Université Sidi Mohamed Ben Abdellah-Fès

École nationale des sciences appliquées de Fès

Département : Génie électrique et informatique

Génie Systèmes Embarqués et Informatique Industrielle

Smart Room Temperature Control System



2022/2023

Sous la supervision :

Pr. Said Hraoui

Realisé par :

Mohammed Handa ILYAS NHASSE

# Remerciements :

Nous tenons à vous exprimer notre sincère gratitude et nos plus chaleureux remerciements pour votre précieuse contribution et votre soutien tout au long de notre projet PFA. Votre expertise, vos conseils éclairés et votre soutien ont été essentiels à la réalisation de notre rapport, et nous tenions à vous témoigner notre profonde reconnaissance.

Votre expertise approfondie dans IoT a été d'une grande valeur pour nous. Vos commentaires perspicaces et vos recommandations pertinentes ont contribué à améliorer la qualité de notre rapport et à renforcer la rigueur de nos analyses. Grâce à vos connaissances approfondies, nous avons pu explorer en profondeur les différentes facettes du projet et présenter des résultats pertinents.

Nous sommes également reconnaissants pour votre disponibilité et votre ouverture à la discussion tout au long de notre collaboration. Vous avez toujours été prêt à écouter nos idées, à répondre à nos questions et à partager vos réflexions, ce qui a grandement enrichi notre compréhension du sujet. Votre soutien constant et vos encouragements ont été une source de motivation pour nous.

Nous tenons également à souligner l'impact positif que vous avez eu sur notre développement académique et professionnel. Vos conseils avisés et votre mentorat ont contribué à renforcer nos compétences et à élargir notre horizon. Grâce à vous, nous avons acquis une expérience précieuse et avons grandi en tant que professionnels.

Enfin, nous tenons à vous remercier pour votre attention aux détails et votre souci de l'excellence. Votre expertise a été cruciale pour veiller à ce que notre rapport soit complet, bien structuré et conforme aux normes de qualité les plus élevées. Votre contribution a considérablement amélioré la qualité globale de notre travail.

Table of content

[I. General introduction 2](#_Toc135053738)

[II. General context 3](#_Toc135053739)

[1. Context 3](#_Toc135053740)

[2. Specifications 3](#_Toc135053741)

[3. Methodology 4](#_Toc135053742)

[a) Components 4](#_Toc135053743)

[b) Assembly 5](#_Toc135053744)

[c) Development environment 5](#_Toc135053745)

[d) Development lifecycle 5](#_Toc135053746)

[III. Design and Modeling 6](#_Toc135053747)

[1. Technical environment 6](#_Toc135053748)

[2. General Architecture 7](#_Toc135053749)

[3. Detailed design 8](#_Toc135053750)

[IV. Realization and testing 12](#_Toc135053751)

[1. Realization 12](#_Toc135053752)

[2. Testing 13](#_Toc135053753)

[a) Unit test 13](#_Toc135053754)

[b) Component test 13](#_Toc135053755)

[V. General conclusion 14](#_Toc135053756)

[VI. Webography 15](#_Toc135053757)

# General introduction

In modern times, the demand for convenience and efficiency is increasingly prominent, and manual tasks can prove to be laborious and time-consuming. For buildings with numerous windows that require daily manual opening and closing, this poses a significant challenge, as it can not only involve physical exertion but also pose safety risks, especially for high-rise buildings.

Moreover, maintaining a comfortable indoor environment in terms of temperature can be an ongoing struggle. During summer, windows may require opening at night and closing right before the temperature starts rising to keep the space as cool as possible, while in winter, they need to remain closed to conserve heat. This can be a daunting task for occupants, who must continually monitor and adjust the windows to sustain a comfortable environment.

Fortunately, technology offers a viable solution to this problem - the automated window control system. This system allows for convenient control of the opening and closing of windows based on user preferences or weather conditions. This technology not only enhances convenience and saves time and effort but also helps in passively cooling buildings which saves an important amount of energy.

# General context

## Context

The automated window control system provides several benefits for building occupants and managers. Firstly, it offers convenience and ease of use, as occupants can remotely control the opening and closing of windows from a central control unit or even their smartphones. This eliminates the need for manual operation, which can be time-consuming and physically demanding, especially in tall buildings.

Secondly, the automated window control system enables the building to maintain a comfortable indoor environment consistently. With its ability to detect weather changes and adjust the windows accordingly, it ensures that the building remains cool during the summer and warm during the winter. This results in enhanced occupant comfort, satisfaction, and productivity, which are essential factors in the success of any building.

Thirdly, the automated window control system is an energy-efficient solution, as it reduces the amount of energy required to cool or heat a building. This, in turn, translates to cost savings on energy bills and reduced carbon emissions. Additionally, the automated system ensures that windows are not left open unnecessarily, preventing unwanted heat loss or gain and improving energy efficiency further.

Overall, the automated window control system is a valuable addition to any building, as it offers numerous benefits such as convenience, comfort, energy efficiency, and cost savings. It presents an excellent opportunity for building managers and occupants to create a sustainable and comfortable indoor environment while reducing the building's carbon footprint.

## Specifications

The proposed project aims to develop an automated window control system to address the challenge of manually opening and closing windows in buildings. The system's primary objective is to enhance building occupant comfort, convenience, and safety while promoting energy efficiency and sustainability.

The system's specifications include a central control unit (Raspberry Pi) that integrates with the building's HVAC (heating, ventilation, and air conditioning) system and the windows.

The system aims to solve the following problems:

* Physical strain and safety risks: Manual opening and closing windows can be physically demanding.
* Inefficient temperature control: Maintaining a comfortable indoor temperature requires careful management of windows.
* Time-consuming: Manually opening and closing windows can be time-consuming.
* Energy waste: Poor windows management can result in air conditioners working harder, therefore, wasting more energy.

The system has several objectives, including:

* Offer a user interface to view statistics and configure the system
* Automate the opening and closing of windows
* Maintain a comfortable indoor environment
* Improve building energy efficiency
* Reduce carbon emissions

In summary, the proposed automated window control system seeks to offer a cost-effective, convenient, and energy-efficient solution to the challenges of manually opening and closing windows in buildings. By achieving its objectives, the system will enhance occupant comfort, promote building safety, and contribute to creating more sustainable environments.

## Methodology

### Components

The automated Windows control system composes of:

* Control unit: Raspberry Pi 4
* Temperature and humidity sensors: DHT11 module
* Stepper motors: 28byj-48 stepper motor

The basic example of AWCS functionality will contain:

* 1 x Raspberry Pi 4
* 2 x DHT11 modules
* 1 x 28byj-48 stepper motor

### Assembly

The system consists of two DHT11 modules, one placed inside the target room and the other outside. A stepper motor will be attached to the window, enabling both automatic and manual control. The Raspberry Pi 4 will be located inside the room in close proximity to the windows, ensuring stable communication with the other components. Through this system, we aim to provide an efficient and reliable solution for regulating indoor temperature and humidity levels.

### Development environment

The system comprises a Raspberry Pi 4, running on the Raspberry Pi OS (aarch64) operating system, which is responsible for controlling a DHT11 module through a custom driver based on the 1-wire protocol. Additionally, the Raspberry Pi 4 controls a stepper motor, allowing for automatic and manual control of the target window. To develop the drivers and main program, we employed the C programming language, while Python was used for unit, component, and integration testing. Visual Studio Code was used as the development environment, connected to the Raspberry Pi 4 through the SSH protocol. Finally, the GCC compiler was used to generate the executable files for the system. This approach provides an efficient and reliable solution for indoor temperature and humidity regulation.

### Development lifecycle

The project was executed following the V-Model, a software development methodology that emphasizes the importance of testing and verification throughout the entire development process. The V-Model consists of a series of stages, including requirements analysis, design, implementation, testing, and maintenance.

# Design and Modeling

## Technical environment

The software/hardware development environment of the automated window control system project involves the following technologies, languages, and platforms:

* Raspberry Pi 4: This is a small, low-cost computer used as the main control unit for the system. It runs on a Linux-based operating system and has several GPIO pins that allow for easy interfacing with other hardware components.
* DHT11 Temperature and Humidity Sensor: This is a digital sensor used to measure the temperature and humidity levels in the environment. It is connected to the Raspberry Pi using the GPIO pins.
* 28byj-48 Stepper Motor: This is a small, low-cost stepper motor used to control the opening and closing of the windows. It is connected to a motor driver module, which is also connected to the Raspberry Pi.
* Python Programming Language: Python is a popular programming language used for developing the control algorithm for the automated window control system. It is easy to learn and has several libraries that simplify interfacing with the hardware components.
* Raspberry Pi OS: This is the operating system installed on the Raspberry Pi, which is a Debian-based Linux distribution specifically designed for the Raspberry Pi hardware.
* Integrated Development Environment (IDE): An IDE, such as Visual Studio Code (VSCode), can be used to write and test the code. Since the Raspberry Pi will be used as the main control unit, it is possible to use VSCode with SSH to connect remotely to the Raspberry Pi and edit the code directly on the Pi. This allows for a more seamless development experience, as developers can use the same familiar tools and workflows they use for local development, but with the added benefit of being able to work directly on the target system. This approach also eliminates the need to transfer files back and forth between the development machine and the Raspberry Pi, simplifying the development process.

In summary, the software/hardware development environment of the project involves the use of the Raspberry Pi 4 as the main control unit, DHT11 temperature, and humidity sensors to measure the environment's conditions, and 28byj-48 stepper motors to control the opening and closing of the windows. Python programming language is used to develop the control algorithm, and an IDE is used to write and test the code. The project runs on the Raspberry Pi OS, which is a Debian-based Linux distribution designed for the Raspberry Pi hardware

## General Architecture

The software for this project will cover the objective mentioned in the specification which are

* User interface
* Window control
* Temperature control.
* Energy efficiency
* Carbon emissions reduction

To achieve the desired automation of window control, it is essential to develop two distinct applications: a client and a server.

The server will be able to:

* Get the temperature from the DHT11 modules
* Operate the stepper motor connected to the window
* Schedule operations to execute later
* Autonomously control the windows based on temperature, time, and other conditions
* Save temperature records to a database

The server must be capable of receiving user inputs transmitted by the client, processing these inputs, and selecting the appropriate program to execute. The server must then obtain the program's result and relay it back to the client.

The client will be able to:

* Communicate with the server
* Get statistics about the temperature.
* Ask the server to control the window (close or open)
* Set operations schedule

The system must enable the reception of user inputs by the client, which will perform validation logic to ensure the reliability of the received inputs before transmission to the server. The client must also be able to receive the server's response and present it to the user in a user-friendly manner.

User interface example scenario:

* The user wants to keep a room as cool as possible during a warm day.
* The user opens a webpage on his computer or smartphone
* The user checks the option “Keep as cool as possible”
* The user presses apply and the inputs are sent to the server

Server example scenario:

* New inputs are received from the client
* The server processes the inputs and decides what program to execute
* The new inputs are the option “Keep as cool as possible”
* The server executes an executable called “keep\_as\_cool\_as\_possible”
* The executable contains the logic to close and open the window to maintain an optimal temperature (more about the different options in the next chapter)

## Detailed design

To achieve the objectives of the automated window control system, the system is composed of six different entities. The first entity is the client, which represents the web browser utilized by the user, and is the only visible component of the system. The second entity is the server, which is a PHP server hosted on the Raspberry Pi, and is responsible for receiving user inputs from the client and forwarding them to the appropriate program. The third entity is the control program, which is a set of programs on the Raspberry Pi, each with specific functionality. The fourth entity is the drivers, which are a set of programs that allow the control programs to interface more easily with the hardware. The fifth entity is the sensors, which are devices, modules, machines, or subsystems that detect events or changes in the environment and send the information back to the control unit. In this case, the DHT11 module is being used as a sensor. Finally, the sixth entity is the operator, which is a device that can perform specific actions and operations on the environment. In this project, a motor is used as an operator.

To better understand the relation between all entities we can consider the following sequence diagram, which represents a general events flow after a user input

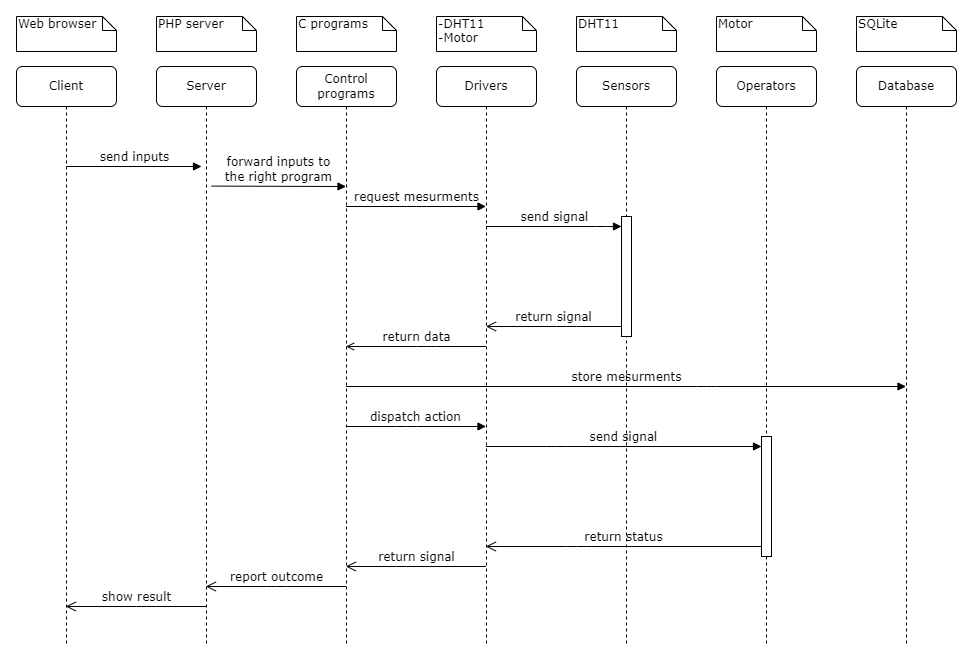


Figure 1 : Action sequence diagram

There are two control programs:

1. Control\_window

The objective of this program is to automate the process of closing or opening a window.

The program receives the window's current state as an input, where 1 represents an open window and 0 represents a closed one. Based on this input, the program operates the stepper motor to achieve the desired behavior.

To facilitate this process, the program utilizes the 28byj-48.h stepper motor driver which provides a straightforward interface to control the motor's movement. This driver abstracts the hardware complexities by allowing us to specify the number of steps required for the motor to rotate.

The functionality of the 28byj-48 stepper motor driver is presented in the chart below:

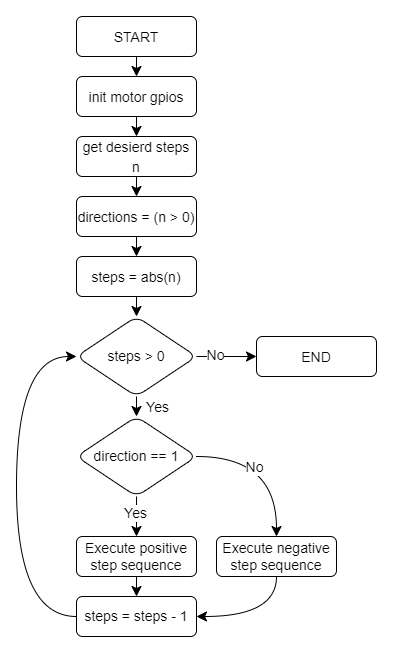


Figure 2 : 28byj-48 driver flow chart

1. Record\_temp

This program aims to obtain the current temperature of a room through the use of a DHT11 module and subsequently store it as a new entry in a database. The program requires the path of the database as an input, and upon receiving it, attempts to obtain the temperature using the DHT11.h driver. If the initial reading fails, the program retries a few more times until a valid reading is obtained. Once a reading is acquired, the program accesses the database and adds the new temperature reading along with the current date. The program leverages the DHT11 module with the assistance of the (DHT11.h) driver, which permits direct access to a function that returns the temperature. The chart below illustrates the functionality of the DHT11.h driver.

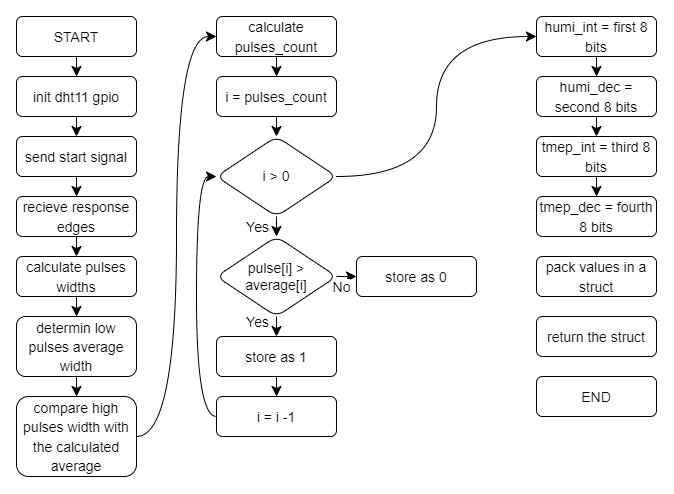


Figure 3 : DHT11 driver flow chart

# Realization and testing

## Realization

To realize this project, we used

* Raspberry Pi 4 as a development board
* PHP as a web server
* SQLite as a database
* C as a low-level language
* Python as a testing language
* DHT11 as a temperature sensor
* 28byj-48 as a stepper motor

The schema below represents the overall relations between each entity

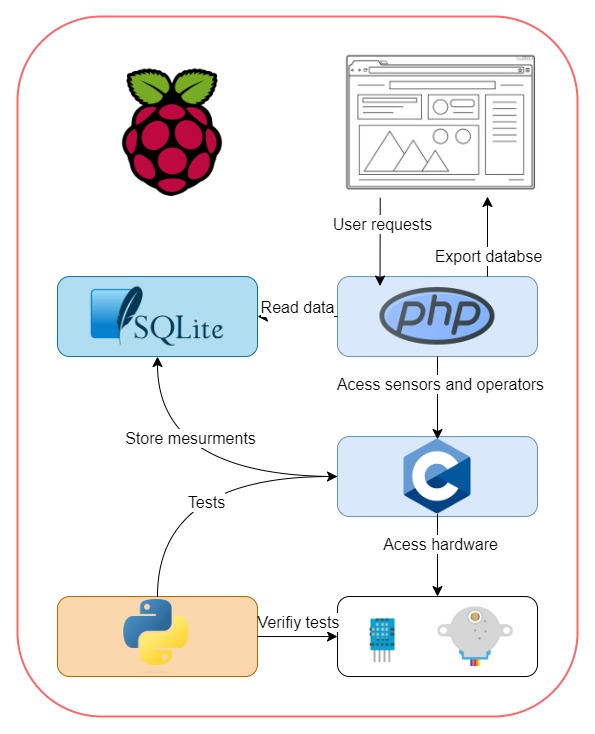


Figure 4 : Realization schema

## Testing

To validate the final product, we need the three tests on the V-Model

### Unit test

For this project we can’t test individual functions because to get comparable results we need the whole component to be functioning, for this reason, unit tests are merged with component tests

### Component test

1. DHT11 module

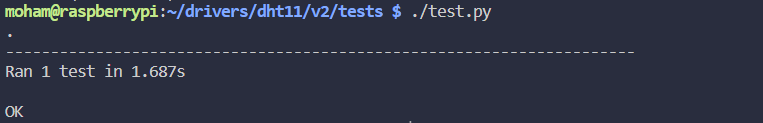


Figure 5 : DHT11 component test

The above screenshot is the output of a unit test Python script for the DHT11 driver. The test uses a c program that interfaces with the driver and prints the humidity and temperature to the standard output, the python script then reads the output and compares it with the readings from the “Adafruit\_DHT” library. If the results match the test passes otherwise the test fails

1. 28byj-48 motor



Figure 6 : 28byj-48 component test

The above program tests the 28byj-48 motor by rotating it one time clockwise and two times anti-clockwise

# General conclusion

In conclusion, this project was aimed at designing and implementing an automated window control system using Raspberry Pi, PHP, SQLite, C, and Python. The system is composed of several entities, including a client, server, control programs, drivers, sensors, and operators, all working together to achieve the desired functionality. The automated window control system is designed to open or close windows based on the user's input and the current room temperature, which is monitored using a DHT11 temperature sensor.

The system was tested and validated using a series of scenarios that included both component and integration tests. These tests were designed to ensure that the system works as expected and meets the project's initial requirements. The results showed that the automated window control system performed well in different situations and was able to respond to user inputs and temperature changes effectively.

Overall, this project demonstrates the potential of using a Raspberry Pi and a combination of programming languages and technologies to create a functional and efficient automated window control system. The system could be further improved by adding more sensors or integrating it with other home automation systems to provide a more complete solution for home automation.

# Webography

<https://github.com/adafruit/Adafruit_Python_DHT>

<https://www.wikiwand.com/en/Memory-mapped_I/O_and_port-mapped_I/O>

<https://lastminuteengineers.com/28byj48-stepper-motor-arduino-tutorial/>

<https://www.php.net/docs.php>

<https://www.w3schools.com/sql/>

<https://sqlite.org/index.html>

<https://github.com/torvalds/linux>

<https://github.com/raspberrypi/linux>