**Abstract**

Food safety and hygiene is a major concern in order to prevent food wastage. The Quality of the food needs to be monitored and it must be prevented from rotting and decaying by the atmospheric factors like temperature, humidity and air quality. Therefore, it is useful to deploy quality monitoring devices at Dairy product store. These quality monitoring devices keep a watch on the environmental factor that cause or pace up decay of the food. Later, the environmental factors can be controlled like refrigeration, vacuum storage etc.

In this project, a similar food quality monitoring device will be designed that will keep watch of environmental factors like temperature, humidity, air quality and methane gas content and. The device is built on [Arduino](https://www.engineersgarage.com/articles/arduino) UNO which is a popular [prototyping board](https://www.engineersgarage.com/blogs/Top-IoT-Boards-to-Quick-Start-Prototyping). The Arduino board is interfaced with various sensors like DHT-11 to monitor temperature and humidity, MQ4 to detect Methane gas content and MQ135 to measure the Air qualit The [sensor](https://www.engineersgarage.com/articles/sensors) data is also displayed on a character [LCD](https://www.engineersgarage.com/electronic-components/16x2-lcd-module-datasheet) interfaced with the Arduino UNO. The IoT platform used for logging and monitoring of sensor data is Freeboard.io. With the power of Internet of Things, the environmental factors affecting food storage can be monitored

Many such devices can be installed at a location for better monitoring and quality control. The Arduino Sketch running over the device implements the various functionalities of the project like reading sensor data, converting them into strings, displaying them on character LCD and passing them to the IoT platform. The Sketch is written, compiled and loaded using the Arduino IDE.

**Introduction**

The food we consume can affect in any form of contamination that may occur due to storage or chemical changes within the food.

• Several viruses and bacteria causes food contamination and leads to numerous food borne diseases.

• About 351,000 people die of food poisoning globally every year.

• It is necessary to develop a system that can help people to identify the freshness of food or quality of food items

In modern times the technology is being developed to ease the day to day work. The technology is enhanced and upgraded for overall development of society in the world of globalization and urbanization. As today the health issue is one of the major reasons with effect the human life. The quality of food lies in its cleanliness and sustain for long time. The quality of the food should checked to prevent it from spoiling under different environment conditions like temperature, humidity, vegetable/fruit characteristics , which will be helpful to check quality through different techniques. Most of the health problems leading to illness or sometimes death is due to eating of unprepared or pink flesh of the animals which when rotted becomes noxious. The snsor senses the food quality through change in its color

[12]. There are various signal processing and pattern recognition techniques to detect food intake time through sensors

[13]. The rotted or not fit for usage food causes a major food related illness called as food poisoning, this is one of the diseases along with various other such diseases related to spoiled food. One of the main objectives of the food spoiler detector is that it will detect the gas released from the spoiled food and tell the user that the food is spoiled and take a look over food. The research scholars present days are now finding a new area of research which is related to recognition of food. The methods employed were very costly to install. The technique for detection of spoiled food is much easier using two approaches. There are different approaches for the detection of the various gases that are released from food. The proposed system is based on Arduino UNO which is a recognized prototyping board which is interfaced with different sensors The LCD panel will show the output from the sensor which is connected with the Arduino board.

**Existing system**

The manual method includes checking fruits and food items by human force. • This includes colour checking, tasting and smelling.

• Food checking are expensive and time consuming.

• Less efficient due to human errors and environmental effects.

**Objective**

* To make an electronic device integrated with biosensors that can detect food spoilage.
* The use sensors that can measure different parameters of food like Methane Gas, Air quality, temperature and humidity
* To display the obtained Methane Gas level and alert message is displayed on LCD display if the gas level is higher than the defined level
* To display the obtained Air quality level and alert message is displayed on LCD display if the gas level is higher than the defined level
* To display the temperature and humidity of the store is displayed on the LCD display

**Proposed system**

* The integrated electronic device with a Methane Gas sensor that can detect food freshness.
* The proposed solution senses the temperature, humidity and display it on the LCD display
* The proposed solution will detect the air quality of the food storage place and if the air is polluted the alert message will be displayed on the screen
* The result will be either the food is fresh or not

**Methodology**

The brain of this project is arduino microcontroller, and we have used the sensors like MQ4, MQ135 and DHT11 to detect the surrounding temperature, humidity, and polluted gases like Methane Gas and Air quality sensor is used to detect the quality of air, the LCD display is used to display all the parameters that is measured using the sensors and alert message is also displayed on the same LCD display.

Here in this project when the device is turn on, it will start to detect the air quality and Methane gas and temperature and humidity of the surrounding dairy store, if the Methane gas level is higher than the normal level it will considered as food spoiling indication, because when ever the food is spoiled the ripped food will emit certain gases in that methane is one of the gas, so if if the methane gas is sensed by the sensor then the food spoilage is detected and it will be displayed on the LCD display.

If the air quality of the surrounding room is polluted then the air quality sensor will detect the polluted air and compare with the given normal air quality value, if the value is higher than the obtained value then the system will send the alert message on the LCD display that the Food is spoiled.

The DHT11 sensor is used to detect the temperature and humidity of the surrounding room and it will be displayed on the LCD display.

**Applications**

* This system will be used to accurately measure the food spoilage
* It can be used in dairy product store to measure the spoilage of food
* It can be used in home to measure the spoilage of food
* It can used in hotels to measure the spoilage of food
* It can be used in storage warehouses to measure the spoilage of food

**Advantages**

* It can detect the food spoilage of dairy products
* It can detect the food spoilage in the Home
* It can detect the food spoilage in the hotels
* It can detect the food spoilage in the food store warehouse

**Disadvantage**

* The microcontroller is battery powered so continuous power supply is required to obtain accurate value

**SOFTWARE DISCRIPTION**

Arduino ide is the software used to write-compile-upload program to arduino.Its a open source software

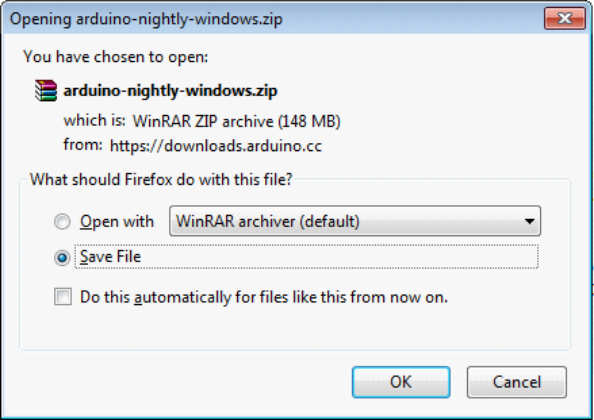
* **Procedure to Install Arduino Software (IDE)**

**Step 1** − First we must have an Arduino board and a USB cable. In case we use Arduino UNO, Arduino Duemilanove, Nano, Arduino Mega 2560, or Diecimila, we will need a standard USB cable (A plug to B plug), the kind we would connect to a USB printer as shown in Fig. 3.1.5.1 (a). In case we use Arduino Nano, we will need an A to Mini-B cable instead as shown in Fig. 3.1.5.1 (b).



**Fig 3.1.5.1 (b) A to Mini-B Cable**

**Step 2 − Download Arduino IDE Software.**

One can get different versions of Arduino IDE from the [Download page](https://www.arduino.cc/en/Main/Software) on the Arduino Official website. We must select our software, which is compatible with our operating system (Windows, IOS, or Linux). Unzip the file, after downloading it completely.

**Fig. 3.1.5.2 Downloading Arduino IDE**

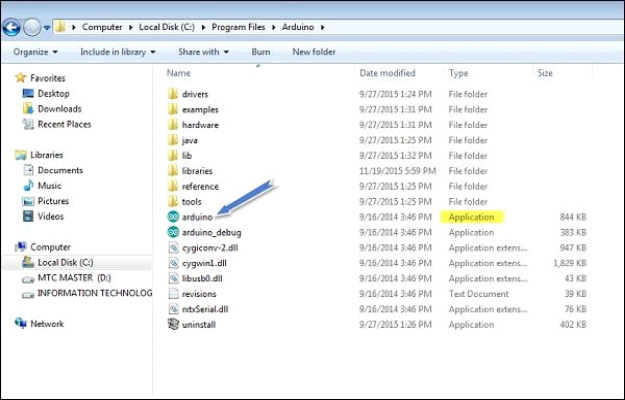
**Step 3 − Power up your board.**

The Arduino Uno, Mega, Duemilanove and Arduino Nano automatically draw power from either, the USB connection to the computer or an external power supply. If you are using an Arduino Diecimila, you have to make sure that the board is configured to draw power from the USB connection. The power source is selected with a jumper, a small piece of

plastic that fits onto two of the three pins between the USB and power jacks. Check that it is on the two pins closest to the USB port.

Connect the Arduino board to computer using the USB cable. The green power LED (labeled PWR) should glow.

**Step 4 − Launch Arduino IDE.**

After your Arduino IDE software is downloaded, you need to unzip the folder. Inside the folder, you can find the application icon with an infinity label (application.exe). Double- click the icon to start the IDE.

**Fig. 3.1.5.4 Launching Arduino IDE**

**Step 5 − Open your first project.**

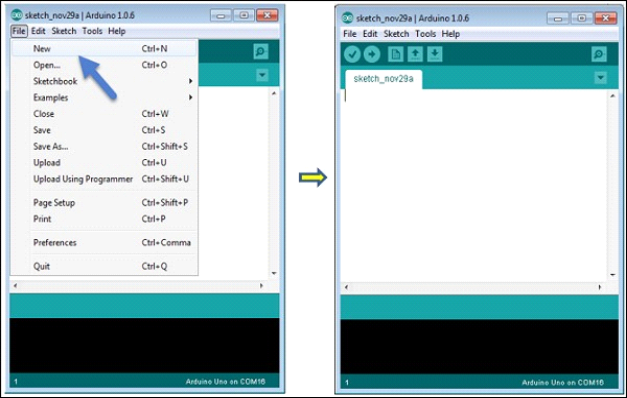
Once the software starts, we have two options:

* Create a new project.

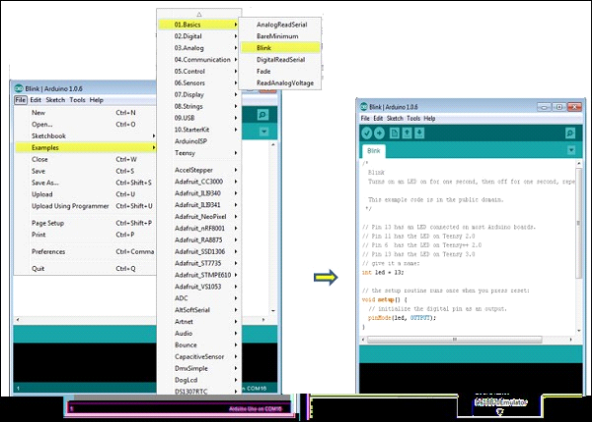
To create a new project, select File →**New,** as shown in Fig. 3.1.5.5 (a).

* Open an existing project example.

To open an existing project example, select File → Example → Basics → Blink, as shown in Fig. 3.1.5.5 (b).



**Fig. 3.1.5.5 (a) Creating a New Project**



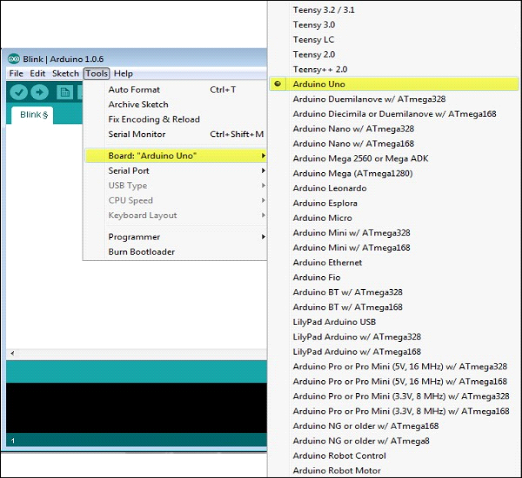
**Fig. 3.1.5.5 (b) Opening an Existing Project**

Here, we are selecting just one of the examples with the name **Blink**. It turns the LED on and off with some time delay. We can select any example from the list.

**Step 6 − Select the respective Arduino board.**

To avoid any error while uploading our program to the board, we must select the correct Arduino board name, which matches with the board connected to our computer.

Go to Tools → Board and select the board.

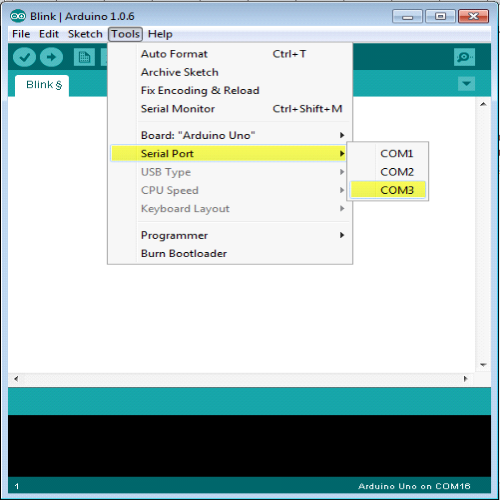


**Fig. 3.1.5.6 Selecting the Arduino Board**

**Step 7 − Select the serial port.**

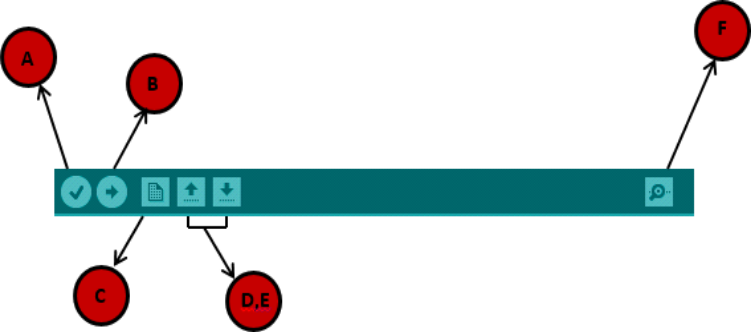
Select the serial device of the Arduino board. Go to **Tools → Serial Port** menu.

This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu, the entry that disappears should be of the Arduino board. Reconnect the board and select that serial port.



**Fig. 3.1.4.7 Selecting the Serial Port**

**Step 8 − Upload the program to the board.**

Before explaining how to upload our program to the board, we should know the function of each symbol appearing in the Arduino IDE toolbar.

**Fig. 3.1.4.8 Arduino IDE Toolbar**

**A** − Used to check if there is any compilation error. **B** − Used to upload a program to the Arduino board. **C** − Shortcut used to create a new sketch.

**D** − Used to directly open one of the example sketch.

**E** − Used to save your sketch.

**F** − Serial monitor used to send and receive the serial data from the board.

Now, simply click the "Upload" button in the environment. Wait few seconds; we will see the RX and TX LEDs on the board, flashing. If the upload is successful, the message "Done uploading" will appear in the status bar.

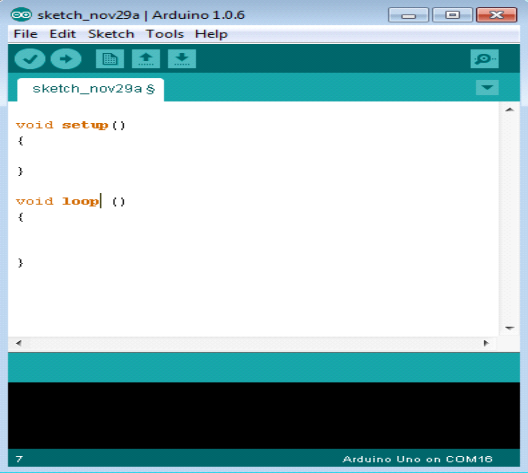
* **Program Structure**

In this section, we will study in depth, the Arduino program structure and we will learn more new terminologies used in the Arduino world. The Arduino software is open-source. The source code for the Java environment is released under the GPL and the C/C++ microcontroller libraries are under the LGPL. Arduino programs are called sketch.

Arduino programs can be divided in three main parts: **Structure, Values** (variables and constants), and **Functions**. Let us learn about the Arduino software program, step by step, and how to write the program without any syntax or compilation error.

Let us start with the **Structure**. Software structure consist of two main functions:

* Setup( ) function
* Loop( ) function



**Fig. 3.1.6 Structure of a Sketch**

The **setup ()** function is called when a sketch starts. It is used to initialize the variables, pin modes, start using libraries, etc. The setup function will only run once, after each power up or reset of the Arduino board.

After creating a **setup ()** function, which initializes and sets the initial values, the **loop ()** function does precisely what its name suggests, and loops consecutively, allowing our program to change and respond. It actively controls the Arduino board.

* **Programming using Embedded C**

C is a high-level programming language intended for system programming. Embedded C is an extension that provides support for developing efficient programs for embedded devices. Yet, it is not a part of the C language. In our Internship program, we employed Embedded C programs to write sketches to be dumped on Arduino Uno.

* **Introduction to Embedded C**

Embedded C programming language is an extension to the traditional C programming language that is used in embedded systems. The embedded C programming language uses the same syntax and semantics as the C programming language.

The only extension in the Embedded C language from normal C Programming Language is the I/O Hardware Addressing, fixed-point arithmetic operations, accessing address spaces, etc.

* **Basic Structure of Embedded C Program:**

The embedded C program has a [structure similar to C programming.](https://www.edureka.co/blog/basic-structure-of-a-c-program/) The five layers of Embedded C programming structure are:

* Comments
* Pre-processor directives
* Global declaration
* Local declaration
* Main function()

The whole code follows the below outline. This is the basic structure of the embedded c program. Each code has a similar outline. Now let us learn about each of this layer in detail.

Outline of an Embedded C code is as shown below:

* Multiline Comments Denoted using /\*……\*/
* Single Line Comments Denoted using //
* Pre-processor Directives #include<…> or #define
* Global Variables Accessible anywhere in the program
* Function Declarations Declaring Function
* Main Function Main Function, execution begins here

7. {

* Local Variables Variables confined to main function
* Function Calls Calling other Functions
* Infinite Loop Like while (1) or for (;;)
* Statements . . . . .

12. ….

13. ….

14. }

15. Function Definitions Defining the Functions

16. {

* Local Variables Local Variables confined to this Function
* Statements . . . . .

19. ….

20. ….

21. }

* **Comment Section:** Comments are simple readable text, written in code to make it more understandable to the reader. Usually comments are written in // or /\* \*/.

**Example:** //Test program

* **Pre-processor Directives Section:** The Pre-Processor directives tell the compiler which files to look in to find the symbols that are not present in the program.

**For Example,** in 8051 Keil compiler we use,

* #include<reg51.h>
* **Global Declaration Section:** The global variables and the user-defined functions are declared in this part of the code. They can be accessed from anywhere.

1. void delay (int);

* **Local Declaration Section:** These variables are declared in the respective functions and cannot be used outside the main function.
* **Main Function Section:** Every C programs need to have the main function. So does an embedded C program. Each main function contains 2 parts. A declaration part and an Execution part. The declaration part is the part where all the variables are declared. The execution part begins with the curly brackets and ends with the curly close bracket. Both the declaration and execution part are inside the curly braces.

**Example:**

1. void main(void) // Main Function 2. {

3. P1 = 0x00;

4. while(1)

5. {

6. P1 = 0xFF;

7. delay(1000);

8. P1 = 0x00;

9. delay(1000);

10. }

11. }

* **Function Definition Section:** The function is defined in this section.
* **Arduino Code libraries**
* **Library Structure**
* A library is a folder comprised with C++ (.cpp) code files and C++ (.h) header files.
* The .h file describes the structure of the library and declares all its variables and functions.
* The .cpp file holds the function implementation.
* **Importing Libraries**

The first thing to do is to find the library we want to use out of the many libraries available online. After downloading it to our computer, we just need to open Arduino IDE and click on Sketch > Include Library > Manage Libraries. We can then select the library we want to import. Once the process is complete the library will be available in the sketch menu.

* **Arduino Code Explanation**

Arduino code is written in C++ with an addition of special methods and functions. C++ is a human-readable programming language. When we create a sketch (the name given to Arduino code files), it is processed and compiled to machine language.

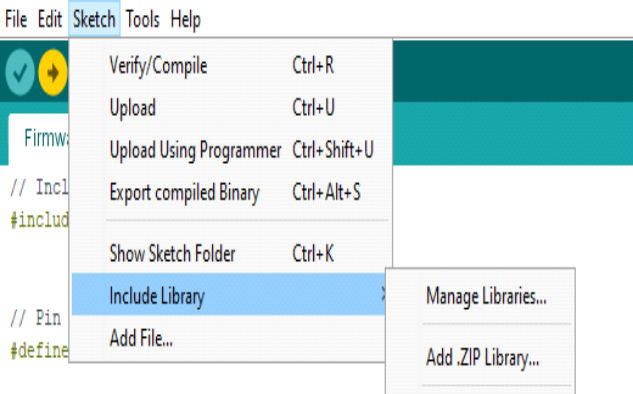
* **Code Structure**

The basic concepts which one should know to write a program on Arduino IDE are discussed below:

* **Libraries**

In Arduino, much like other leading programming platforms, there are built-in libraries that provide basic functionality. In addition, its possible to import other libraries and expand the Arduino board capabilities and features. These libraries are roughly divided into libraries that interact with a specific component or those that implement new functions.

To import a new library, we need to go to Sketch > Import Library



**Fig. 3.2.2.1 Including Libraries on IDE**

In addition, at the top of our file, we have to use #include to include external libraries. We can also create custom libraries to use in isolated sketches.

* **Pin Definitions**

To use the Arduino pins, we need to define which pin is being used and its functionality. A convenient way to define the used pins is by using:

**#define pinName pinNumber.**

The functionality is either input or output and is defined using the pinMode () method in the setup section.

* **Declarations**
* **Variables**

Whenever were using Arduino, we need to declare global variables and instances to be used later on. In a nutshell, a variable allows us to name and store a value to be used in the future. For example, we would store data acquired from a sensor in order to use it later. To declare a variable we simply define its type, name and initial value. Its worth mentioning that declaring global variables isnt an absolute necessity. However, its advisable to declare variables to make it easy to utilize our values further down the line.

* **Instances**

In software programming, a class is a collection of functions and variables that are kept together in one place. Each class has a special function known as a constructor, which is used to create an instance of the class. In order to use the functions of the class, we need to declare an instance for it.

* **Setup()**

Every Arduino sketch must have a setup function. This function defines the initial state of the Arduino upon boot and runs only once.

Here well define the following:

* Pin functionality using the pinMode function
* Initial state of pins
* Initialize classes
* Initialize variables
* Code logic
* **Loop()**

The loop function is also a must for every Arduino sketch and executes once setup () is complete. It is the main function and as its name hints, it runs in a loop over and over again. The loop describes the main logic of our circuit.

* **Code Logic**

The basic Arduino code logic is an if-then structure and can be divided into 4 blocks:

* **Setup** - will usually be written in the setup section of the Arduino code, and performs things that need to be done only once, such as sensor calibration.
* **Input** - at the beginning of the loop, read the inputs. These values will be used as conditions (if) such as the ambient light reading from an LDR using analogRead ().
* **Manipulate Data** - this section is used to transform the data into a more convenient form or perform calculations. For instance, the AnalogRead () gives a reading of 0-1023 which can be mapped to a range of 0-255 to be used for PWM. (See analogWrite ())
* **Output** - this section defines the final outcome of the logic (then) according to the data calculated in the previous step. Looking at an example of the LDR and PWM, turns on an LED only when the ambient light level goes below a certain threshold.
* **From Software to Hardware**

There is a lot to be said of Arduinos software capabilities, but its important to remember that the platform is comprised of both software and hardware. The two work in tandem to run a complex operating system.

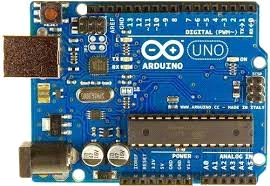
Code → Compile → Upload → Run

At the core of Arduino, is the ability to compile and run the code.

After writing the code in the IDE we need to upload it to the Arduino. Clicking the Upload button (the right-facing arrow icon), will compile the code and upload it if it passed compilation. Once the upload is complete, the program will start running automatically.

**Hardware requirements**

**ARDUINO UNO**

****

The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc.The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits.The board has 14 digital I/O pins (six capable of PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable. It can be powered by the USB cable or by an external 9-volt battery, though it accepts voltages between 7 and 20 volts. It is similar to the Arduino Nano and Leonardo.

While the Uno communicates using the original STK500 protocol, it differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it uses the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

battery



This LG INR18650 M26 2600mAh Lithium-Ion Battery gives value for your money. It comes with a rated voltage of 3.7 volts and a capacity of 2600mAh. It is a single cell, compact, and powerful battery cell with 2600 mAh capacity. It is very convenient to install in your project to fulfill the 3.7 Volt requirement with high capacity.

The battery terminals can use in any compatible battery adapter/holder or it can be permanently soldered to your applications power source wires.

Features :

High energy density

High working voltage for single battery cells.

Pollution-free

Long cycle life

No memory effect

Capacity, resistance, Voltage, platform time consistency is good.

Good consistency and low self-discharge.

Lightweight, small size

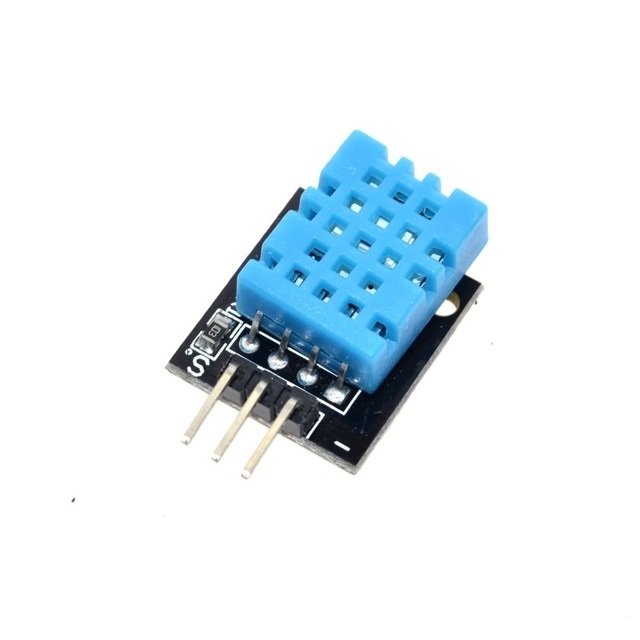
Shape: Cylindrical Battery

Battery Type: Lithium-Ion Battery

High performance and capacity

Flat top to suit many devices fitting.

# **DHT11 Digital Relative Humidity and Temperature Sensor Module**



This DHT11 Digital Relative Humidity & Temperature Sensor Module is pre-calibrated with resistive sense technology coupled with NTC thermistor, for the precise reading of the relative Humidity and surrounding temperature [DHT 11](https://robu.in/product-tag/dht11/) break-out board is a very popular, low-cost sensor from Aosong, the breakout provides easy installation of the [DHT11](https://robu.in/product-tag/dht11/) sensor module.

***The board is also equipped with high- performance 8-Bit***[***microcontroller***](https://robu.in/product-category/development-board/)***which is connected to the DHT11 sensor module. The output of the DHT11 is in the form of a digital signal on a single data pin. The sensing update frequency is to be measured at every 2sec (0.5Hz).***

The complete arrangement makes the device an ideal sensing setup to be hooked up directly to any kind of microcontroller boards like [*Arduino*](https://robu.in/product-category/arduino-2/arduino/)’s. The board is extra featured with onboard LED, a bypass capacitor between Vcc and Gnd and a pull-up resistor across the data line and Vcc.

The DHT-11 Digital Temperature And Humidity Sensor is a basic, ultra low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air and spits out a digital signal on the data pin (no analog input pins needed).

***Its fairly simple to use, but requires careful timing to grab data. The only real downside of this sensor is you can only get new data from it once every 2 seconds, so in your code please use sensor reading interval at 2 seconds or more. Compared to the***[***DHT22***](https://robu.in/product/dht22-digital-temperature-humidity-sensor-temperature-humidity-module-am2302/)***, this sensor is less precise, less accurate and works in a smaller range of temperature/humidity.***

But despite its disadvantages over [DHT22](https://robu.in/product/dht22-digital-temperature-humidity-sensor-temperature-humidity-module-am2302/), it is a smaller and less expensive sensor for temperature and humidity measurement.

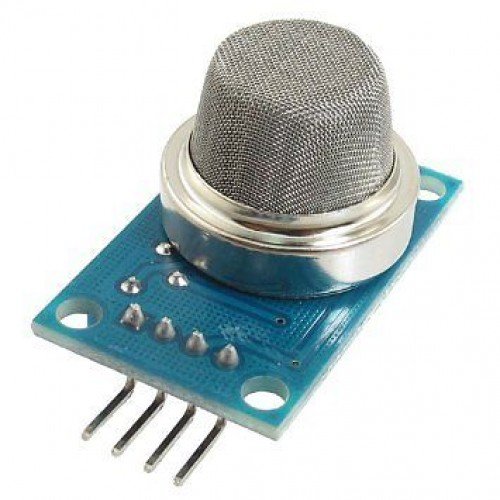
##### **Application :**

HVAC, Testing and inspection equipment, Automotive, Datalogger, Consumer goods, Automatic control, Weather station, Home appliances, Humidity regulator, Medical, Dehumidifier, etc.

#### **Features :**

1. Low power consumption.
2. DHT11 sensor adopts
3. The module can detect the surrounding environment of the humidity and temperature
4. High reliability and excellent long-term stability
5. The output from the digital output
6. Has fixed bolt hole and easy installation

# **MQ-4 High Sensitivity Gas Methane (CNG) Detector Sensor Module**



If you want to detect methane gases with your projects, you will need sensors. If you need any data on the sensitive material then, the MQ-4 High Sensitivity Gas Methane (CNG) Detector [**Gas Sensor**](https://robu.in/product-category/sensor/gas-sensor/)  Module is a quite good solution.

This is a simple to use MQ-4 High Sensitivity Gas Methane (CNG) Detector Sensor Module which can sense the presence of Compressed Natural Gas(CNG), methane(CH4), etc in the air. The module uses our MQ-4 sensor. It simplifies the interface to the odd pin spacing of the sensor and provides the interface through standard 4 x 0.1″ header pins.

It provides an analog output corresponding to the concentration of the gases in the air and an easy-to-use digital output. The onboard potentiometer can be used to set the maximum gas concentration beyond which the digital output gets triggered. The MQ-4 can detect natural gas concentrations anywhere from 200 to 10000ppm.

Just power the module with 5V set the threshold and you can start getting the gas concentration of the air around the sensor! An onboard LED signals the presence of any gas.

**The digital output can be easily interfaced with microcontrollers and other circuits. The analog output can be hooked up to an ADC of a microcontroller to get a wide range of sensor readings. This is a simple-to-use compressed natural gas (CNG) sensor, suitable for sensing natural gas (composed of mostly Methane [CH4]) concentrations in the air.**

**If you want to explore the more variety of gas sensors, then click on the link** [**Gas Sensor**](https://robu.in/product-category/sensor/gas-sensor/)

#### **Features :**

1. Signal output indicator
2. Dual Signals output (analog output and TTL output)
3. The TTL output signal is low level, allows to connect with SCM directly
4. Analog output 0~5V voltage
5. High sensitivity
6. Long working life and stable performance
7. Quick response and recovery.

# **MQ-135 Gas Sensor**



If you want to detect different types of gases, you will need sensors. If you need any data on the sensitive material then, the Waveshare MQ-135 [**Gas Sensor**](https://robu.in/product-category/sensor/gas-sensor/) is a quite good solution.

The MQ-135 sensor is a  gas sensor module that incorporates MQ-135, designed for air quality monitoring. It can be used to measure benzene, alcohol, smoke, then generate digital signals for triggering air treatment devices. The MQ-135 is used in Air quality Monitor in consumer and industry applications, this sensor is suitable for detecting benzene, alcohol, smoke. Avoid the noise of LPG, natural gas, coal gas. The sensitivity can be adjusted by the potentiometer.MQ-135 **gas sensor** applies SnO2 which has lower conductivity in the clear air as a gas-sensing material. In an atmosphere where there may be polluting gas, the conductivity of the gas sensor raises along with the concentration of the polluting gas increases

The sensitive materials of the MQ-135 gas sensor are benzene, alcohol, and smoke. When the target combustible gas exists, The sensor’s conductivity is higher along with the gas concentration rising. Please use a simple electro circuit, Convert the change of conductivity to the corresponding output signal of gas concentration.

**Connections**

**VCC** ↔ 2.5V ~ 5.0V

**GND** ↔ power supply ground

**AOUT** ↔ MCU.IO (analog output)

**DOUT** ↔ MCU.IO (digital output)

##### Application :

Air quality monitor

**For more Gas Sensors Click below**

[GAS sensor](https://robu.in/product-category/sensor/gas-sensor/)

Features:

1. Sensitive for benzene, alcohol, smoke
2. Output voltage boosts along with the concentration of the measured gases increases
3. Fast response and recovery
4. Adjustable sensitivity
5. Signal output indicator

# **LCD Display**



If you want to add some visual output to your Arduino projects, you’ll need a display. If you need only a little to display, the LCD1602 Parallel LCD Display is a quite good solution.

This is LCD1602 Parallel LCD Display that provides a simple and cost-effective solution for adding a 16×2 Black on RGB Liquid Crystal Display into your project. **The display is 16 character by 2 line display that has a very clear and high contrast black text upon a yellow background/backlight.**

This is the great yellow backlight [LCD display](https://robu.in/product-category/led-lcd-and-display-boards/lcd-display/). It is fantastic for Arduino-based projects. This LCD1602 Parallel LCD Display with Yellow Backlight is very easy to interface with [Arduino](https://robu.in/product-category/arduino-2/arduino/) or [Other Microcontrollers.](https://robu.in/product-category/development-board/)

The values shown on the display can be either simple text or numerical values read by the [sensors](https://robu.in/product-category/sensors/), such as temperature or pressure, or even the number of cycles that the Arduino is performing.

One thing to consider is you’ll waste about 8 Pins on your Arduino for the display to get working. Luckily there exists an I2C adapter that you can solder right onto the pins of the [*display*](https://robu.in/product-category/led-lcd-and-display-boards/). So all you need to connect are the I2C pins, which shows a good library and little of coding. ***We have the***

#### **Features :**

1. 16 characters wide, 2 rows
2. Black text on the yellow background
3. Single LED backlight included can be dimmed easily with a resistor or PWM.
4. Can be fully controlled with only 6 digital lines! (Any analog/digital pins can be used)

**Conclusion**

The main aim of this project is to build the device which is detect the food spoilage and alert the user by displaying on the LCD display, and we have achieved that by using MQ3, MQ135 and DHT11 sensors which provides us the Methane Gas, Air quality amd temperature and humidity of the surrounding storage room, if the methane gas of air quality is higher than the usual then the alert will be displayed on the LCD display, and constant measuring of temperature and humidity will be displayed on the LCD display so the user can monitor the temperature ad humidity of surrounding room.

**References**

1.Sumathi MS, Thejaswini S, Pranav Kashyap, ShahinaAnjum, Shashi Shanker, Shreya GK "IoT based project for food quality and monitoring" International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169 Volume: 3 Issue: 5 3172 – 3174

2. Fatima Mustafa and Silvana Andreescu \*Department of Chemistry and Biomolecular Science, Clarkson University, Potsdam, NY 13699 USA Chemical and Biological Sensors for Food-Quality Monitoring and Smart Packaging 3.https://create.arduino.cc/projecthub/taifur/drs-enabledsmart pantry8d2f8b?ref=tag&ref\_id=iot&offset=42"Smart storage of food based on IoT"

4. Mr.A.Venkatesh,T.Saravanakumar, S.Vairamsrinivasan, A.Vigneshwar, M.Santhosh Kumar."A Food Monitoring System Based on Bluetooth Low Energy and Internet of Things". Mr.A.Venkatesh et al. Int. Journal of Engineering Research and Application www.ijera.com ISSN: 2248- 9622, Vol. 7, Issue 3, (Part -6) March 2017, pp.30-34.

5.Samaneh Matindoust, Majidbaghei-Nejad, Mohammed Hadishahrokh Abadi, ZhouZou, Li-Rongzheng. "Food quality and safety monitoring using gas sensor array in intelligent packaging".

6.Ashish Kumar Singh and Neelam Verma, ”Quartz Crystal Microbalance Based Approach for Food Quality” Current Biotechnology, 2014, 3, 000-000

7.Chanthini.B, Manivannan.D, Umamakeswari.A “Perishable Food Quality Monitoring – An Internet of Things (IoT) Approach” ISSN: 1311-8080 (printed version); ISSN: 1314-3395 (on-line version) url: <http://www.ijpam.eu>

8.Kong Xiansheng and Sun Jing “Design and implementation of food intelligent monitoring system based on pH sensor”, Department of Computer & Information, Xinxiang University, China Xinxiang Radio and Television University, Xinxiang, China Journal of Chemical and Pharmaceutical Research, 2014, 6(7):1662-16666.

9. Syeda Erfana Zohora, A. M. Khan, A .K. Srivastava, Nisar Hundewale International, “Electronic Noses Application to Food Analysis Using Metal Oxide Sensors: A Review” Journal of Soft Computing and Engineering(IJSCE) ISSN: 2231-2307, Volume-3, Issue-5, November 2013.

10.Radislav A. Potyrailo, Nandini Nagraj, Zhexiong Tang, Frank J. Mondello, Cheryl Surman, and William Morris, J Agric Food Chem.,“Battery-free radio frequency identification (RFID) sensors for food quality and safety” Sep 5, 2012; 60(35): 8535– 8543.

11.Wen-Ding Huang, Sanchali Deb, Young-SikSeo, Smitha Rao, Mu Chiao, and J.C. Chiao, “A Passive RadioFrequency pHSensing Tag for Wireless Food-Quality Monitoring” 2010 IEEE.

12.Hayes, J. ; Dublin City Univ., Dublin ; Pacquit, A. ; Crowley, K.; Kim Lau “Web-based colorimetric sensing for food quality monitoring” Sensors, 2006.

13.E.sazonov, J.M Fontana, A sensor system for Automatic detection of food intake through non-invasive monitoring of chewing, IEEE Sens.J 1340-1348.