**International Journal of Engineering and Applied Physics (IJEAP)**
Vol. 3, No. 3, September 2023, pp. 879~890
ISSN: 2737-8071



# Empowering Flexible Learning: The Role of Embedded Systems in Remote Laboratories

**Abdessamad MALAOUI**
Research Laboratory in Physicals and Sciences for Engineers (LRPSI), Sultan Moulay Slimane University, Béni Mellal, Morocco

**Article Info**

***Article history:***

Received Jun 5, 2023
Revised Jun 19, 2023
Accepted July 30, 2023

***Keywords:***

Remote experience
Embedded systems
Low-cost device
Arduino microcontroller
Remote Practical Works

**ABSTRACT**

This paper explores developing and implementing a practical work remote control operational architecture using embedded systems. This architecture is particularly relevant to electronics university courses, offering innovative approaches to low-cost remote laboratories. Students through the university's local network at their convenience, providing a more flexible learning approach compared to traditional fixed schedules, can access these virtual labs. This strategy addresses the challenges posed by the high number of enrollees in Moroccan Faculties, especially in open-access institutions.

The study presents a detailed examination of this prototype's hardware and software components, with the potential for application across various disciplines. The hardware element comprises an Arduino microcontroller, responsible for managing electronic experiments involving a single-phase rectifier, chosen as an illustrative example of remote-controlled manipulation.

To evaluate the efficacy of this educational system, it underwent testing by students in the field of electronics and industrial systems. The outcomes of this evaluation highlight the numerous advantages of this innovation.



***Corresponding Author:***

Abdessamad MALAOUI,
Research Laboratory in Physicals and Sciences for Engineers (LRPSI),
Sultan Moulay Slimane University,
Béni Mellal, Morocco.
Email: a.malaoui@usms.ma

## 1. INTRODUCTION

In recent years, the enrollment of students in open-access faculties has witnessed a remarkable surge. Technical training within these faculties has faced the challenge of accommodating Practical Works (PW) in a context where physical interaction and space utilization have been constrained (especially during the COVID-19 pandemic). The logistical predicament of these faculties has been exacerbated by the surge in student numbers, amplified by the shortage of available teachers and adequate spaces for conducting PW. In parallel, the rapid and expansive evolution of communication and instrumentation technology has become particularly pronounced, rendering it accessible across a wide array of institutions and, to some extent, to individual students. Leveraging this technology for scientific research is imperative in addressing the predicaments outlined [1].

The Polydisciplinary Faculty of Béni Mellal has collaborated as a partner in an international Bachelor's program, which emphasizes practical remote training. This initiative takes the form of the Tempus-EOLES Project, uniting more than 15 universities from Europe (France, Portugal, Romania, Belgium) and North Africa (Morocco, Algeria, and Tunisia) [2]. The outcomes of this endeavor underscore the feasibility of establishing remote labs [3, 4], thus inspiring us to localize and adapt this experience within the framework of Beni Mellal's establishment, to offer innovative solutions to the challenges previously highlighted.

The success of this approach pivots on the automation of instrumentation, facilitated by embedded systems and programmable electronics, thereby minimizing the need for direct human intervention. The rapid advancements in information technology have further enriched the potential of this approach. Remote acquisition of measurement instruments, even through wireless connections, has become a viable reality, reshaping traditional boundaries [5]. These capabilities not only yield cost efficiencies but also enhance the precision of technical manipulations [6].

This study aims to investigate an educational solution tailored for teaching third-year practical works in the academic cycle, a mission that holds even more relevance in the context of the COVID-19 pandemic. Our proposal involves designing and implementing an electronic device rooted in embedded systems, enabling remote manipulation of electronics-based practical works. Our focus lies in remote measurements and control of electronic circuits, executed through an Arduino board interfacing with a computer. This data will be processed by the computer before being relayed to the Arduino, with the local network bridging the connection between these components. The integration of such a system would empower students within these institutions to access and conduct PW through Wifi connections from any corner of the faculty, thereby circumventing the constraints imposed by time and space limitations.

The phased implementation of this system encompasses three core stages: the realization of an electronic system designed to automate the acquisition of electronic signals, the development of a computer-based platform for remote hardware management, and the validation and preliminary assessment of this prototype facility. Rigorous testing was conducted by student groups from the third year of electronic training, aimed at evaluating the system's potential in terms of educational effectiveness, time efficiency, cost optimization, spatial utilization, effort minimization, and other pertinent criteria.

## 2. IMPEMENTED ARCHITECTURE

Remote Practical Works (PW) are situated within the realm of remote laboratories [7, 8, 9]. This framework is structured around a communication system overseen by a controller (computer or embedded system), which manages communication with an array of distant instruments. The process of remote control can be executed within a local network, leveraging various communication interfaces with measurement devices. This control can be realized through multiple platforms, including the following structures.

Within this architecture, numerous students establish connections via a LAN to a client-server under the supervision of an administrator. The Arduino assumes the role of the server, facilitated by an ENC28J60 Ethernet module. Additionally, the embedded system efficiently oversees the manipulation of various experiments by employing a multiplexing mechanism comprising relays and decoders. To distinguish it from other embedded systems, the Arduino is programmed with a distinct IP address, ensuring uniqueness.

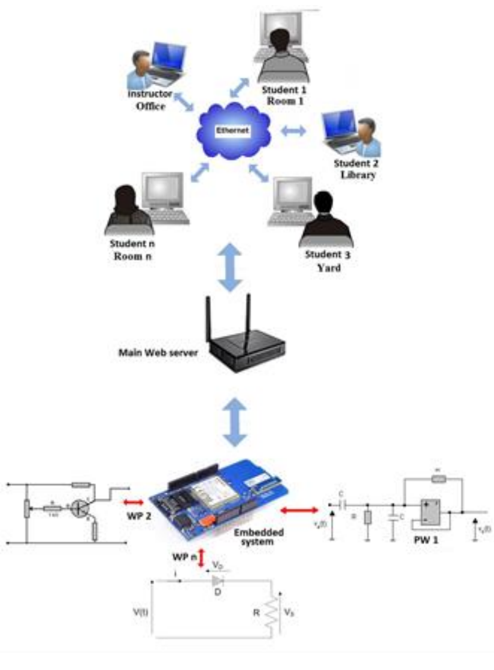


Figure 1. Used Remote Lab Architecture

## 3. HARDWARE DESCRIPTION

### 3.1. Used embedded system

This involves an Arduino Mega 2560 built upon the ATmega2560 microcontroller framework. It boasts 54 digital input/output pins (with 15 potentially serving as PWM outputs), along with 16 analog inputs, 4 UART serial ports, a 16 MHz crystal oscillator, USB connectivity, a power jack, and an ICSP header.

The ATmega2560 is equipped with 256 KB of flash memory designated for code storage (of which 8 KB is allocated for the bootloader), along with 8 KB of SRAM and 4 KB of EEPROM. The recommended power supply range for the Arduino Mega spans from 7 to 12 volts, and it can operate using an external supply within the range of 6 to 20 volts. While the 5V pin might deliver slightly less than five volts, it's vital to note that utilizing a supply exceeding 12V could lead to overheating of the voltage regulator, potentially causing damage to the board.

The ATmega2560 furnishes an array of I/O pins, offering versatility. Each of the 54 digital pins is adaptable for input or output functions, attainable through the pinMode(), digitalWrite(), and digitalRead() functions. These pins can function at 5 volts and support a maximum current of 40 mA, along with an inherent pull-up resistor (disengaged by default) ranging from 20 to 50 kOhms. Furthermore, specialized functions are attributed to specific pins as follows:

• Serial Communication: Pins 0 (RX) and 1 (TX); Serial 1: Pins 19 (RX) and 18 (TX); Serial 2: Pins 17 (RX) and 16 (TX); Serial 3: Pins 15 (RX) and 14 (TX). These pins are employed for receiving (RX) and transmitting (TX) TTL serial data. Notably, Pins 0 and 1 are interconnected with the corresponding pins of the ATmega16U2 USB-to-TTL Serial chip.

• External Interrupts: Pins 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2). Configurable to trigger an interrupt on a low value, a rising or falling edge, or a change in value. For detailed information, consult the attachInterrupt() function.

• Pulse Width Modulation (PWM): Pins 2 through 13 and 44 through 46. Facilitate 8-bit PWM output using the analogWrite() function.

• Serial Peripheral Interface (SPI): Pins 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS). These pins support SPI communication through the SPI library. Notably, the SPI pins are also accessible through the ICSP header, which is physically compatible with Uno, Duemilanove, and Diecimila.

• Two-Wire Interface (TWI): Pins 20 (SDA) and 21 (SCL). Enable TWI communication through the Wire library. It's important to note that these pins are not situated in the same location as the TWI pins on the Duemilanove or Diecimila.

Additionally, the Mega2560 features 16 analog inputs, each offering 10 bits of resolution spanning from ground to 5 volts. A couple of other pins are present on the board:

• AREF: Serves as the reference voltage for the analog inputs. Utilized in conjunction with analogReference().

• Reset: Reducing this line to a LOW state initiates a microcontroller reset. This functionality is typically employed to incorporate a reset button on shields that may obstruct the onboard reset button.

Similarly, the Arduino Mega2560 offers an array of functionalities aimed at facilitating seamless communication with a computer. It is equipped with four hardware UARTs designed to support TTL (5V) serial communication. The Arduino software encompasses a serial monitor, a tool that enables a straightforward exchange of textual data between the board and the computer. Notably, when data is transmitted via the ATmega8U2/ATmega16U2 chip and the USB connection to the computer, the RX and TX LEDs on the board will illuminate. It's worth mentioning that this illumination doesn't apply to serial communication occurring through pins 0 and 1.

### 3.2. Internet connection module

Present here is the ENC28J60 Ethernet module, serving the purpose of establishing connections either within a Local Area Network (LAN) or via an RJ45 cable, connecting seamlessly with the Arduino. Notably, the Arduino becomes accessible from any location through a network-connected computer. This module encompasses an open-source TCP/IP protocol stack accessible as an Arduino library. The module's pins are delineated as follows:

- CS: Designated as the selection pin, particularly applicable if the master circuit is interfaced with multiple slave circuits.
- IF: Designated as the slave input, this pin is reserved for the reception of data transmitted from the master circuit.
- SO: Acting as the slave output, this pin is allocated for transmitting data to the master circuit.

- SCK: This pin takes on the role of synchronizing the clock signals between the dual master and slave circuits.
- INT: A pin designed to trigger an interrupt-driven process.
- RESET: Pin allocated for initiating a reset operation.
- VCC: Serves as the supply source for the circuit.
- GND: Ground connection.

### 3.3. Connected modules with peripherals

These modules are comprised of binary decoders and 5V relays, all of which are directly controllable by the Arduino. Noteworthy features include:

1. Each 5V relay operates with a current of 20mA.
2. A direct 5V signal can be transmitted from the microcontroller.
3. Each relay is equipped with an illuminating LED for status indication.
4. Maximum relay specifications: 10A / 30VDC - 10A / 250VAC.

The primary function of these relays is to facilitate variations in passive components (such as resistors, capacitors, and coils) and to manage the switching of I/O connections within the Arduino system.

### 3.4. Experimental assembly.

The Ethernet module encompasses distinct power pins, as well as pins for data transmission, reception, and clock synchronization. The IPU library, tailored for Ethernet functionality, establishes a dedicated connection between the module and the Arduino, as illustrated in the table below (Table 1).

Within these practical exercises, students are engaged in conducting measurements on the Half Wave Single Phase Rectifier. This task involves altering the various passive components of the load (R, L, and C) through relay control. Signal generation is achieved via an Arduino-controlled power supply, with the measurement of voltages and currents facilitated through its analog inputs.

Table 1. Ethernet module and Arduino connection

| Ethernet module ENC28J60 | Arduino Mega2560 |
|---|---|
| CS | D53 |
| SI | D51 |
| SO | D50 |
| SCK | D52 |
| RESET | RESET |
| INT | D2 |
| VCC | 3V3 |
| GND | GND |

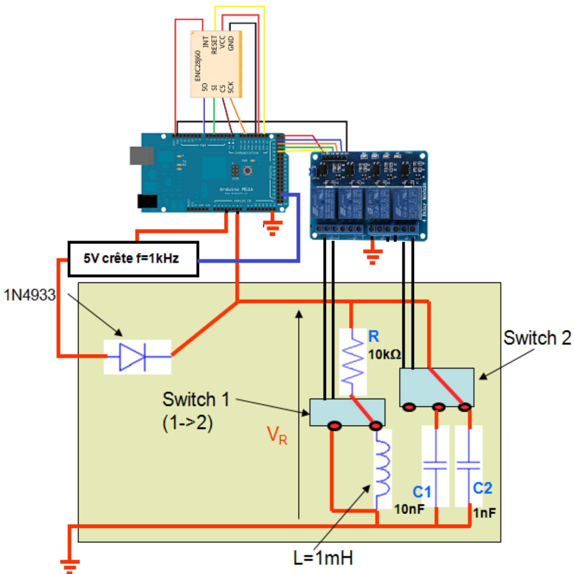


Figure 2. Connecting mounting Arduino Mega 2560 with Modules

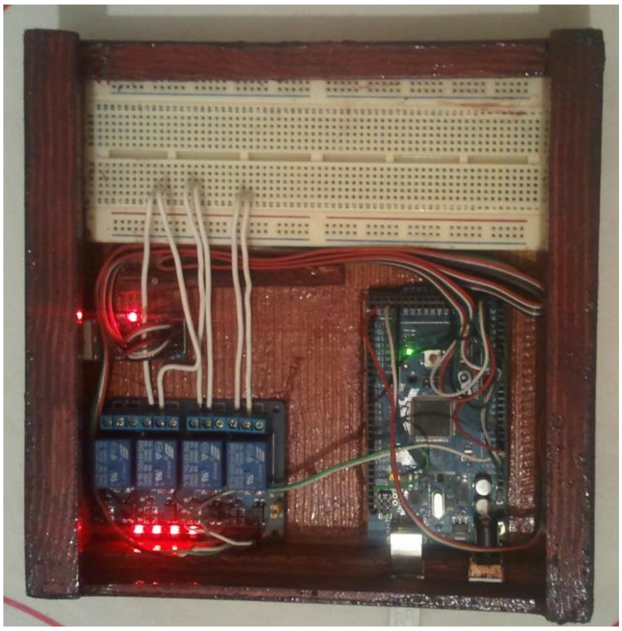

Figure 3. Realized electronic assembly.

## 4. SOFTWARE DESCRIPTION

### 4.1. Used PHP-languages and techniques

The first computer tool used in this work is the PHP. It is an interpreted language (a scripting language) executed on the server side (such as CGI scripts, ASP ...), not the client side (a script written in JavaScript or a Java applet....). The language syntax is derived from those of the C language, Perl and Java. Its main advantages are:

- A large community of developers sharing a big number of PHP scripting examples;
- Free access and availability of source code (PHP is distributed under GNU GPL);
- The simple write of scripts;
- The ability to include the PHP script within an HTML page;
- The simplicity of interfacing with databases (many DBMS are supported, but the most used with this language is MySQL, a free DBMS available on many platforms: Unix, Linux, Windows, MacOS X, Solaris, etc. );
- Integration into many web servers (Apache, Ms IIS, etc).

### 4.2. CSS style sheets

These Cascading Style Sheets are a language that describes the presentation of HTML documents. The CSS standards are published by the World Wide Web Consortium (W3C). The principle of style sheets is to bring together in one document formatting characteristics associated with groups of elements. Simply define by name a set of definitions and formatting features, and call to apply it to a text. Style sheets include:

• Using proprietary HTML extensions
• Converting text into images
• Using images for white space control
• Use of tables for page layout
• Writing a program instead of using HTML

These techniques considerably increase the complexity of Web pages, offer limited flexibility, suffer from interoperability problems, and create hardships for people with disabilities.

### 4.3. MySQL Database Management System

The MySQL database management system (DBMS) is employed here. It operates under a dual licensing model, encompassing both GPL and proprietary licenses. It stands among the world's most widely used database management software, serving not only the general public but also professionals, competing alongside Oracle, Informix, and Microsoft SQL Server. MySQL has been selected for this study due to its cost-free nature and its enhanced compatibility with web applications, especially those utilizing PHP. Its seamless integration with PHP further ensures optimal performance.

### 4.4. Apache Web Server

This software enables clients to access web pages, effectively HTML-format files, through a browser installed on their remote computers. The open-source Apache HTTP Server (Apache) is a renowned HTTP server developed and maintained by the Apache Foundation. It stands as the most popular HTTP server on the World Wide Web, distributed under the terms of the Apache License.

**4.5. Used Softwares**

a) WampServer:

This is a locally-operating web development platform of the WAMP type, standing for "Windows Apache MySQL PHP." WampServer, while not software itself, presents an environment hosting two servers (Apache and MySQL), a PHP script interpreter, and phpMyAdmin for the management of MySQL databases on the web. WAMP is essentially a variant of LAMP designed for Windows systems. Frequently, it's installed as a bundled software package comprising Apache, MySQL, and PHP. This configuration finds extensive application in web development and internal testing, while also capable of serving live websites.

Of utmost significance within the WAMP package is Apache, also known as the "Apache HTTP Server." This component is employed to operate the web server within the Windows environment. Running a local Apache web server on a Windows computer allows web developers to test webpages in a browser without the need to publish them live on the Internet.

b) Sublime Text:

Sublime Text is a versatile text and source code editor encompassing a Python application programming interface (API). This platform provides support for an extensive range of programming languages and markup languages, with the option for users to enhance its functionalities through plugins, often crafted and upheld by the community under free-software licenses. It's worth noting that Sublime Text is proprietary software, with all license proceeds directed to the developer. Nonetheless, the editor offers a spectrum of advanced features, including:

- Minimap: Offering a comprehensive file preview within a sidebar.
- Concurrent selection and editing in multiple code segments.
- Bookmarks for convenient navigation within files.
- Automatic backup functionality.
- Robust search and replace capabilities utilizing regular expressions.
- Extensive support for macros and Python-based plugins.
- Customizable keyboard shortcuts, allowing users to tailor the interface to their preferences.

c) Arduino EDI:

The integrated development environment, abbreviated as "IDE," serves as the software program tailored for the Arduino board. Facilitating an open-source environment, Arduino simplifies the coding process and facilitates uploading to the Arduino board. This platform operates seamlessly across Windows, Mac OS X, and Linux systems. Constructed using Java and based on Processing, avr-gcc, and other open-source software components, the IDE encompasses various functionalities. These usually encompass a graphical interface for accessing diverse tools, a source code editor, a compiler, a debugger, and often a utility for constructing graphical interface programs. The editor functionally mirrors that of a conventional text editor, the compiler assists programmers in identifying syntax errors within the source code, and the debugger is invaluable in identifying logical errors.

**4.6. Creation of the program**

Classes and functions used in the control program and acquiring electric quantities via the Arduino are:

- Setup (): this function is called only once after each start or reset the Arduino. It is often used to initialize variables, set the operating mode of the pins and initialize the library.
- Loop (): this function does exactly what its name suggests, it runs in a loop consecutively giving the program the opportunity to change and respond. It is used to actively control the Arduino board.
- PinMode (): Set the operating mode of a digital pin of the parameters we provided.
- AnalogRead (): it returns a value ente 0 and 1023 proportional to the value of a voltage between 0 and 5Volt, applied to a specified analog input.
- DigitalRead (): Plays the value of a voltage applied to a digital input previously defined pinMode (), it returns the value for a HIGH voltage of 5 volts and LOW value for a voltage of 0 Volt.
- DigitalWrite (): lets you apply a voltage of 5V or 0 volts, a digital release previously defined pinMode (), according to the given parameters.
- EthernetServer (): Creates a server that continues to listen for incoming connections on a specified port.
- Server.begin (): requests the server to start listening for incoming connections.
- Server.available (): returns a client that has data available to read.
- Ethernet.begin () initializes the Ethernet library and network settings.
- EthernetClient (): creates a client that can connect to a specified IP address and port.

- Client.connected (): checks whether the client is connected to the server.
- Client.available (): returns the number of bytes available to read.
- Client.print (): allows you to send serial data to the client
- Client.stop (): allows you to disconnect the client from the server.

Since the Ethernet library included with Arduino IDE to be incompatible with the ENC28J60 Ethernet Module we are going to use another library called Ethernet-UIP.

## 5. RESULTATS AND DISCUSSIONS

### 5.1. Program source code

In order to construct a program that effectively manages client requests, delivering responses or executing actions based on the received requests, our objective is to transform the Arduino into a server using the classes, methods, and functions elucidated earlier. The initial step involves assigning an IP address to our Arduino, rendering it identifiable within the local network. Subsequently, we proceed to establish a server tuned to port 80, geared to capture and process the queries dispatched by clients.

To institute a coherent system, each action will be denoted by a specific symbol, following these guidelines:

- To query an analog input, a distinctive symbol shall be employed, formed by the letter "i" succeeded by the corresponding lowercase alphabetical character. This character corresponds to the input's numerical designation; for instance, "a" represents 0, "b" represents 1, and so forth.
- Similarly, querying a digital input is represented by the combination of the letter "d" followed by the appropriate lowercase alphabetical character. This character aligns with the input's numerical designation. For inputs with numerals surpassing 25, capital letters will be employed, with "A" signifying 26, "B" signifying 27, and so on.
- Executing a 5-volt output to a digital pin is symbolized by a two-letter sequence, initiated by the letter "o" succeeded by the corresponding lowercase alphabetical character. Just like before, capital letters will be enlisted for outputs with numerals exceeding 25.
- Conversely, to apply a 0-volt output to a digital pin, a two-letter code is employed, commencing with the letter "0" followed by the corresponding lowercase alphabetical character. Similarly, capital letters will be used for outputs with numerals greater than 25.

The final source code is:

```
#include <UIPEthernet . h>
#include <UIPServer . h>
int mesure=17;
int s o r t i =33;
int d i g i t =33;
int inout=0;
byte mac[]={0 x54 , 0 x34 , 0 x41 , 0 x30 , 0 x30 , 0 x31 } ;
byte ip [ ]={ 1 9 2 , 1 6 8 , 1 , 2 0 } ;
byte r out e r [ ]={ 1 9 2 , 1 6 8 , 1 , 1 } ;
Ethe rne tSe rve r s e r v e r ( 8 0 ) ;
void setup ( ) {
pinMode (31 ,INPUT) ;
pinMode (13 ,OUTPUT) ;
S e r i a l . begin ( 9 6 0 0 ) ;
Ethernet . begin (mac , ip , r out e r ) ;
s e r v e r . begin ( ) ;
S e r i a l . pr int ( " IP Address :   " ) ;
S e r i a l . p r i n t l n ( Ethernet . l o c a l IP ( ) ) ;
}
void loop ( ) {
Ethe rne tCl i ent c l i e n t ;
c l i e n t=s e r v e r . a v a i l a b l e ( ) ;
i f ( c l i e n t )
{
boolean cur r entLine I sBl ank=true ;
while ( c l i e n t . connected ( ) )
{i
f ( c l i e n t . a v a i l a b l e ( ) ) {
char c=c l i e n t . read ( ) ;
S e r i a l . pr int ( c ) ;
```

```
i f ( c==’ \n ’ )
{ cur r entLine I sBl ank=true ; }
else i f ( c != ’ \ r ’ )
{ cur r entLine I sBl ank=fal s e ; }
i f ( cur r entLine I sBl ank ) {
S e r i a l . p r i n t l n (mesure ) ;
S e r i a l . p r i n t l n ( s o r t i ) ;
i f (mesure >=0 && mesure<16 && inout ==1)
{ c l i e n t . pr int ( analogRead (mesure ) ) ; }
i f ( s o r t i >=0 && s o r t i <=32 && inout==2)
{ d i g i t a lWr i t e ( s o r t i ,HIGH) ;
c l i e n t . pr int ( "ON" ) ;
}i
f ( s o r t i >=0 && s o r t i <=32 && inout==3)
{ d i g i t a lWr i t e ( s o r t i ,LOW) ;
c l i e n t . pr int ( "OFF" ) ;
}i
f ( d i g i t >=0 && d i g i t <=32 && inout==4)
{ c l i e n t . pr int ( di g i t a lRe ad ( d i g i t ) ) ; }
break ;
}i
f ( ! inout )
{if
(
c==’ i’
)
{ inout=1; }
i f ( c==’ o ’ )
{ inout=2; }
i f ( c==’O’ )
{ inout=3; }
i f ( c==’d ’ )
{ inout=4; }
}i
f ( c>=’ a ’ && c<=’p ’ && inout==1)
{ mesure=c−’ a ’ ; }
i f ( c>=’ a ’ && c<=’ z ’ && inout==2 | | inout==3))
{ s o r t i=c−’ a ’ ; }
i f ( c>=’A’ && c<=’G’ && ( inout==2 | | inout==3))
{ s o r t i=26+(c−’A’ ) ; }
i f ( c>=’ a ’ && c<=’ z ’ && ( inout==4))
{ d i g i t=c−’ a ’ ; }
i f ( c>=’A’ && c<=’G’ && ( inout==4))
{ d i g i t=26+(c−’A’ ) ; }
}}
delay ( 1 0 0 0 ) ;
c l i e n t . stop ( ) ;
inout = 0 ;
mesure = 1 7 ;
s o r t i = 3 3 ;
d i g i t = 3 3 ; }}
```

### 5.2. Deployment and Database

With our code now prepared, our next step involves compiling it to ensure the coding is error-free. Subsequently, we will establish a connection between our Arduino board and the computer using a USB cable, and then proceed to upload the program using the tools furnished within the Arduino IDE. Notably, the USB port of the computer serves as the power source.

It is essential to verify that all devices linked to the network possess distinct IP addresses, thus mitigating potential conflicts. With these preparations complete, our system is primed for use. To initiate requests, a web browser becomes our interface. This process entails entering the Arduino's IP address into the browser's address bar, accompanied by the designated symbol corresponding to the desired action. For instance, to retrieve the value captured by analog pin 0, the address would resemble: http://192.168.1.20/ia.

Up to this juncture, our focus has centered on establishing the connection between the Arduino and the computer, guaranteeing seamless data communication. However, it's important to acknowledge that the current methods and usability may lack practicality. As a remedy, the creation of a more efficient and user-friendly graphical interface is prudent.

To address this, a well-structured website is crafted, granting secure access to the Arduino while streamlining user management. Furthermore, this platform enables the allocation of specific users and facilitates configuration of the Arduino's IP address. Certain designated "user" entities hold the privilege of access authorization.

Table 2. Sessions Identification parameters

| Name | Type | Description | Login |
|---|---|---|---|
| Username | AN | Username | Yes |
| Password | AN | Password | No |
| Name | A | User | No |
| First name | A | User | No |
| Admin-Username | AN | Administrator | Yes |
| Admin-Password | AN | Administrator | No |
| Admin-Name | A | Administrator | No |
| Admin-Firstname | A | Administrator | No |
| URL | AN | IP Arduino address | No |
| Number-pin | N | N Arduino Pin | Yes |
| Type-pin | A | Arduino Pin | Yes |

**5.3. Conceptuel Data Model (CDM)**

Designers and analysts utilize the Conceptual Data Model (CDM) to illustrate the data relevant to the subject under consideration through the use of diagrams. Importantly, the CDM is independent of specific Database Management Systems (DBMS) or programming languages, adhering to a neutral approach.

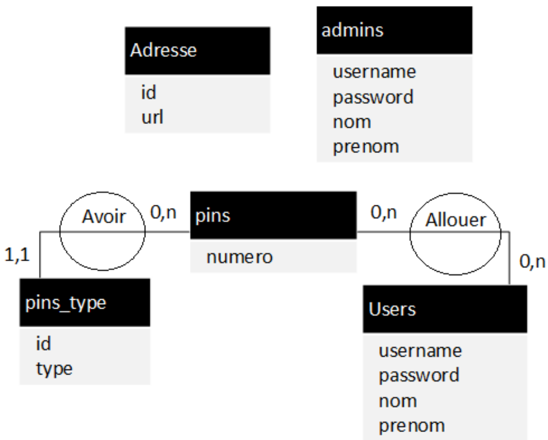


Figure 4. Diagram of Conceptual Data Model

**5.4. Logical Data Model (LDM)**

The Logical Data Model (LDM) follows the CDM stage chronologically, aiming to translate the entities outlined in the CDM into a format compatible with a Database Management System (DBMS). Within the realm of relational databases, these represented entities are transformed into tables (using SQL) along with the associated relationships between them. This ensures the seamless alignment of conceptual design with practical database implementation.

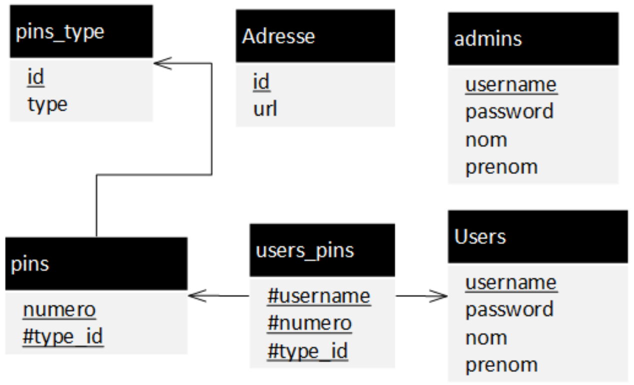


Figure 5. Diagram of Logical Data Model

It is important to highlight that the code examples furnished in this study are abridged renditions of the primary source code. This intentional simplification serves the purpose of refining the explanation process and enhancing the accessibility of comprehending the functionality in its entirety.

**5.5. SQL Code**

The process of creating an Arduino database involves:

```
CHARSET=ut f 8 ;
USE arduino ;
CREATE TABLE ‘adresse‘ (id int PRIMARY key ,‘ url ‘ varchar (80)) ;
INSERT INTO adresse VALUES( 1 , ’ http : / / 1 9 2 . 1 6 8 . 1 . 2 0 / ’ ) ;
CREATE TABLE ‘users‘ (
‘ username‘ varchar (20) PRIMARY KEY,
‘ password‘ varchar (60) NOT NULL,
‘nom‘ varchar (50) NOT NULL,
‘ prenom‘ varchar (50) NOT NULL
) ;
CREATE TABLE ‘ admins ‘ (
‘username‘ varchar (20) PRIMARY KEY,
‘ password‘ varchar (60) NOT NULL,
‘nom‘ varchar (50) NOT NULL,
‘ prenom‘ varchar (50) NOT NULL
) ;
CREATE TABLE ‘pins_type‘ (
‘ id‘ INT PRIMARY KEY,
‘type‘ VARCHAR(50) ) ;
CREATE TABLE ‘pins‘ (
numero INT NOT NULL,
type INT NOT NULL,
CONSTRAINT fk_pins_type FOREIGN KEY (type) REFERENCES pins_type ( id ) ,
CONSTRAINT pk_pins PRIMARY KEY( numero , type )
) ;
CREATE TABLE ‘ users_pins ‘ (
username VARCHAR(20) NOT NULL,
numero INT NOT NULL,
type INT NOT NULL,
CONSTRAINT fk_usr_pins FOREIGN KEY (username)
REFERENCES users (username) ON DELETE CASCADE,
CONSTRAINT fk_num_pins FOREIGN KEY (numero, type )
REFERENCES pins (numero, type) ON DELETE CASCADE,
CONSTRAINT pk_usrs_pins PRIMARY KEY( username , numero, type ).
```

### 5.6. Evaluation Results

The assessment of this setup is conducted through two distinct methods:

1. A survey involving 120 third-year electronic license students. These students participated in an informative online seminar focused on remote-controlled practical works, conducted in collaboration with international partners. This seminar was held in June 2014, coinciding with the inaugural Mediterranean e-learning and remote laboratories conference [10]. The students who partook in the survey responded to both open-ended and closed-ended questionnaires. The primary outcomes of this study are succinctly presented in Table 3.
2. The second evaluation approach involves a comparative analysis of results derived from an electronic practical work (PW) scenario, specifically concerning the measurement of a single-phase rectifier. A group of 15 students engaged in this PW exercise through two distinct methods. The first method involves the conventional hands-on approach, while the second method utilizes the newly developed remote system. The key findings are consolidated in Table 4 for easy reference.

Table 3. Survey Results

| **Questions** | **Yes (%)** | **No (%)** | **Do not know %** |
|---|---|---|---|
| **Already known that PW system?** | **41** | **52** | **7** |
| **This system is anti pedagogical?** | **25** | **63** | **12** |
| **This system can replace the old system of PW?** | **47** | **34** | **19** |
| **The objectives of PW are obtained by this system?** | **50** | **28** | **22** |
| **The role of the teacher is always present?** | **70** | **18** | **12** |
| **This system can solve the problems by Face to face PW?** | **79** | **13** | **8** |
| **This system is easy to handle?** | **60** | **30** | **10** |
| **This system is expensive?** | **48** | **41** | **11** |
| **This system can be used in other disciplines?** | **30** | **45** | **35** |
| **This system secure?** | **78** | **15** | **7** |

Considering the insights derived from Tables 3 and 4, it becomes evident that the remote PW system offers notable advantages, alongside certain inherent weaknesses. However, in an overarching perspective, it substantiates its potential suitability for integration within the existing educational framework, provided that the identified critical aspects are duly addressed. Importantly, this system holds the promise of effectively mitigating a primary concern within the present educational landscape - that of overcrowding.

Table 4. Practice results

| Parameters | Remote PW | Classic PW |
|---|---|---|
| Completion time of PW | 45min | 1h30min |
| Realized Sheet | 100% | 75% |
| Traceability | 100% | 20% |
| practical errors | 20% | 60% |
| Number of PW / 24h | 10 | 1 to 2 |
| freedom of PW realization | 24h | 4h |
| Sensation of practical embodiment | 60% | 90% |
| Touching the equipment and instrument | 10% | 100% |
| Understand the purpose of the PW | 70% | 70% |

## 6. CONCLUSION

This work stemmed from a genuine demand to enhance practical teaching, particularly within open-access faculties. Our endeavor centered on conceptualizing a remote educational apparatus tailored for university electronics laboratories. At the core of this accomplished system lies an economical embedded platform - the Arduino Mega2560 - boasting commendable specifications. The computational facet of this innovation is meticulously devised employing accessible and straightforward tools, characterized by their open-source nature.

University students are empowered to engage with this system via either local network connectivity or WiFi, granting them flexible access throughout the Faculty premises. This work comprehensively elucidates the distinct constituents of this innovation. To validate its reliability, rigorous assessments were performed, involving cohorts of third-year university students. A comparative analysis of Practical Work (PW) outcomes between our system and the conventional approach was conducted, yielding insightful results.

These findings not only endorse the viability of our developed system but also underscore its potential for broader application. This solution holds the promise of addressing specific challenges frequently encountered within the contemporary education system.

**REFERENCES**


[1] Amine Moulay Taj, Jesús Chacón, Luis de la Torre, Abdessamad Malaoui, Sebastián Dormido, “ An architecture to implement generalized sampling in Online Laboratories”, IFAC-PapersOnLine, Volume 55, Issue 17, pp. 332-337,2022.

[2] Tempus “EOLES” granted by the European program TEMPUS “530466-1-2012-FR-TEMPUS-JPCR”. 2012-2015.

[3] Abdessamad Malaoui, Monji Kherallah, Lila Ghomri, Mustapha Raoufi, Andrieu Guillaume, Fredon Thomas, Barataud Denis. "Implementation and Validation of a new Strategy of Online Practical Works of Power Electronics for Embedded Systems". International Journal of Online Engineering (iJOE), V(3), N°4, pp.29-44, 2017.

[4] M. Gericota, D. Barataud, , G. Andrieu, R. De Craemer, M. Cristea, A. Benachenhou, M. Ankrim, K. Bouchlaghem, P. Ferreira”, The EOLES project”, Global Engineering Education Conference (EDUCON), 2014 IEEE, pp. 943 - 946, 2014.

[5] Chamunorwa, Tinashe & Modran, Horia & Ursutiu, Doru & Samoilă, Cornel & Hedeșiu, Horia. Reconfigurable Wireless Sensor Node Remote Laboratory Platform with Cloud Connectivity. Sensors. 21. 6405. 10.3390/s21196405. (2021).

[6] A. Moulay Taj, A. Abouhilal, N. Taifi, J. Antari, A. Malaoui. "Low-cost Smart Architecture for Classroom Response System Using Raspberry: RaspCRS".International Journal of Engineering & Technology; Vol 7, No 4.32, 2018.

[7] E. D. Lindsay, M. C. Good. “Effects of laboratory access modes upon learning outcomes”. Education, IEEE Transactions on, 48(4), pp. 619-631, 2005.

[8] A. Agrawal, S. Srivastava. "WebLab: A Generic Architecture for Remote Laboratories ", Advanced Computing and Communications, ADCOM 2007. International Conference on, pp. 301-306, 2007.

[9] Malaoui Abdessamad, " Embedded Electronics Applied in Remote Labs and Practical Works: A technological Revolution in the Future, What Limits and Impacts?," ICSES Transactions on Computer Hardware and Electrical Engineering (ITCHEE), vol. 4, no. 1, pp. 1-2, 2018.

[10] Abdelmoula ABOUHILAL, Amine Moulay Taj, Naîma Taifi, Abdessamad Malaoui. "Embedded Electronics Applied in Remote Laboratories Using NodeJs". International Journal of Èmerging Trends in Engineering Research. Vol. 8. N 5, May 2020.