transistor for more isolation between the water pump and the ESP8266 module.
- **AC Relays:** we can use the AC relays instead of the transistor and motor driver when the actuators are AC devices. The relays are used to separate the high voltage zone (the actuators) from the low voltage zone (the microcontrollers).

3.3. Software requirements

3.3.1. Arduino IDE


ARDUINO
Genuino
sketch_may01b | Arduino 1.8.4
Fichier Édition Croquis Outils Aide
sketch_may01b
aration des cartes

AN OPEN PROJECT WRITTEN, DEBUGGED, AND SUPPORTED BY ARDUINO.CC AND THE ARDUINO COMMUNITY WORLDWIDE

LEARN MORE ABOUT THE CONTRIBUTORS OF ARDUINO.CC on arduino.cc/credits

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

1GR2 & mini, 80 MHz, 4M (1M SPIFFS), v2 Prebuilt (MSS=536), Disabled, None, 921600 sur COM6

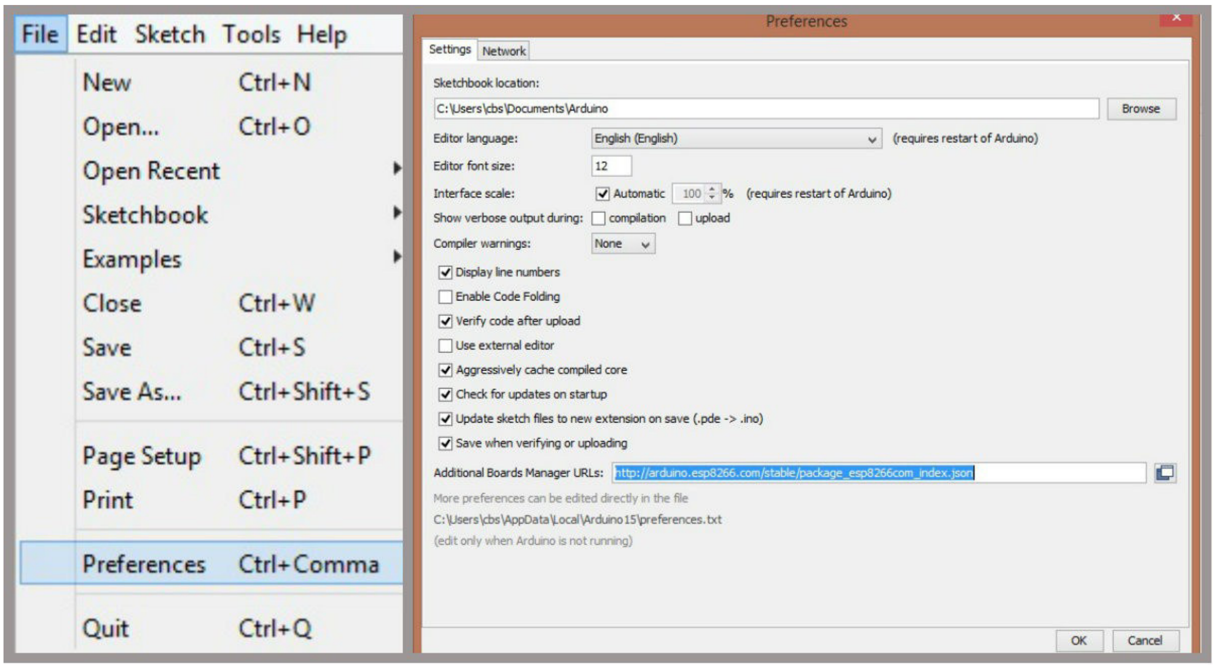
Fig 3.9 Arduino IDE.

Arduino IDE[1] or Arduino Software is a cross-platform[2] application that is written in the programing language Java and based on Processing and other open-source software; it runs on Windows, Linux, and Mac OS X. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino software can be used with an Arduino board and even other boards such as the 32U4 family of microcontrollers and esp8266 boards.

In our project, we will use the Arduino software for programming the ESP8266 boards. However, to use it well with the ESP8266 boards we must pass on a few more steps first: after installing the newer version of Arduino IDE, we should add the package of the ESP8266 by going to “File”, selecting “ Preferences” and entering this link under Additional boards manager(http://arduino.esp8266.com/stable/package_esp8266com_index.json).

1 **IDE:**(Integrated Development Environment) is a software application, which provides comprehensive facilities to computer programmers for software development

2 **Cross-Platform:** called also multi-platform software or platform-independent software, it is a computer software that is implemented on multiple computing platforms



Sketchbook location: C:\Users\cbs\Documents\Arduino

Editor language: English (English) (requires restart of Arduino)

Editor font size: 12

Interface scale: ☑ Automatic 100 % (requires restart of Arduino)

Show verbose output during: ☐ compilation ☐ upload

Compiler warnings: None

☑ Display line numbers

☐ Enable Code Folding

☑ Verify code after upload

☐ Use external editor

☑ Aggressively cache compiled core

☑ Check for updates on startup

☑ Update sketch files to new extension on save (.pde -> .ino)

☑ Save when verifying or uploading

Additional Boards Manager URLs: http://arduino.esp8266.com/stable/package_esp8266com_index.json

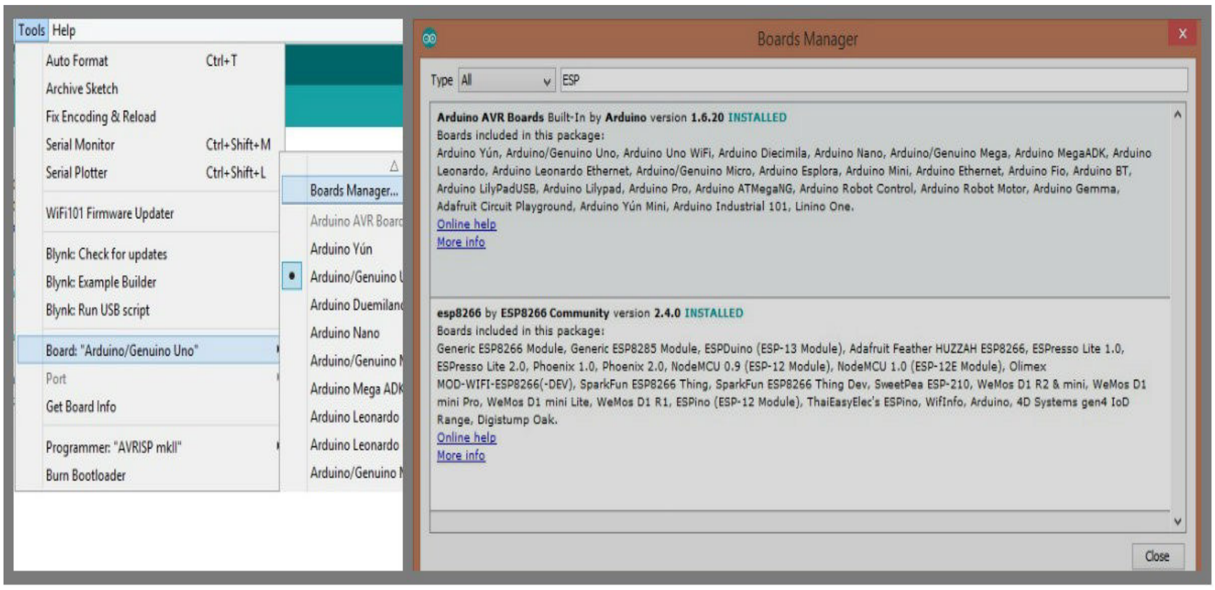
More preferences can be edited directly in the file

C:\Users\cbs\AppData\Local\Arduino15\preferences.txt

(edit only when Arduino is not running)

Fig 3.10 Installing the package.

Then, go to “Tools”, select “Boards” “Boards manager”, find ESP8266and press Install button.



Type All ESP

Arduino AVR Boards Built-In by **Arduino** version **1.6.20** INSTALLED
Boards included in this package:
Arduino Yún, Arduino/Genuino Uno, Arduino Uno WiFi, Arduino Diecimila, Arduino Nano, Arduino/Genuino Mega, Arduino MegaADK, Arduino Leonardo, Arduino Leonardo Ethernet, Arduino/Genuino Micro, Arduino Esplora, Arduino Mini, Arduino Ethernet, Arduino Fio, Arduino BT, Arduino LilyPadUSB, Arduino Lilypad, Arduino Pro, Arduino ATMegaNG, Arduino Robot Control, Arduino Robot Motor, Arduino Gemma, Adafruit Circuit Playground, Arduino Yún Mini, Arduino Industrial 101, Linino One.
Online help
More info

esp8266 by **ESP8266 Community** version **2.4.0** INSTALLED
Boards included in this package:
Generic ESP8266 Module, Generic ESP8285 Module, ESPDuino (ESP-13 Module), Adafruit Feather HUZZAH ESP8266, ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 2.0, NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Olimex MOD-WIFI-ESP8266(-DEV), SparkFun ESP8266 Thing, SparkFun ESP8266 Thing Dev, SweetPea ESP-210, WeMos D1 R2 & mini, WeMos D1 mini Pro, WeMos D1 mini Lite, WeMos D1 R1, ESPino (ESP-12 Module), ThaiEasyElec's ESPino, WifInfo, Arduino, 4D Systems gen4 IoD Range, Digistump Oak.
Online help
More info

Fig 3.11 Installing the boards’ package.

After these steps, we are ready to program our ESP8266 boards by selecting the right board’s name from “Tools” “Boards”, “Wemos D1 R2 &

mini” when we use Wemos D1 mini board, and “NodeMCU 1.0 (ESP-12E module)” when we use our Node MCU board.

Some of our modules such as DHT11 and BMP180 may need “libraries” to work. Libraries are collections of software functions geared towards a single purpose, such as communicating with a specific device.

3.3.2. Hosting platforms

- **Dweet.io:** this website will be our first connection with the internet; we will post our data to this site to use them later by other dashboards. Posting data to dweet.io is very simple, it is done by calling a URL: (https://dweet.io/dweet/for/my-thing-name?hello=world) and changing “my-thing-name” to the name of our thing, “hello” to the name of the parameter that we are posting and “world” to the value of this parameter.
- **Freeboard.io:** One of the main reasons why of posting sensor data online is so that we can easily monitor the data over time. This work will be done by the cloud dashboards, which take the posted data and convert it to meaningful graphical presentations. The use of the graphical representations provided by the *dweet.io* have a small barrier is that we do not have any other options for the presentation of the posted data. The solution is by using the *Freeboard.io* dashboard that is friendly with the *dweet.io*.
- **Google drive:** we need account in google drive to send data into its google spreadsheet and store these data for a long time. We are using Google Apps Scripts as “bridge service” in order to redirect data from ESP8266 into Google Spreadsheet.

3.4. Conclusion

In this chapter, we have seen all what we need for our project, and we have classified the needs into two classes: hardware requirements where we have put our tangible needs such as sensors, actuators … etc., and software requirements where we have put our software needs such as the host platforms … etc.

The next chapter will be dedicated to discuss the practical work.

4.1. Introduction

The best way to understand the real meaning of the Internet of Things using the ESP8266 module is the practice. An Internet of things project using the ESP8266 will show us how the ESP8266 modules work and how easy they are.

Our project dedicated to building an automated greenhouse project based on the ESP8266. It will allow us to monitor some parameters such as temperature, humidity, pressure and the light amount inside the greenhouse, in addition to the amount of moisture in the soil. Moreover, it will allow us remotely control certain actuators such as an air ventilator and a water pump that are responsible for air conditioning the greenhouse when the temperature is so high and irrigating the plants when the soil is dry.

4.2. Hardware discussion

To build our greenhouse, we created a wooden frame with a length of 53 cm, a width of 35 cm and a height of 44 cm.These dimensions are related to the dimensions of our agricultural basin.

We covered this frame with a piece of thin transparent plastic using metal pins.

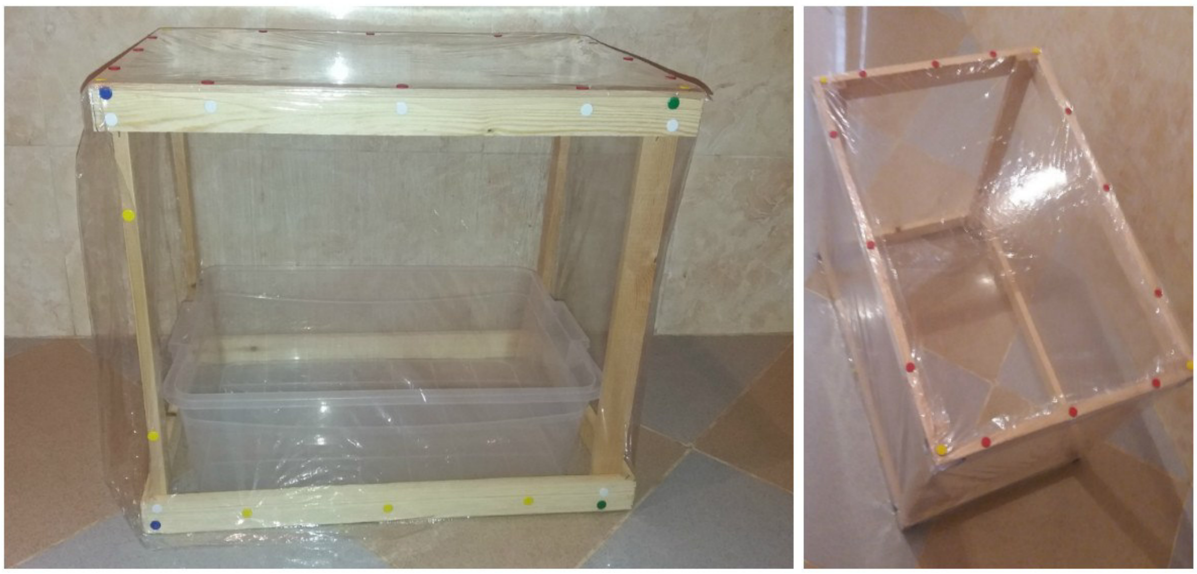


Fig 4.1 The cover of the greenhouse.

Our agricultural basin is transparent box of 17 liters generally used as box of bread. Here where we will plant our seedlings. The basin handles will be used as placements for devices.

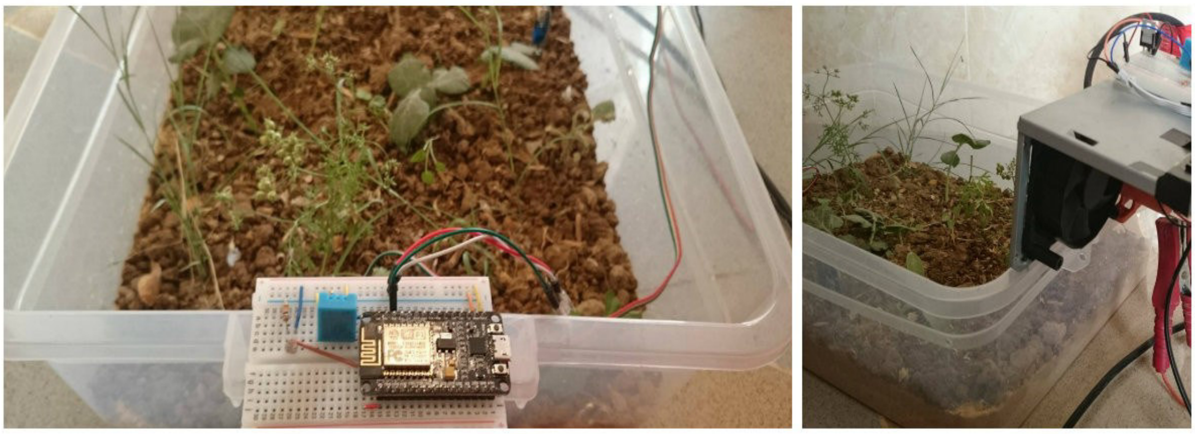


Fig 4.2 The basin of the greenhouse.

Since we have two ESP8266 modules, the hardware setup is divided into two parts, one for each module. We will name each part by the name of its ESP8266 module to facilitate things.

4.2.1. NodeMCU part

This part will be inside the green house (on one of the handles), it is mainly dedicated to the sensing (**Fig 4.3**). In addition to the NodeMCU (1), we used four sensors: DHT11 (2), BMP180 (3), Soil moisture sensor (4) and LDR (5).

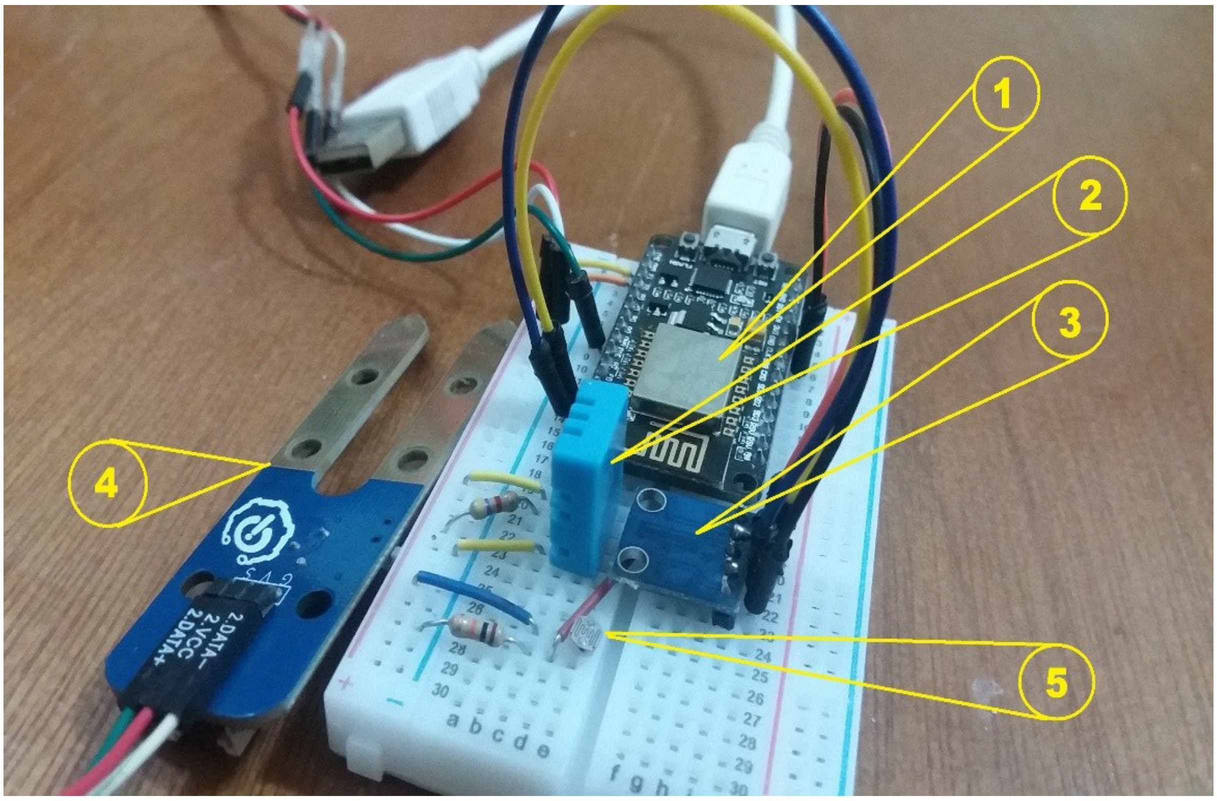


Fig 4.3 Hardware setup of NodeMCU part.

The wiring of the NodeMCU part is illustrated in (**Fig 4.4**). The SCL and SDA pins of the BMP180 module should be connected to I2C pins (D1 and D2, respectively). The signal pins of the soil moisture sensor and DHT11 should be connected to PWM pins (D5 and D3, respectively).

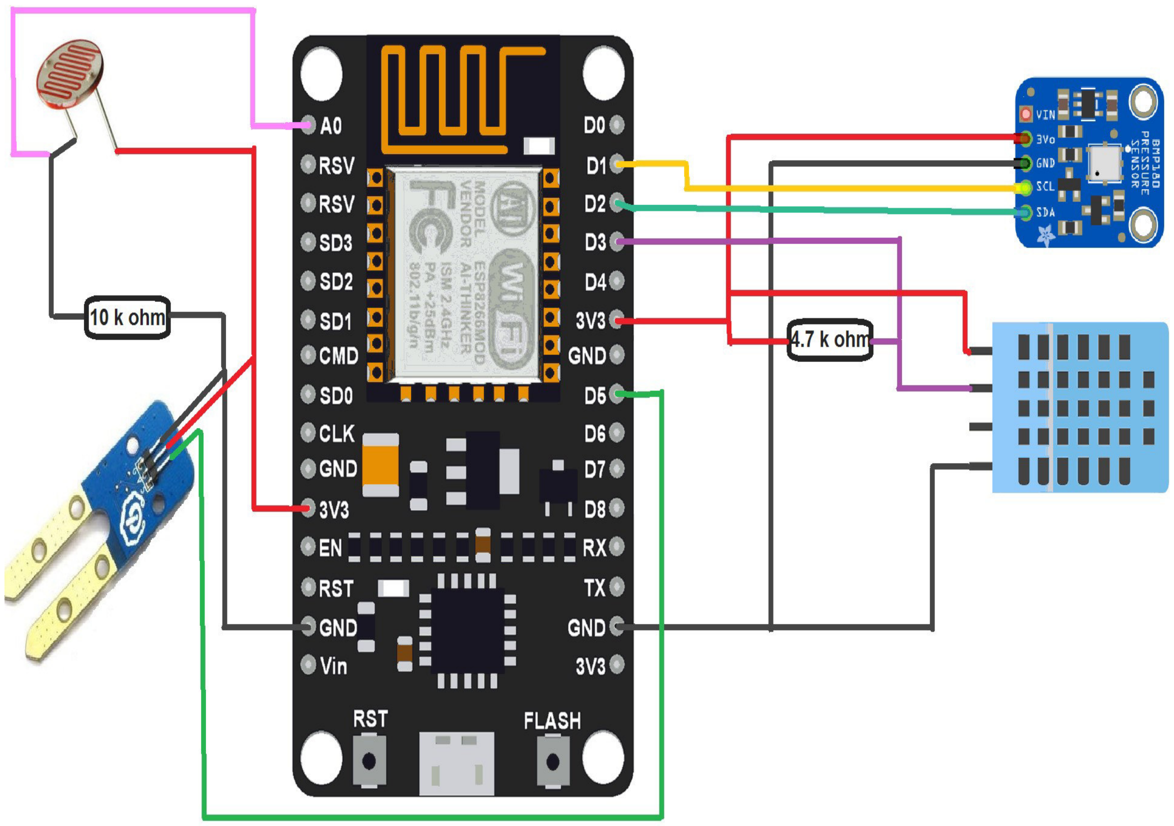


Fig 4.4 The wiring of NodeMCU part.

4.2.2. Wemos part

This part will be outside the green house (on the other handle); the Wemos D1 MINI is responsible for driving the actuators with the help of preactuators. In addition to the Wemos D1 MINI, we used two actuators, air ventilator and water pump. We used also motor driver L298N, BD911NPN transistor, diode, and 4n35 optocoupler.

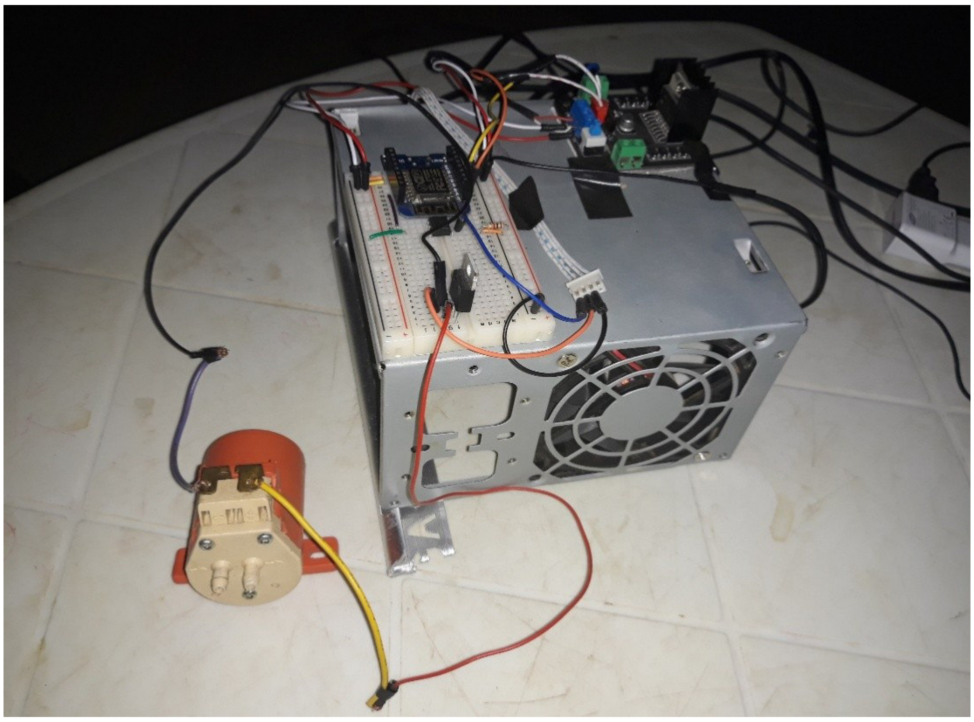


Fig 4.5 Hardware setup of the Wemos part.

The (**Fig 4.6**) illustrates the wiring of Wemos part. The WeMos controls the air ventilator through the pins D5 and D6, and controls the water pump through the pin D2.

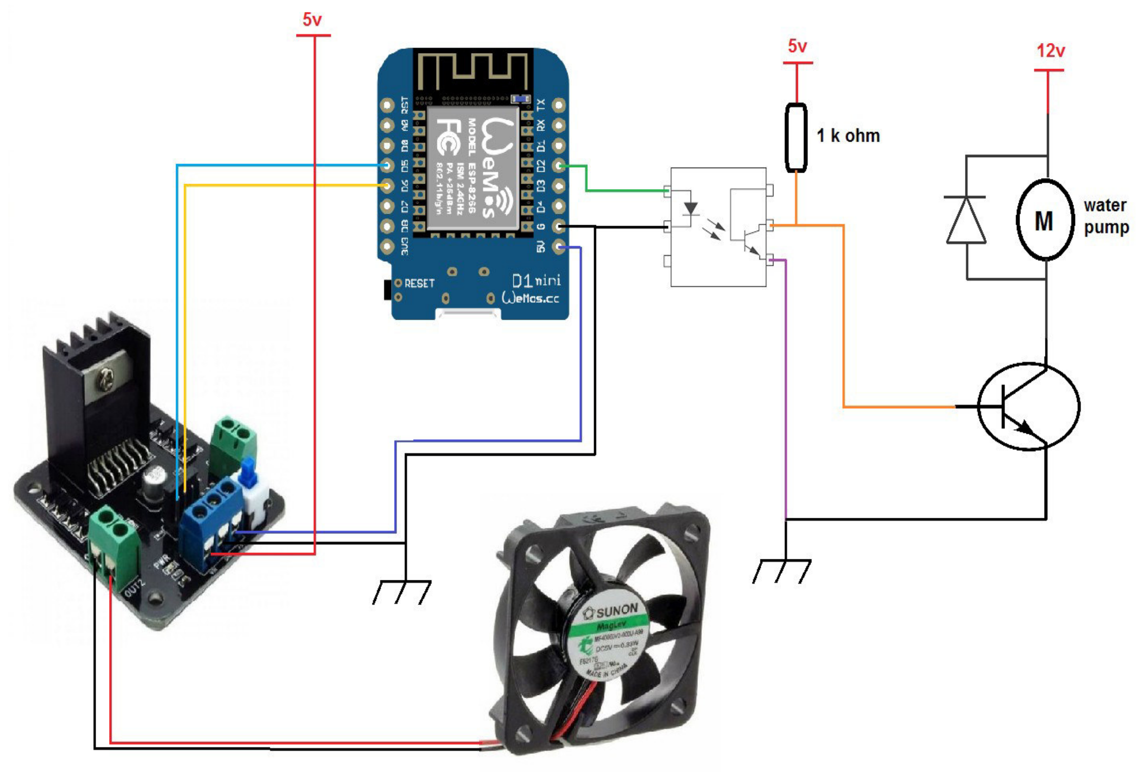


Fig 4.6 The wiring of Wemos part.

The (**Fig 4.6**) presents our prototype's wiring; in the reality, generally we use the AC devices such as the AC lamp, AC water pump and AC ventilator or electrical gate using AC motor. This is what forces us to separate definitively the control zone from the actuators zone.

We can use the AC relays to separate the control zone from the actuators zone. The wiring of the Wemos partusing the relays is described in (**Fig 4.7**), we just need an AC relay in the place of the motor shield and the NPN transistor.

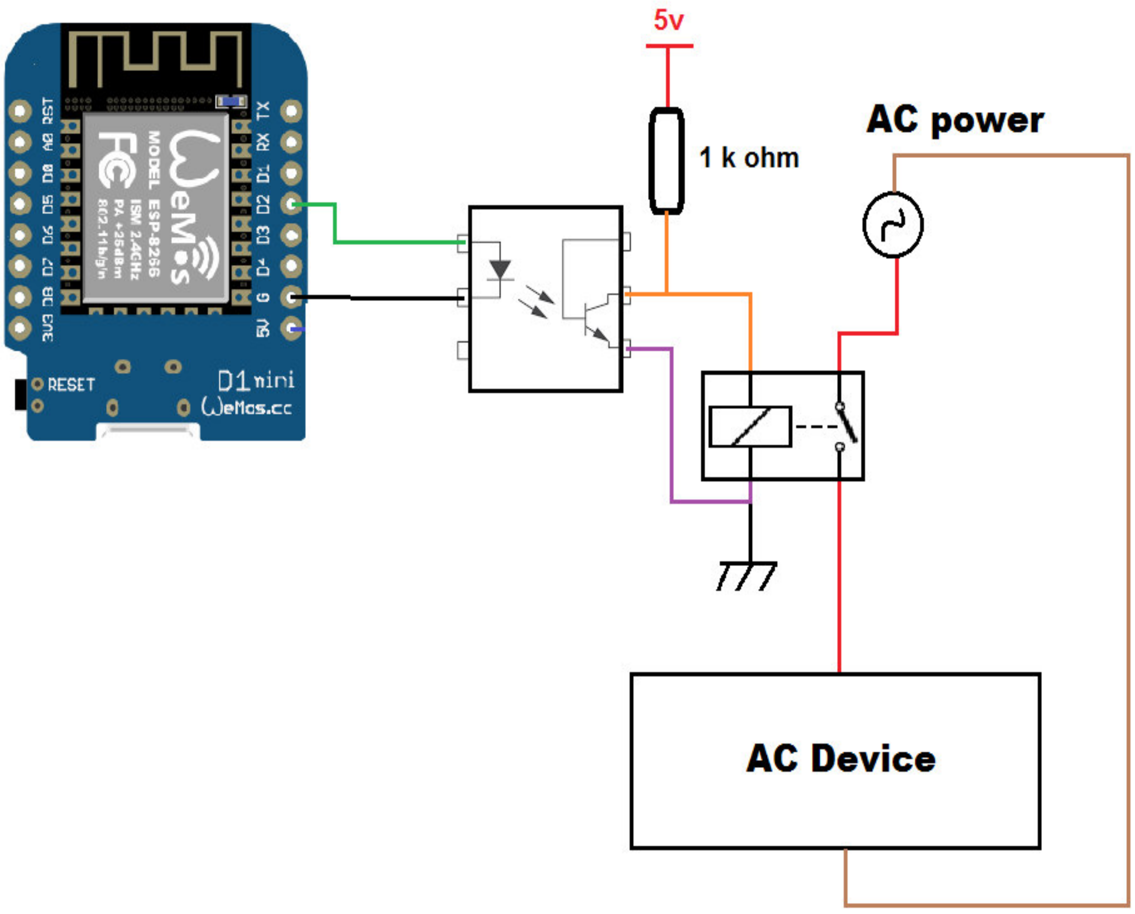


Fig 4.7 The wiring of Wemos part using the relay.

4.3. Software discussion

The following chart shows us a general view of how our system works (**Fig 4.8**):

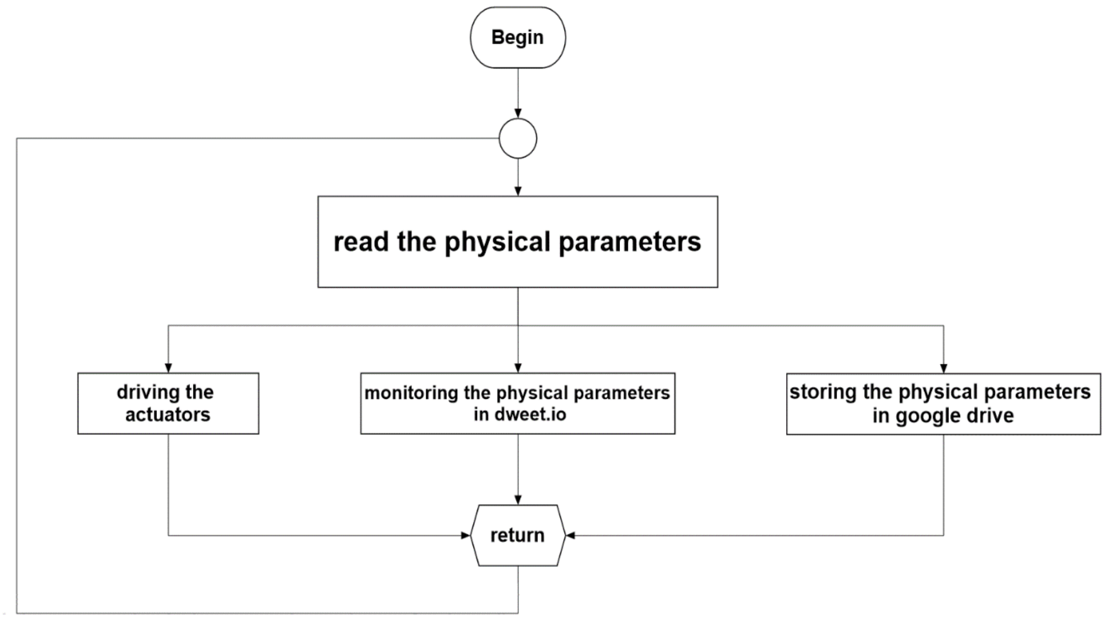
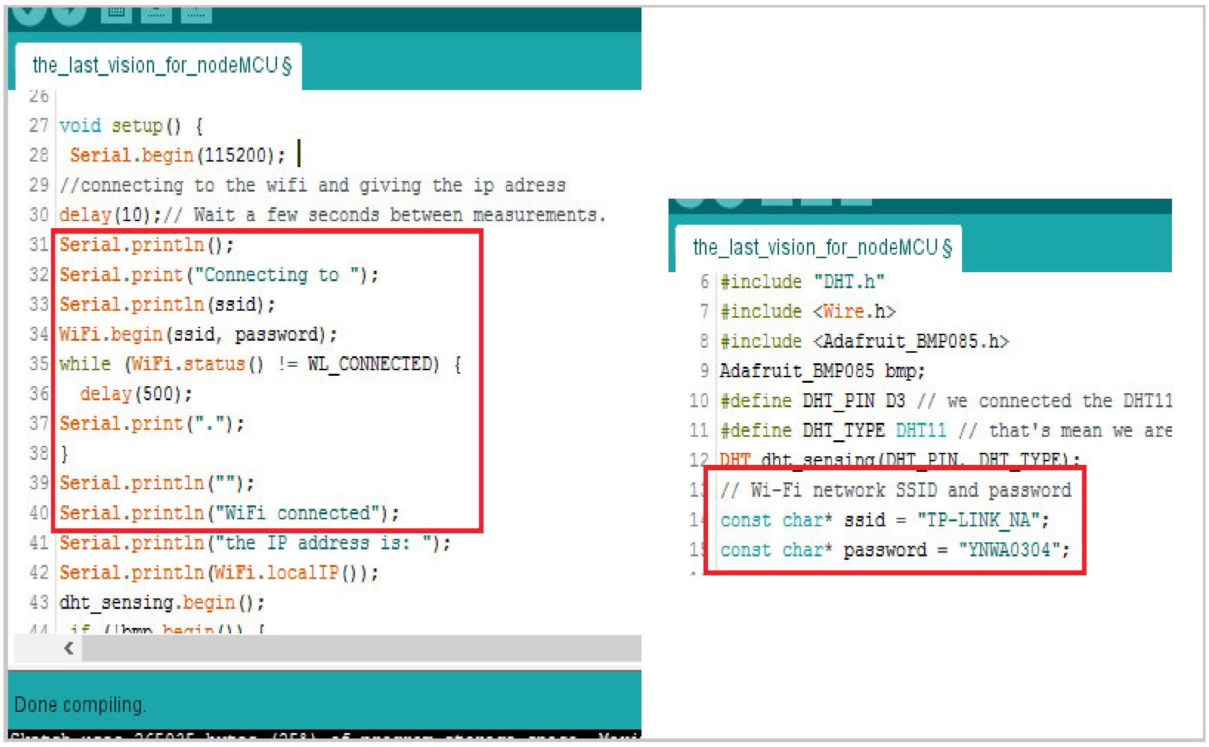

Begin
read the physical parameters
driving the actuators
monitoring the physical parameters in dweet.io
storing the physical parameters in google drive
return

Fig 4.8 General operation of the system.

4.3.1. Connect to the access point



```
void setup() {
 Serial.begin(115200);
//connecting to the wifi and giving the ip adress
delay(10);// Wait a few seconds between measurements.
Serial.println();
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");
Serial.println("the IP address is: ");
Serial.println(WiFi.localIP());
dht_sensing.begin();
 if (!bmp.begin()) {
```

```
#include "DHT.h"
#include <Wire.h>
#include <Adafruit_BMP085.h>
Adafruit_BMP085 bmp;
#define DHT_PIN D3 // we connected the DHT11
#define DHT_TYPE DHT11 // that's mean we are
DHT dht_sensing(DHT_PIN, DHT_TYPE);
// Wi-Fi network SSID and password
const char* ssid = "TP-LINK_NA";
const char* password = "YNWA0304";
```

Fig 4.9 Connect to the access point.

To connect our ESP8266 modules to the internet, we used <ESP8266WiFi.h> library functions. The functions of this library allow us to connect the ESP8266 modules to local access point such as the router. We need to access the SSID (Service Set Identifier) and the password of the router or access point that we use.

4.3.2. Communicating between the two ESP8266 modules

We tried to communicate between our ESP8266s to connect the two parts to each other. To do that we chose a local M2M with TCP communication.

To do a M2M communication between our two ESP8266s, they must be connected to the same access point. We programmed our Wemos as a server and our NodeMCU as a client using the function of the <ESP8266WiFi.h> library. The client triggers the communication asking the server and the server will send an answer. The access point will give the Wemos an IP address during the connection. We have to put that IP address in the client program to connect to the server through it.

The client will send the temperature value and the status of soil through a TCP URL, the server will receive this URL and compare the value of the temperature with a threshold value stored in its memory, then it decides to running the actuators or not.

4.3.3. Displaying data through the internet

- **Connecting to *Dweet.io* server**

Our first station for displaying data through the internet is *dweet.io* host server. We will send an URL to the website "*dweet.io*" containing the values of our physical parameters and the status of our actuators, and then the site will read the URL, export our values from it, and display them.

We used dweet.io because it is free allows other dashboards such as *Freeboard.io* to use its database, in addition to the simplicity of post data in it, and we do not have to sign in.

Both Wemos and NodeMCU will connect to *Dweet.io*, but each one will connect independently, that means each module will have an independent thing's name.

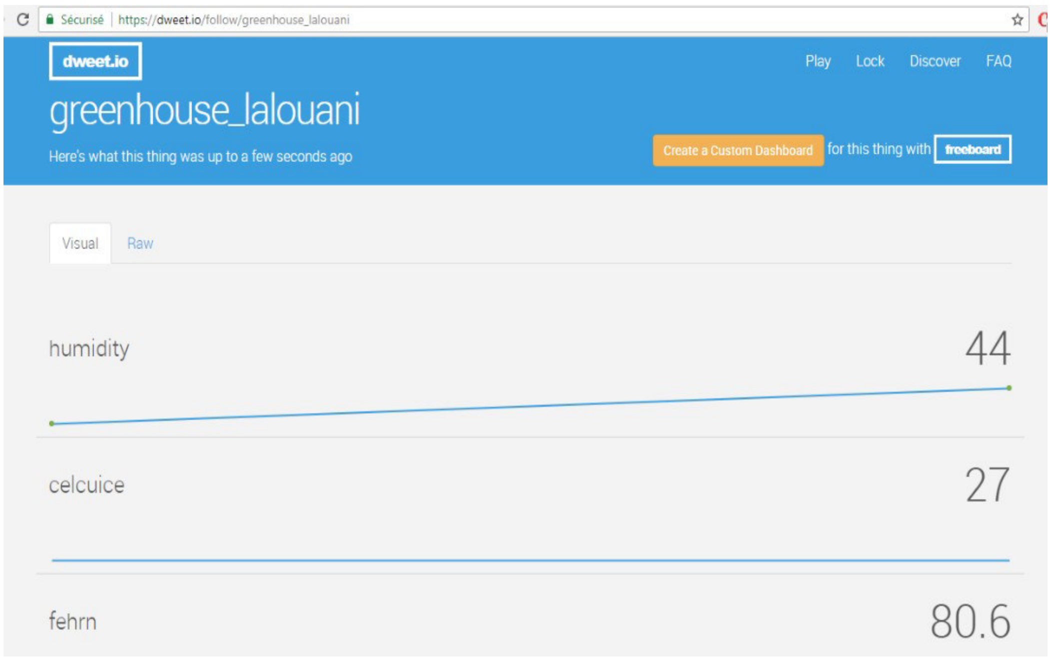


Fig 4.10 Displaying data on dweet.io.

- **Displaying the data on *Freeboard.io***

We will use *Freeboard.io* for two reasons, for taking advantage of its charts, and for collecting the two parameters (the physical parameters and the status of our actuators) and displaying them on one page.

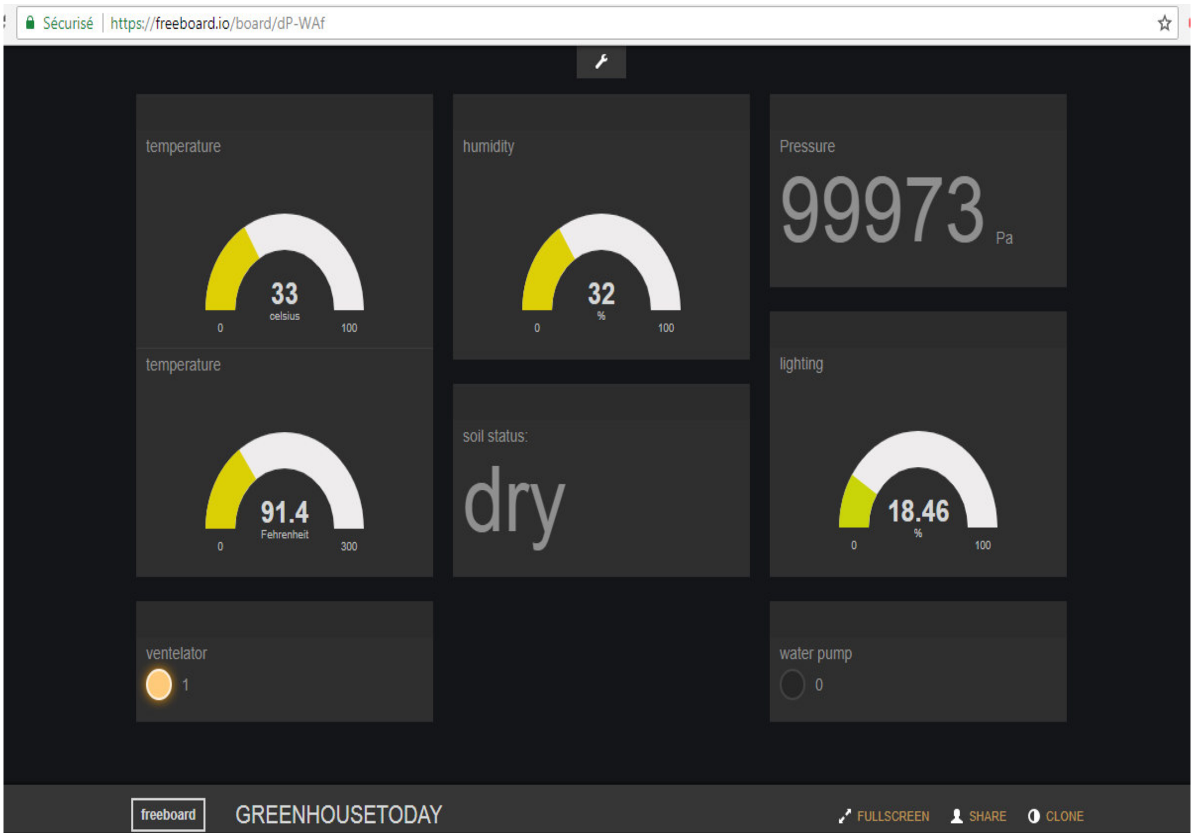


Fig 4.11 Displaying data to Feeboard.io.

4.3.4. Storing data through the internet

We tried to send data from our NodeMCU into Google Drive or Google Spreadsheet without using the third-party services such as Temboo, Thingspeak ... etc. Instead, we used Google Apps Scripts as a "bridge service" in order to redirect ESP8266 data to Google Spreadsheet.

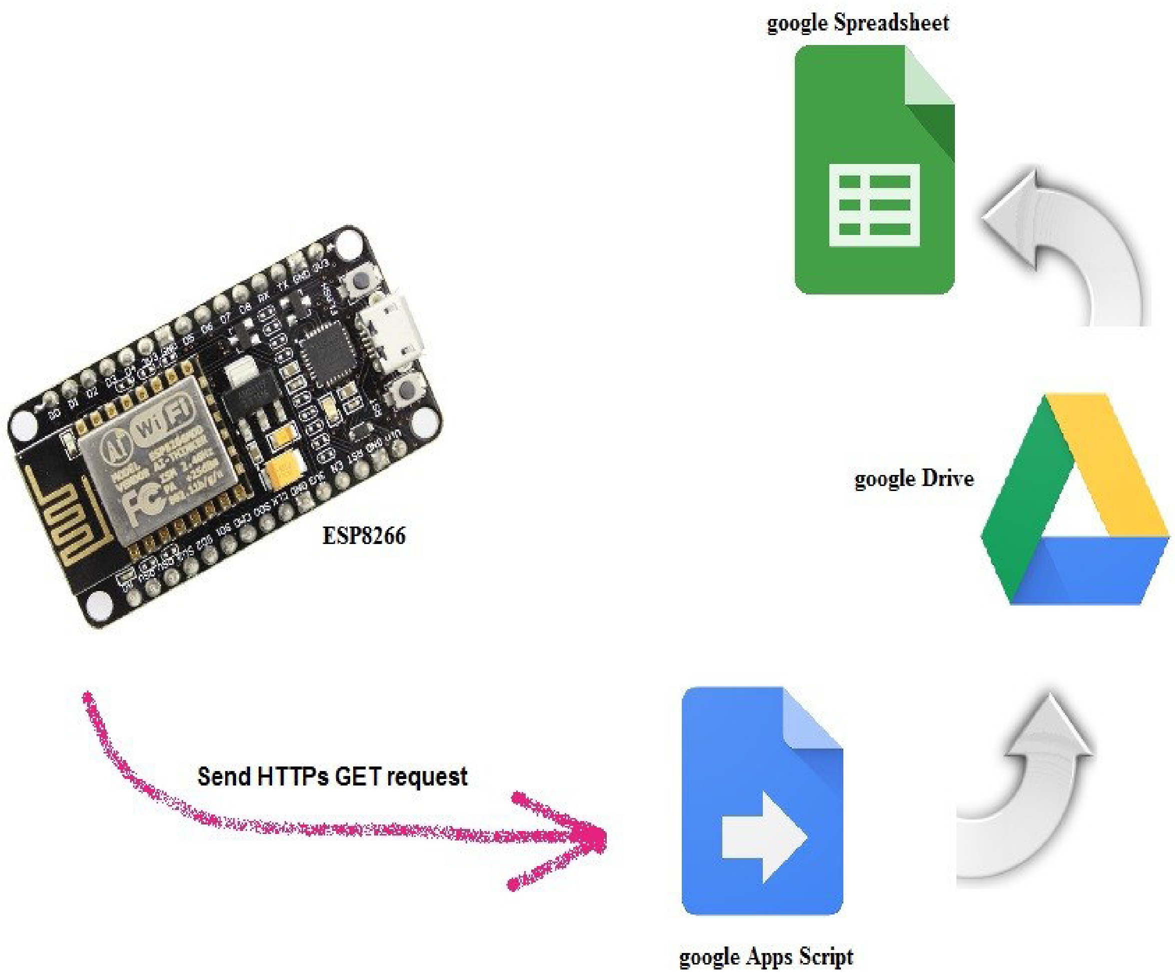
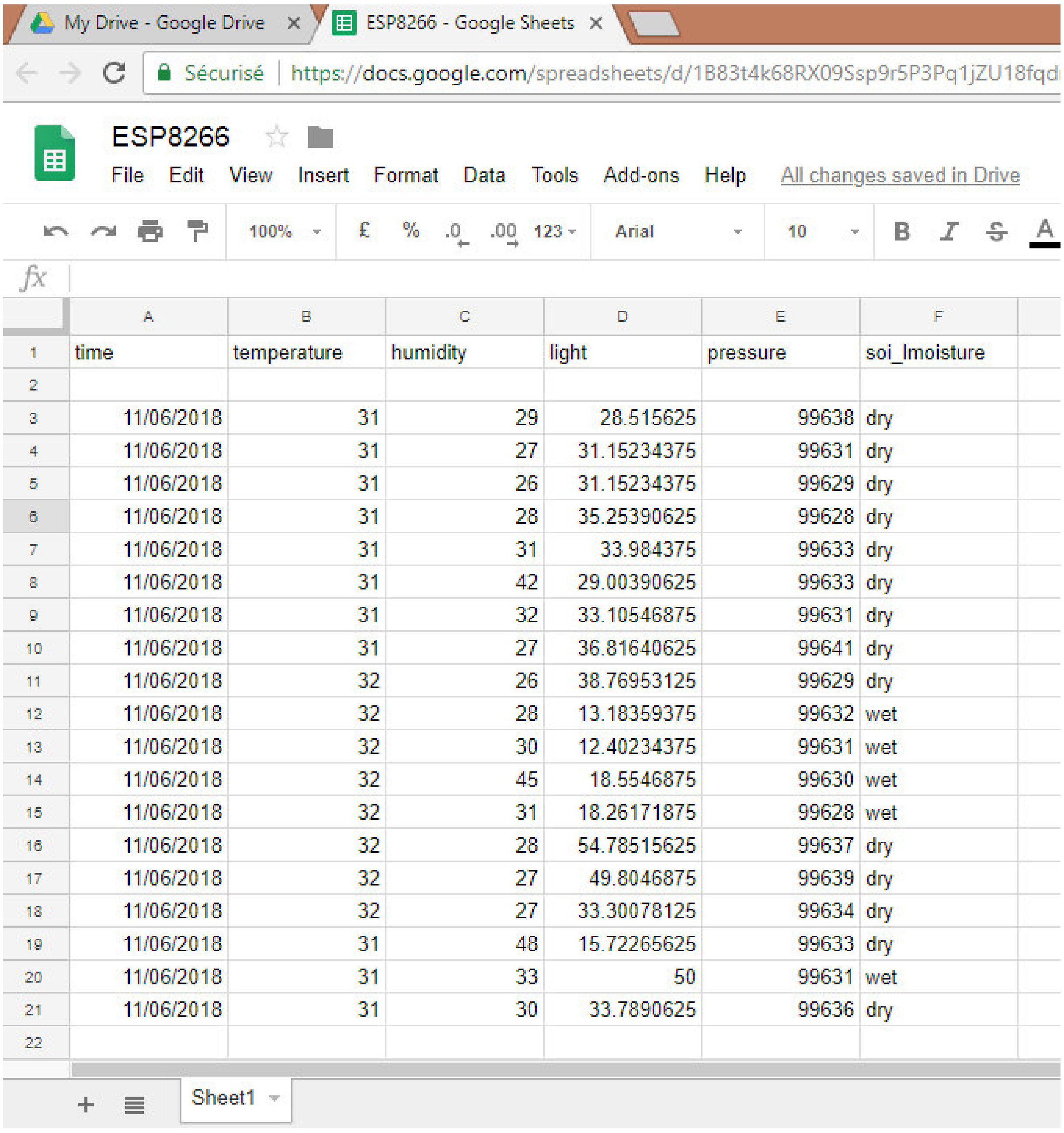


Fig 4.12 Storing data on google spreadsheet.

All we need is a GAS ID (Google Apps Scripts Identification) from Google Apps Scripts to send data into our own spreadsheet.

We used the functions of the library <WiFiClientSecure.h> to connect our NodeMCU to Google Apps Scripts.


My Drive - Google Drive
ESP8266 - Google Sheets
Sécurisé
https://docs.google.com/spreadsheets/d/1B83t4k68RX09Ssp9r5P3Pq1jZU18fqd
ESP8266
File Edit View Insert Format Data Tools Add-ons Help
All changes saved in Drive
100%
Arial
10
Sheet1

time	temperature	humidity	light	pressure	soi_lmoisture
11/06/2018	31	29	28.515625	99638	dry
11/06/2018	31	27	31.15234375	99631	dry
11/06/2018	31	26	31.15234375	99629	dry
11/06/2018	31	28	35.25390625	99628	dry
11/06/2018	31	31	33.984375	99633	dry
11/06/2018	31	42	29.00390625	99633	dry
11/06/2018	31	32	33.10546875	99631	dry
11/06/2018	31	27	36.81640625	99641	dry
11/06/2018	32	26	38.76953125	99629	dry
11/06/2018	32	28	13.18359375	99632	wet
11/06/2018	32	30	12.40234375	99631	wet
11/06/2018	32	45	18.5546875	99630	wet
11/06/2018	32	31	18.26171875	99628	wet
11/06/2018	32	28	54.78515625	99637	dry
11/06/2018	32	27	49.8046875	99639	dry
11/06/2018	32	27	33.30078125	99634	dry
11/06/2018	31	48	15.72265625	99633	dry
11/06/2018	31	33	50	99631	wet
11/06/2018	31	30	33.7890625	99636	dry

Fig 4.13 Google spreadsheet.

4.4.Conclusion

In this chapter, we have presented the important stations in our realization, and we discussed the hardware and software installation needed to build a smart and wireless greenhouse project.

Conclusion

About 20 years ago, Kevin Ashton, one of British technology pioneers, has brought out into the light a concept that has accelerated the development wheel in the fields of information technology and automation.

We can say that the internet of things, which is considered the next wave in the era of computing, is the trend of everyone these days, despite of the fact that it is just in the first steps of development and have a lot of potential development. Especially after the emergence of the ESP8266 modules.

In the past couple years, the ESP8266 platform has flourished dramatically and emerged as one of the most popular hardware tools among electronics hobbyists and IoT enthusiasts. Now, The ESP8266 is the most integrated and affordable Wi-Fi solution available in the current IoT market space.

This end of studies' project wanted to give an overview on the Internet of Things and its projects using the ESP8266 modules.

In this work that consists of four chapters, we tried to get closer to the Internet of Things and its world, and we investigated the origin of its term, how it works, its architecture and its most protocols. Then, we identified the wireless modules used in IoT projects, where we focused more on the ESP8266 modules.

Finally, we built an IoT project using the ESP8266 modules represented in a smart and wireless greenhouse. As well as a detailed explanation of all our needs in this project.

We got good results at the end of the work, where we have been able to send the reads of the physical parameters and the status of the actuators through the internet to show them in freeboard.io.

In the future we are looking to add other devices and options to the project such as automatic fertilizing and live streaming camera to monitor the green house remotely.

after that point arrived, the next question which needs answers is: how to secure these uploaded information? And what are the best methods to secure them?

Bibliography

[1] SANTUCCI, Gerald. "The Internet of Things: Between the Revolution of the Internet and the Metamorphosis of Objects", *Vision and Challenges for Realizing the Internet of Things,* CERP-IoT (now IERC), March 2010.

[2] CHOKSHI, Aanal; PATEL, Shivam. *Internet of Things (IoT),* IoT Architecture: Security Challenges in IoT & Role of IoT in Healthcare Industry. *International Journal of Engineering Technology Science and Research*, [online]. October 2017, Volume 4, Issue 10, p.822-825. Available on: http://ijetsr.com/images/short_pdf/1509009166_822-825-mccia850_ijetsr.pdf (accessed on: 20/6/2018).

[3] Lopez Research. *An Introduction to the Internet of Things (IoT). Part 1. Of "The IoT Series"*. November 2013. San Francisco: Lopez Research.

[4] RAYES, Ammar; SALAM, Samer. *Internet of Things-From Hype to Reality*: *The Road to Digitization*. Springer. Switzerland. 2017. 350p. (ISBN 978-3-319-44860-2).

[5] UCKELMANN, Dieter; HARRISON, Mark; MICHAHELLES, Florian. *Architecting the Internet of Things.* Springer. Switzerland. 2011. 386p. (ISBN 978-3-642-19156-5).

[6] ASHTON, Kevin. that 'internet of thing' thing: In the real world, things matter more than ideas. In: RFID journal [online]. June 22, 2009. Available on: http://www.rfidjournal.com/articles/view?4986. (Accessed on : June 20, 2018)

[7] LEWIS, James. Add wireless with one of these five wireless modules. In: baldengineer.com [online]. (Posted on Wednesday, May 20, 2015). Available on: https://www.baldengineer.com/five-wireless-modules-for wireless-projects.html (Accessed on: June 1, 2018).

[8] Rouse, Margaret. Bluetooth Low Energy (Bluetooth LE). In: Techtarget Network [online]. (Updated in November 2014). Available on: http://internetofthingsagenda.techtarget.com/definition/Bluetooth-Low-Energy-Bluetooth-LE. (Accessed on: June 1, 2018).

[9] WIKIPEDIA. Bluetooth Low Energy [online]. *Wikipedia.org.* (edited on 12 June 2018). Available on: https://en.wikipedia.org/wiki/Bluetooth_Low_Energy (Accessed on: June 15, 2018).

[10] NORDIC SEMICONDUCTOR: Preliminary Product Specification. nRF24L01. (edited on March 2006) Available on: https://www.sparkfun.com/ datasheets/Components/nRF24L01_prelim_prod_spec_1_2.pdf (Accessed on: June 1, 2018).

[11] NORDIC SEMICONDUCTOR: *nRF24L01*. (edited on March 2005) Available on: https://www.nordicsemi.com/eng/Products/2.4GHz-RF/nRF24L01P (Accessed on: May 1, 2018).

[12] ARDUINO. Arduino Wifi Shield. *Arduino.cc.* Available on: https://store.arduino.cc/arduino-wifi-shield (Accessed on: May 23, 2018).

[13] WIKIPEDIA. ESP8266 [online]. *Wikipedia.org.* (edited on 13 June 2018). Available on: https://fr.wikipedia.org/wiki/ESP8266.

(Accessed on: June 15, 2018).

[14] WIKIPEDIA. ESP8266 [online]. *Wikipedia.org.* (edited on 13 June 2018). Available on: https://en.wikipedia.org/wiki/ESP8266. (Accessed on: June 15, 2018).

[15] WIKIPEDIA. Frequency band [online]. *Wikipedia.org.* (edited on 19 November 2017). Available on: https://en.wikipedia.org/wiki/Frequency_band. (Accessed on: June 15, 2018).

[16] TEKTRONIX. How to Select your Wi-Fi Module: Application Note [online]. In: tektronix.com/wifi. Available on: https://www.testequity.com/ documents/pdf/applications/wi-fi-module-an.pdf. (Accessed on: June 15, 2018).

[17] WIKIPEDIA. Software development [online]. *Wikipedia.org.* (edited on 17 June 2017). Available on: https://en.wikipedia.org/wiki/Software_development. (Accessed on: June 15, 2018).

[18] WIKIPEDIA. Software development kit [online]. *Wikipedia.org.* (edited on 11 June 2017). Available on: https://en.wikipedia.org/wiki/Software_ development _kit. (Accessed on: June 15, 2018).

ANNEX.A

The NodeMCU’s program:

```
#include <WiFiClientSecure.h>
#include <ESP8266WiFi.h>
#include "DHT.h"
#include <Wire.h>
#include <Adafruit_BMP085.h>
Adafruit_BMP085 bmp;
#define DHT_PIN D3  // we connected the DHT11 with digital pin D1.
#define DHT_TYPE DHT11  // that is mean we are using DHT 11 not dht22.
DHT dht_sensing(DHT_PIN, DHT_TYPE);
// Wi-Fi network SSID and password
const char* ssid = "TP-LINK_NA";
const char* password = "YNWA0304";
// Host for dweet
const char* dweet_host = "dweet.io";
const char* dweet_name = "greenhouse_lalouani";
// Host for tcp conection
const char* tcp_host = "192.168.1.100";  // this IP is given from the WeMos
const char* host = "script.google.com";
//google apps script host
const int httpsPort = 443;
// using WiFiClientSecure class to create a TLS connection
WiFiClientSecure client;
String GAS_ID =
"AKfycbx85qGmCMTTSTJq_0DxCZ_JyYhuiXPTCUR2Oy5BdBzO5pD4GIyr";
int val; //this Val is used to compare the temperature with a reference
int ref= 30; // it is the reference which compared with the actual temp
const char* soil_status;
int digital_soil = D5;  // the soil moister sensor is connected to the pin D2
void setup() {
 Serial.begin(115200);
//connecting to the wifi and giving the ip address
delay(10); // Wait a few seconds between measurements.
Serial.println();
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");
Serial.println("the IP address is: ");
Serial.println(WiFi.localIP());
dht_sensing.begin();
 if (!bmp.begin()) {
 Serial.println("there is a wrong with pressure sensor wiring!");
  while (1) {} }
}

void loop() {
  // first we start by reading all the sensors
float humid = dht_sensing.readHumidity(); // get humidity reading
float temp = dht_sensing.readTemperature(); // get temperature reading in
Celsius
float fehr = dht_sensing.readTemperature(true); // get temperature reading
in Fahrenheit
```

```
// Check if any reads failed and exit early (to try again).
if (isnan(humid) || isnan(temp) || isnan(fehr)) {
Serial.println("Failed to read from DHT sensor!");
return; }
float soil_digital = digitalRead(digital_soil);
if (soil_digital == 1){soil_status="wet";}
else if (soil_digital == 0){soil_status="dry";}
float lighte = analogRead(A0);
float x_light = lighte * 100  /1024;
delay(500);
float pressure_Pa= bmp.readPressure();
Serial.print("Pressure ="); Serial.print(pressure_Pa);Serial.println(" Pa");
Serial.print("temperature = "); Serial.print(temp);Serial.print(" C  ");
Serial.print(fehr);Serial.println(" F");
Serial.print("humidity = ");Serial.print (humid); Serial.println(" %")
Serial.print("soil status = ");Serial.println(soil_status);
Serial.print("light amount = "); Serial.print ( x_light); Serial.println(" %");
Serial.print("seil de tempeerature = ");Serial.println(ref);
 // connecting to google drive
 Serial.print("connecting to ");
  Serial.println(host);
  if (!client.connect(host, httpsPort)) {
    Serial.println("connection failed");
    return; }
  String string_temperature =  String(temp, DEC);
  String string_humidity =  String(humid, DEC);
  String string_light =  String(x_light, DEC);
  String string_pressure =  String(pressure_Pa, DEC);
   String string_soil =  String(soil_status);
  String url = "/macros/s/" + GAS_ID + "/exec?temperature=" +
string_temperature + "&humidity=" + string_humidity + "&light=" +
string_light

+ "&pressure=" +string_pressure + "&soi_lmoisture=" + string_soil;

Serial.print("requesting URL: "); Serial.println(url);

client.print(String("GET ") + url + " HTTP/1.1\r\n" + "Host: " + host + "\r\n"
+ "Connection: close\r\n\r\n");

Serial.println("request sent");

while (client.connected()) {

String line = client.readStringUntil('\n');

if (line == "\r") { Serial.println("headers received"); break;}  }

  String line = client.readStringUntil('\n');

  if (line.startsWith("{\"state\":\"success\"")) {

  Serial.println("esp8266/Arduino CI successfull!");}

else {

  Serial.println("esp8266/Arduino CI has failed");}

  Serial.println("reply was:");

  Serial.println(line);

  Serial.println("closing connection");

 //   connection with dweet first

Serial.print("connecting to ");

Serial.println(dweet_host);

const int httpPort = 80;

WiFiClient client;

if (!client.connect(dweet_host, httpPort)) {

Serial.println("connection to dweet.io failed");

return;}

String dweet_url = "/dweet/for/";

 dweet_url += String(dweet_name);

 dweet_url += "?humidity=";  dweet_url += String(humid);

 dweet_url += "&celcuice=";  dweet_url += String(temp);
 dweet_url += "&fehrn="; dweet_url += String(fehr);
 dweet_url += "&dsoil=";  dweet_url += String(soil_status);
 dweet_url += "&pressure="; dweet_url += String(pressure_Pa);
 dweet_url += "&light=";  dweet_url += String(x_light);
Serial.print("Requesting URL for dweet: "); Serial.println(dweet_url);
client.print(String("GET ") + dweet_url + " HTTP/1.1\r\n" +  "Host: " +
dweet_host + "\r\n" +
"Connection: close\r\n\r\n");
unsigned long timout = millis();
while (client.available() == 0) {
if (millis() - timout > 2000) {
Serial.println(">> dweet Client Timeout !");
client.stop();
return; } }
while(client.available()){
String dweet_line = client.readStringUntil('\r');
Serial.print(dweet_line); }
// Close connecting with dweet.io
Serial.println();
Serial.println("closing connection with dweet.io");
client.stop();
delay(5000);
if (temp >=  ref){val= 1;}
else if (temp <  ref) {val= 0;}
// now lets make a tcp client connection
Serial.print("connecting to ");
Serial.println(tcp_host);
if (!client.connect(tcp_host, 80)) {
Serial.println("connection to wemos failed");
return; }
 // We now create a URI for the request
```

```
String tcp_url = "/temp/";
tcp_url += (val)?"1":"0";
  tcp_url += "/soil/" ;
  tcp_url += (soil_digital)?"1":"0";
Serial.print("Requesting URL: ");
Serial.println(tcp_url);
client.print(String("GET ") + tcp_url + " HTTP/1.1\r\n" + "Host: " + tcp_host
+ "\r\n" + "Connection: close\r\n\r\n");
unsigned long timeout = millis();
while (client.available() == 0) {
if (millis() - timeout > 2000) {
Serial.println(">>> Client Timeout !");
client.stop();
return; } }
while(client.available()){
String line = client.readStringUntil('\r');
Serial.print(line); }
Serial.println();
Serial.println("closing connection");
client.stop();
 delay(30000);
}
```

ANNEX.B

The Wemos's program:

```
#include <ESP8266WiFi.h>

#include <WiFiClient.h>

#include <ESP8266WebServer.h>

const char* ssid = "TP-LINK_NA";

const char* password = "YNWA0304";

WiFiServer server(80);

// Host for dweet

const char* dweet_host = "dweet.io";

const char* dweet_name = "greenhouse_lalouani_Wemos";

int ventelo1 = D6;  int ventelo2 = D5;

int wp = D2;    int water;     int val;


void setup() {

Serial.begin(9600);

delay(10);

pinMode(ventelo1, OUTPUT);

pinMode(ventelo2, OUTPUT);

digitalWrite(D6, 0);

digitalWrite(D5, 0);

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

Serial.println(ssid);

WiFi.begin(ssid, password);

while (WiFi.status() != WL_CONNECTED) {

delay(200);

Serial.print("."); }

Serial.println("");

Serial.println("WiFi connected");

Serial.println("the server's IP address is: ");

Serial.println(WiFi.localIP());

server.begin();

Serial.println("the HTTP server is started");

}


void loop() {

 //now lets do a communication between the two modules

//configure the Wemos as a server

 WiFiClient client;

 client = server.available();

if (!client) { return; }

Serial.println("new client");

while(!client.available()){

Serial.print('.');

delay(1); }
```

```
// Reading of the first line of the request

String request = client.readStringUntil('\r');

Serial.println(request);

client.flush();

if (request.indexOf("/temp/0") != -1)

{val = 0;Serial.println("no");}

else if (request.indexOf("/temp/1") != -1)

{val = 1;Serial.println("yes");}

else {

Serial.println("the request is not valid");}

if (request.indexOf("/soil/0") != -1)

{water = 0; Serial.println("no s");}

else if (request.indexOf("/soil/1") != -1)

{water = 1; Serial.println("yes s");}

else {

Serial.println("the request is not valid 2");

client.stop();

return; }

if (val == 0) {digitalWrite(D6, 0); digitalWrite(D5, 0);}

else if (val == 1) {digitalWrite(D6, 1); digitalWrite(D5, 0);}

else {

Serial.println("failur in reading .3.");

return; }

if (water == 0) {digitalWrite(wp, 0);}
```

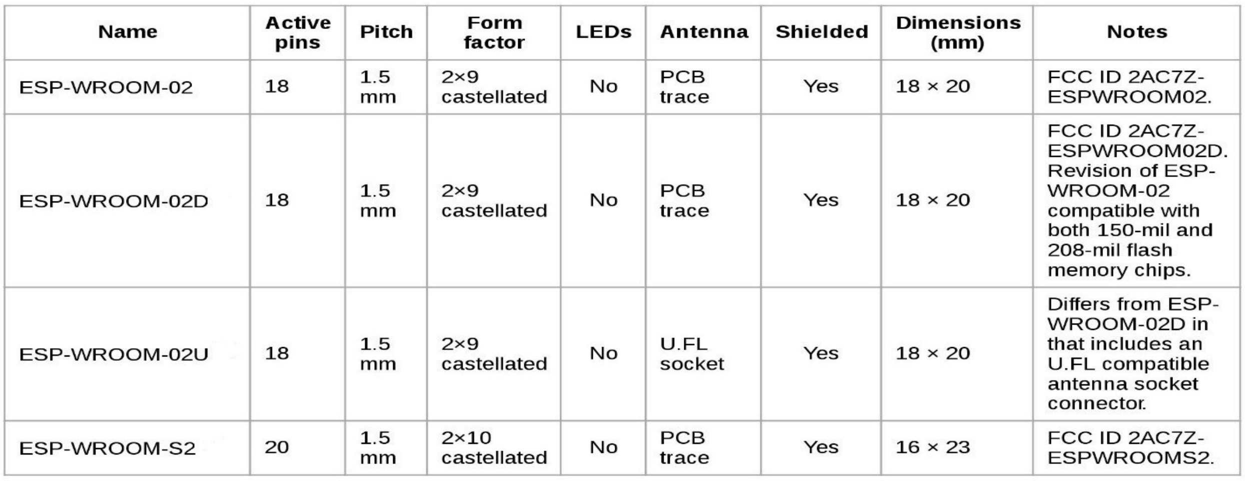
```
else if (water == 1) {digitalWrite(D6, 1);}

else { Serial.println("failur in reading .4."); return; }

client.flush();

String xx = (val)?"high":"low";

// Send the response to the client

client.print(xx);

delay(1);

Serial.println("Client disonnected");

 delay(5000);

 // connecting with the dweet.io server

Serial.print("connecting to "); Serial.println(dweet_host);

if (!client.connect(dweet_host, 80)) {

Serial.println("connection to dweet.io failed");

return; }

String dweet_url = "/dweet/for/";

 dweet_url += String(dweet_name);

 dweet_url += "?vent_status="; dweet_url += String(val);

 dweet_url += "&water_p_status="; dweet_url += String(water);

Serial.print("Requesting URL for dweet: "); Serial.println(dweet_url);

client.print(String("GET ") + dweet_url + " HTTP/1.1\r\n" + "Host: " +
dweet_host + "\r\n" +

"Connection: close\r\n\r\n");

unsigned long timout = millis();

while (client.available() == 0) {
```

```
if (millis() - timout > 2000) {

Serial.println(">> dweet Client Timeout !");

client.stop();

return; } }

while(client.available()){

String dweet_line = client.readStringUntil('\r');

Serial.print(dweet_line); }

// Close connecting with dweet.io

Serial.println(); Serial.println("closing connection with dweet.io");

client.stop();

 delay(5000);

}
```

ANNEX.C

(Table 1), **(Table 2)** and **(Table 3)** present tables of the series of ESP8266-based modules provided by “Espressif”, “AI-Thinker” and some of other ESP8266 boards. Each table have nine columns: the “Name” include the name of the ESP8266 chips. The “active pins” referred to the number of the active pins in each chip. The "Pitch" is the space between pins on the ESP8266 module, it is important to know if you are going to breadboard the device. The "Form factor" describes the module packaging, for example "2 × 9 DIL" means nine pins arranged in two rows "Dual In Line", like the pins of DIL ICs (Dual In-Line packaged Integrated Circuits). The “LEDs” means the small on-board LEDs that can be included in some of ESP-xx modules; they can be programmed for blinking to indicate the activity. The “Antenna” referred to the type of the antenna used in the module, there are many antenna options for ESP-xx boards, for example a PCB (Printed Circuit Board) trace antenna, an on-board ceramic antenna, and an external connector that allows you to attach an external Wi-Fi antenna. The “Dimensions” describes the Length and width of each chip. The “Notes” is specialized to give some important notes about each chip. Since Wi-Fi communications generates a lot of RFI (Radio Frequency Interference), governmental bodies like the FCC (Federal Communications Commission) like shielded electronics to minimize interference with other devices. Some of the ESP-xx modules come housed within a metal box with an FCC seal of approval stamped on it. First and second world markets will likely demand FCC approval and shielded Wi-Fi devices [13-14].

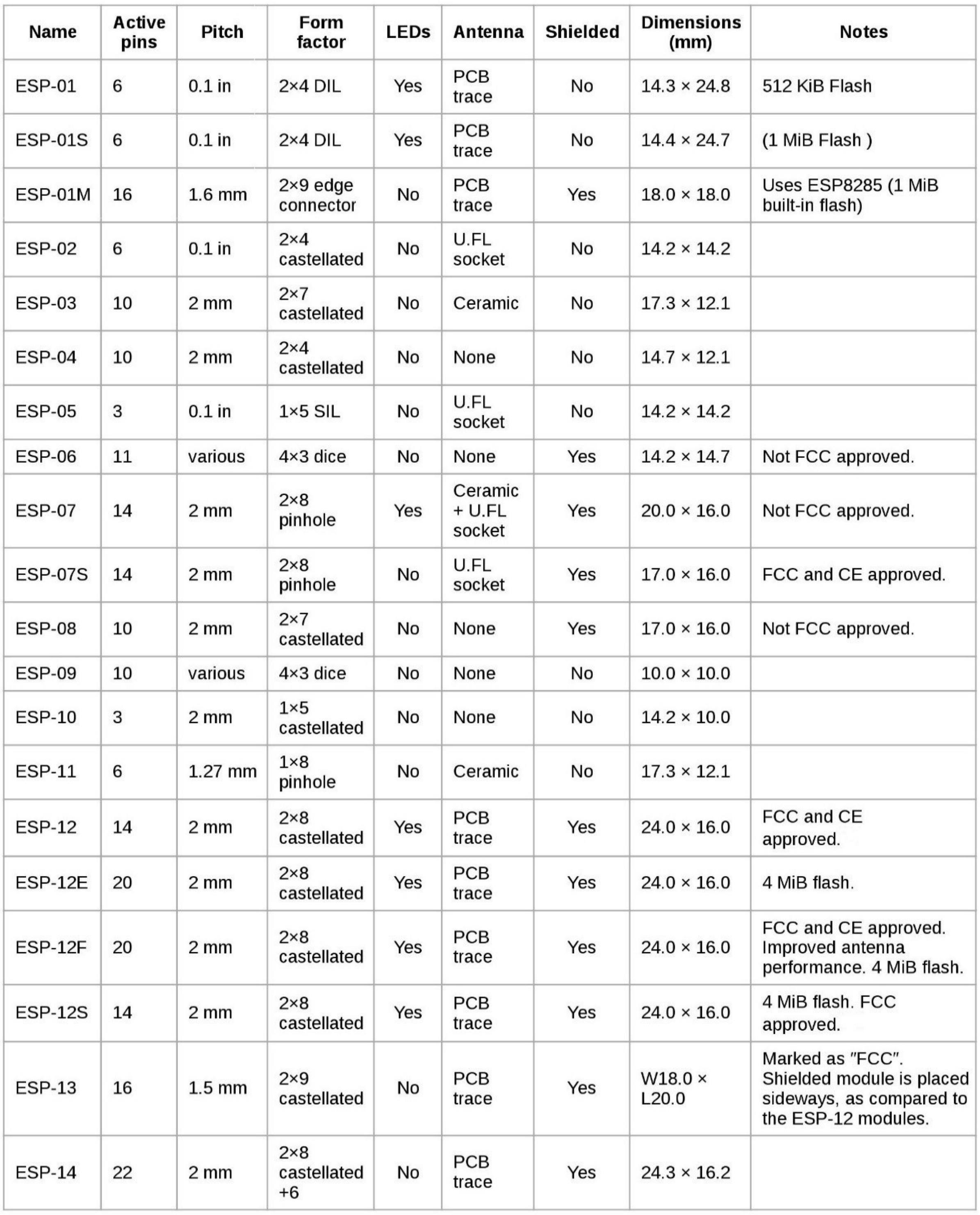
Espressif modules:



Name	Active pins	Pitch	Form factor	LEDs	Antenna	Shielded	Dimensions (mm)	Notes
ESP-WROOM-02	18	1.5 mm	2×9 castellated	No	PCB trace	Yes	18 × 20	FCC ID 2AC7Z-ESPWROOM02.
ESP-WROOM-02D	18	1.5 mm	2×9 castellated	No	PCB trace	Yes	18 × 20	FCC ID 2AC7Z-ESPWROOM02D. Revision of ESP-WROOM-02 compatible with both 150-mil and 208-mil flash memory chips.
ESP-WROOM-02U	18	1.5 mm	2×9 castellated	No	U.FL socket	Yes	18 × 20	Differs from ESP-WROOM-02D in that includes an U.FL compatible antenna socket connector.
ESP-WROOM-S2	20	1.5 mm	2×10 castellated	No	PCB trace	Yes	16 × 23	FCC ID 2AC7Z-ESPWROOMS2.

Table 1 Espressif modules.

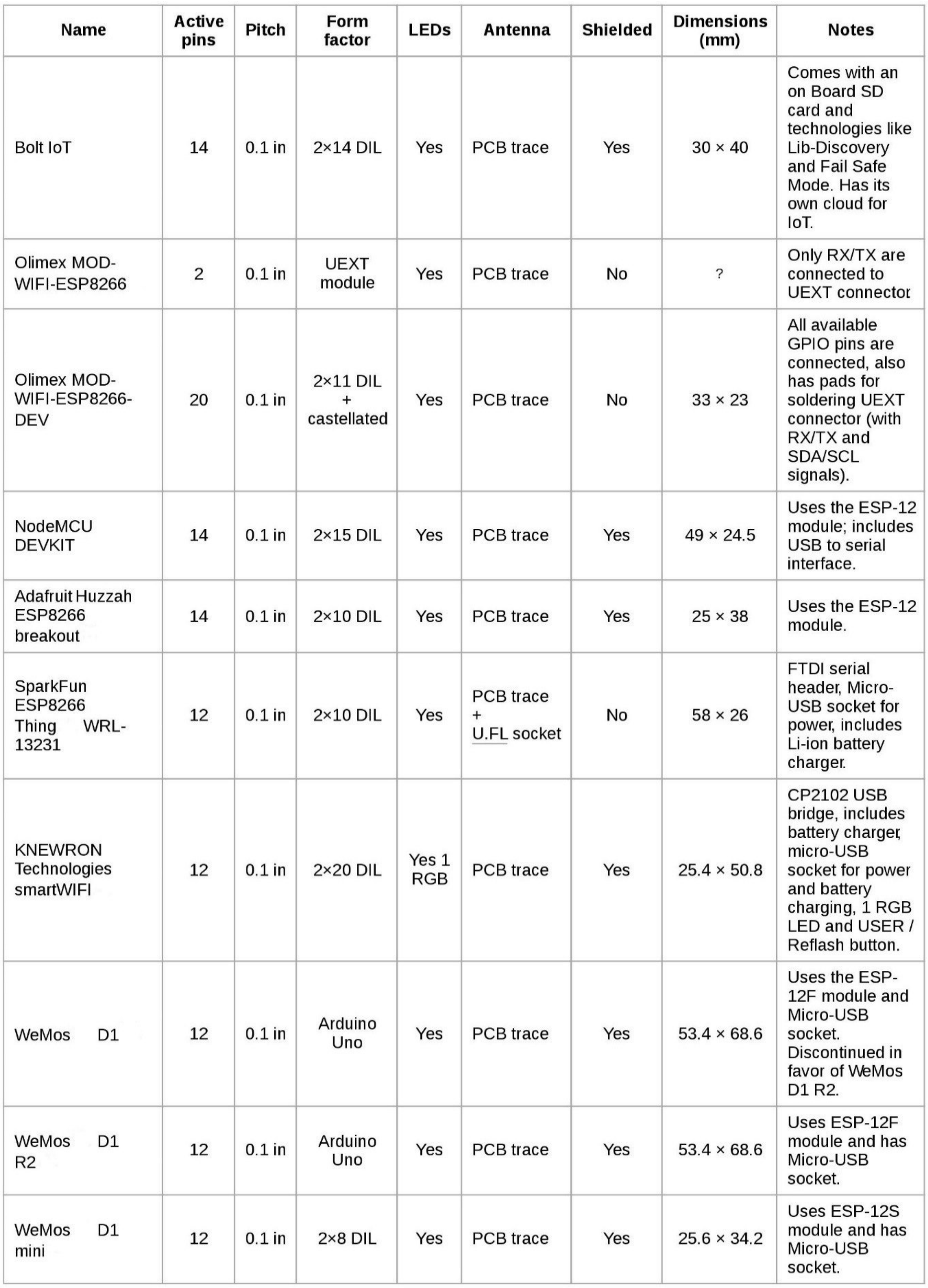
AI-Thinker modules:



Name	Active pins	Pitch	Form factor	LEDs	Antenna	Shielded	Dimensions (mm)	Notes
ESP-01	6	0.1 in	2×4 DIL	Yes	PCB trace	No	14.3 × 24.8	512 KiB Flash
ESP-01S	6	0.1 in	2×4 DIL	Yes	PCB trace	No	14.4 × 24.7	(1 MiB Flash)
ESP-01M	16	1.6 mm	2×9 edge connector	No	PCB trace	Yes	18.0 × 18.0	Uses ESP8285 (1 MiB built-in flash)
ESP-02	6	0.1 in	2×4 castellated	No	U.FL socket	No	14.2 × 14.2	
ESP-03	10	2 mm	2×7 castellated	No	Ceramic	No	17.3 × 12.1	
ESP-04	10	2 mm	2×4 castellated	No	None	No	14.7 × 12.1	
ESP-05	3	0.1 in	1×5 SIL	No	U.FL socket	No	14.2 × 14.2	
ESP-06	11	various	4×3 dice	No	None	Yes	14.2 × 14.7	Not FCC approved.
ESP-07	14	2 mm	2×8 pinhole	Yes	Ceramic + U.FL socket	Yes	20.0 × 16.0	Not FCC approved.
ESP-07S	14	2 mm	2×8 pinhole	No	U.FL socket	Yes	17.0 × 16.0	FCC and CE approved.
ESP-08	10	2 mm	2×7 castellated	No	None	Yes	17.0 × 16.0	Not FCC approved.
ESP-09	10	various	4×3 dice	No	None	No	10.0 × 10.0	
ESP-10	3	2 mm	1×5 castellated	No	None	No	14.2 × 10.0	
ESP-11	6	1.27 mm	1×8 pinhole	No	Ceramic	No	17.3 × 12.1	
ESP-12	14	2 mm	2×8 castellated	Yes	PCB trace	Yes	24.0 × 16.0	FCC and CE approved.
ESP-12E	20	2 mm	2×8 castellated	Yes	PCB trace	Yes	24.0 × 16.0	4 MiB flash.
ESP-12F	20	2 mm	2×8 castellated	Yes	PCB trace	Yes	24.0 × 16.0	FCC and CE approved. Improved antenna performance. 4 MiB flash.
ESP-12S	14	2 mm	2×8 castellated	Yes	PCB trace	Yes	24.0 × 16.0	4 MiB flash. FCC approved.
ESP-13	16	1.5 mm	2×9 castellated	No	PCB trace	Yes	W18.0 × L20.0	Marked as "FCC". Shielded module is placed sideways, as compared to the ESP-12 modules.
ESP-14	22	2 mm	2×8 castellated +6	No	PCB trace	Yes	24.3 × 16.2	

Table 2 AI-Thinker modules.

Some of other boards:



Name	Active pins	Pitch	Form factor	LEDs	Antenna	Shielded	Dimensions (mm)	Notes
Bolt IoT	14	0.1 in	2×14 DIL	Yes	PCB trace	Yes	30 × 40	Comes with an on Board SD card and technologies like Lib-Discovery and Fail Safe Mode. Has its own cloud for IoT.
Olimex MOD-WIFI-ESP8266	2	0.1 in	UEXT module	Yes	PCB trace	No	?	Only RX/TX are connected to UEXT connector
Olimex MOD-WIFI-ESP8266-DEV	20	0.1 in	2×11 DIL + castellated	Yes	PCB trace	No	33 × 23	All available GPIO pins are connected, also has pads for soldering UEXT connector (with RX/TX and SDA/SCL signals).
NodeMCU DEVKIT	14	0.1 in	2×15 DIL	Yes	PCB trace	Yes	49 × 24.5	Uses the ESP-12 module; includes USB to serial interface.
Adafruit Huzzah ESP8266 breakout	14	0.1 in	2×10 DIL	Yes	PCB trace	Yes	25 × 38	Uses the ESP-12 module.
SparkFun ESP8266 Thing WRL-13231	12	0.1 in	2×10 DIL	Yes	PCB trace + U.FL socket	No	58 × 26	FTDI serial header, Micro-USB socket for power, includes Li-ion battery charger.
KNEWRON Technologies smartWIFI	12	0.1 in	2×20 DIL	Yes 1 RGB	PCB trace	Yes	25.4 × 50.8	CP2102 USB bridge, includes battery charger, micro-USB socket for power and battery charging, 1 RGB LED and USER / Reflash button.
WeMos D1	12	0.1 in	Arduino Uno	Yes	PCB trace	Yes	53.4 × 68.6	Uses the ESP-12F module and Micro-USB socket. Discontinued in favor of WeMos D1 R2.
WeMos D1 R2	12	0.1 in	Arduino Uno	Yes	PCB trace	Yes	53.4 × 68.6	Uses ESP-12F module and has Micro-USB socket.
WeMos D1 mini	12	0.1 in	2×8 DIL	Yes	PCB trace	Yes	25.6 × 34.2	Uses ESP-12S module and has Micro-USB socket.

Table 3 Other ESP8266 boards.