**PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA**
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
SAAD DAHLEB UNIVERSITY OF BLIDA 1
FACULTY OF TECHNOLOGY - AUTOMATIC DEPARTMENT

جامعة البليدة 1
UNIVERSITE BLIDA 1

# DAMS

distribution arm with monitoring system

UNIVERSITY OF BLIDA — 2023

*Created by :*
**MOHAMAD LOUNNAS**
*Framed by :*
**KAZED BOUALAME**

# Thanks

I extend my heartfelt thanks to all who have supported me throughout my academic journey. I am grateful to my family for their unwavering love and belief in my abilities. To my professors, mentors, and advisors, thank you for your guidance and nurturing my intellectual growth. I am grateful to my classmates and friends for the shared experiences and camaraderie. Appreciation goes to the staff and administration for creating a conducive learning environment. Lastly, I am inspired by those who have made a difference in their fields. Thank you all for being a part of my academic success.

# Summary


عند العمل مع رقائق صغيرة مثل أردوينو و**ESP32** ، يمكن أن تجعل حدود **الذاكرة** وسرعة المتحكم عملية تصحيح الأخطاء والمتابعة صعبة للغاية.

خصوصا في البيئات التي قد لا تعمل أدوات التطوير المعروفة فيها بفعالية أو لست مخصصة لها على الإطلاق. في هذا المشروع، قمت بتطوير نظام مراقبة وتصحيح اخطاء باستخدام **MicroPython** وبروتوكول **I2C** وبعض مبادئ تصميم البرمجيات لتمكين عملية تصحيح الأخطاء ومراقبة كل مكون.

When working with small chips like **Arduino** and **ESP32**, limitations in memory and controller speed can make debugging challenging.

Especially in environments where development tools may not work effectively. In this project, I developed a system using **MicroPython**, the **I2C bus**, and **design principles** to enable debugging and monitoring of each component.

Lorsque l'on travaille avec de petites puces comme **Arduino** et **ESP32**, les limitations de mémoire et de vitesse du contrôleur peuvent rendre le débogage difficile. Surtout dans des environnements où les outils de développement peuvent ne pas fonctionner efficacement. Dans ce projet, j'ai développé un système utilisant **MicroPython**, le bus **I2C** et des principes de conception pour permettre le débogage et la surveillance de chaque composant.

# Lists of figures:

# Index

# General Introduction

The final result of the project is an arm equipped with a scale and a cup holder. The purpose of the arm is to retrieve cups that are placed on top of a tall structure (referred to as the "tscall"), transfer them under a reservoir, and pour a specific material into the cup when it is open. The arm then evaluates the balance of the scale and returns the cup to the user.

Once the cup is in position, the reservoir opens, and the desired material flows down into the cup. The arm takes advantage of an ultrasonic sensor located on the arm to detect whether there is a cup present on the scale or not. This feature enables the system to ensure that a cup is present before proceeding with the pouring process.

After the material has been dispensed into the cup, the arm evaluates the balance of the scale. If the weight of the cup, including the material, exceeds a predefined threshold (balcne + 5g), indicating that the cup is too heavy or oversized, the arm will automatically discard the cup by throwing it away.

On the other hand, if the weight is within the acceptable range, the arm carefully lifts the cup and returns it to the user. This action completes the cycle of retrieving, filling, and delivering the cup.

By incorporating the scale, cup holder, and ultrasonic sensor into the design, the arm system is capable of autonomously managing the cup handling process. The integration of these features provides an efficient and automated solution for dispensing materials into cups, ensuring proper balance, and discarding cups that exceed the weight threshold.

In addition to the functionalities described earlier, the ESP8266 microcontroller plays a crucial role in controlling and monitoring the entire system. The ESP8266 is a popular and powerful microcontroller with built-in Wi-Fi capabilities, making it an ideal choice for network communication and data transfer.

Using the ESP8266, the system can be controlled remotely via a network connection. This means that users can interact with the arm system through a user interface or application, sending commands to initiate specific actions such as retrieving cups, pouring materials, or discarding cups. The ESP8266 receives these commands over the network and translates them into actions performed by the arm.

Furthermore, the ESP8266 facilitates real-time monitoring of the system's status and performance. It collects data from various sensors, including the ultrasonic sensor, scale, and any other relevant sensors integrated into the system. This data can include cup presence on the scale, weight measurements, and other relevant parameters.

The ESP8266 can then transmit this data over the network, allowing users to remotely monitor the system's operations and receive real-time updates. The data can be sent to a server or displayed on a user interface, enabling users to keep track of cup handling, material pouring, and any other relevant information.

By leveraging the ESP8266's networking capabilities, the system becomes more accessible and user-friendly. Users can control and monitor the arm system from anywhere with a network connection, facilitating remote operation and oversight. This feature adds convenience and flexibility to the overall functionality of the system, making it a versatile solution for automated cup handling and material dispensing tasks.

# Chapter 01: mechanical part

## 1) Arm mecanism

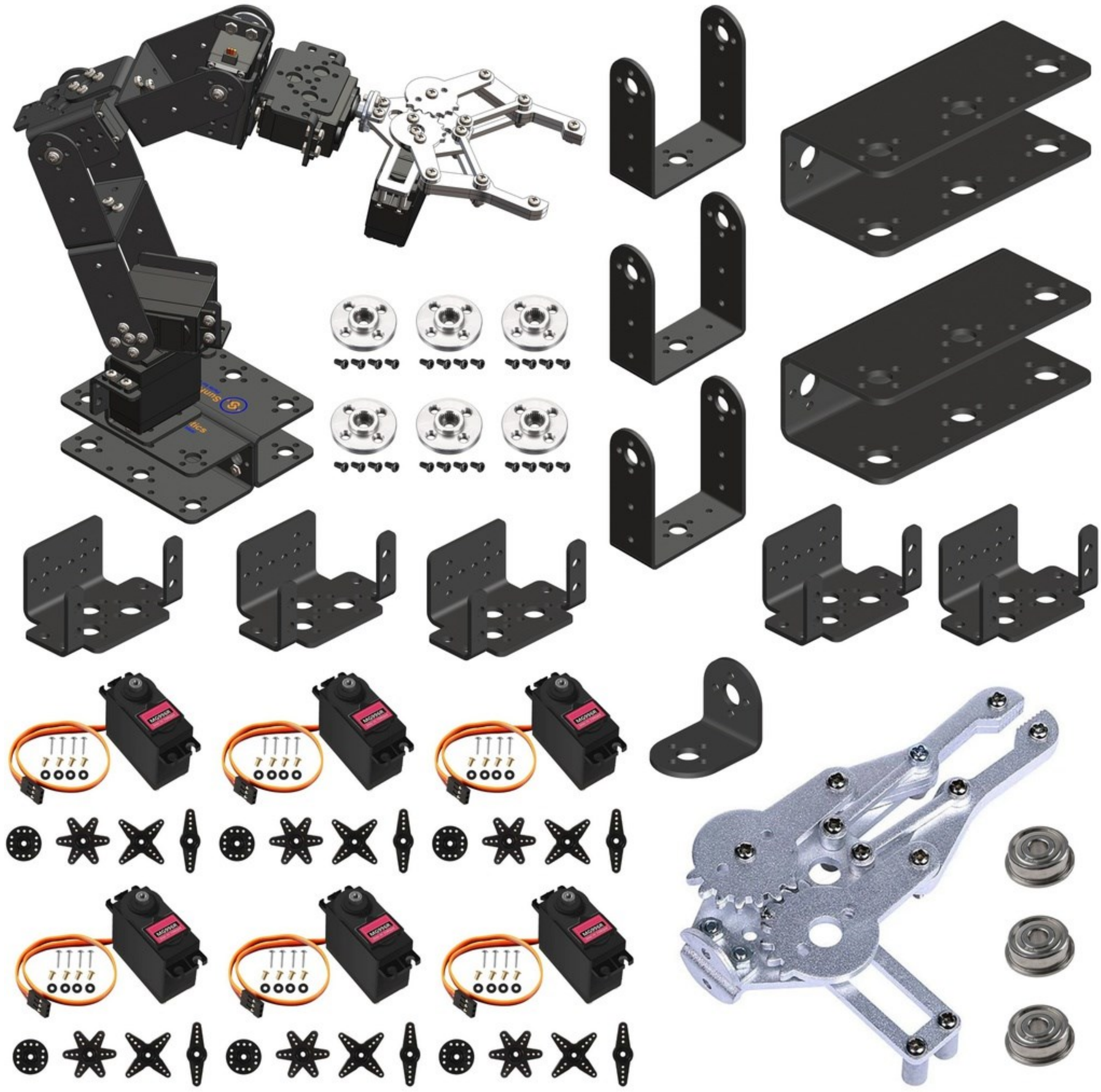

*Figure 1: Arm mecanism*

We utilized a highly versatile arm model in our project, which proved to be instrumental in reducing the layout construction time significantly. The arm's customizable nature allowed us to effortlessly add or remove various parts and integrate it with other components seamlessly. This flexibility played a vital role in adapting the arm to our specific requirements and optimizing its performance within the project's scope. By leveraging the advantages of this highly customizable arm, we were able to streamline our workflow and achieve efficient and tailored results.

## 2) Pictures of realization

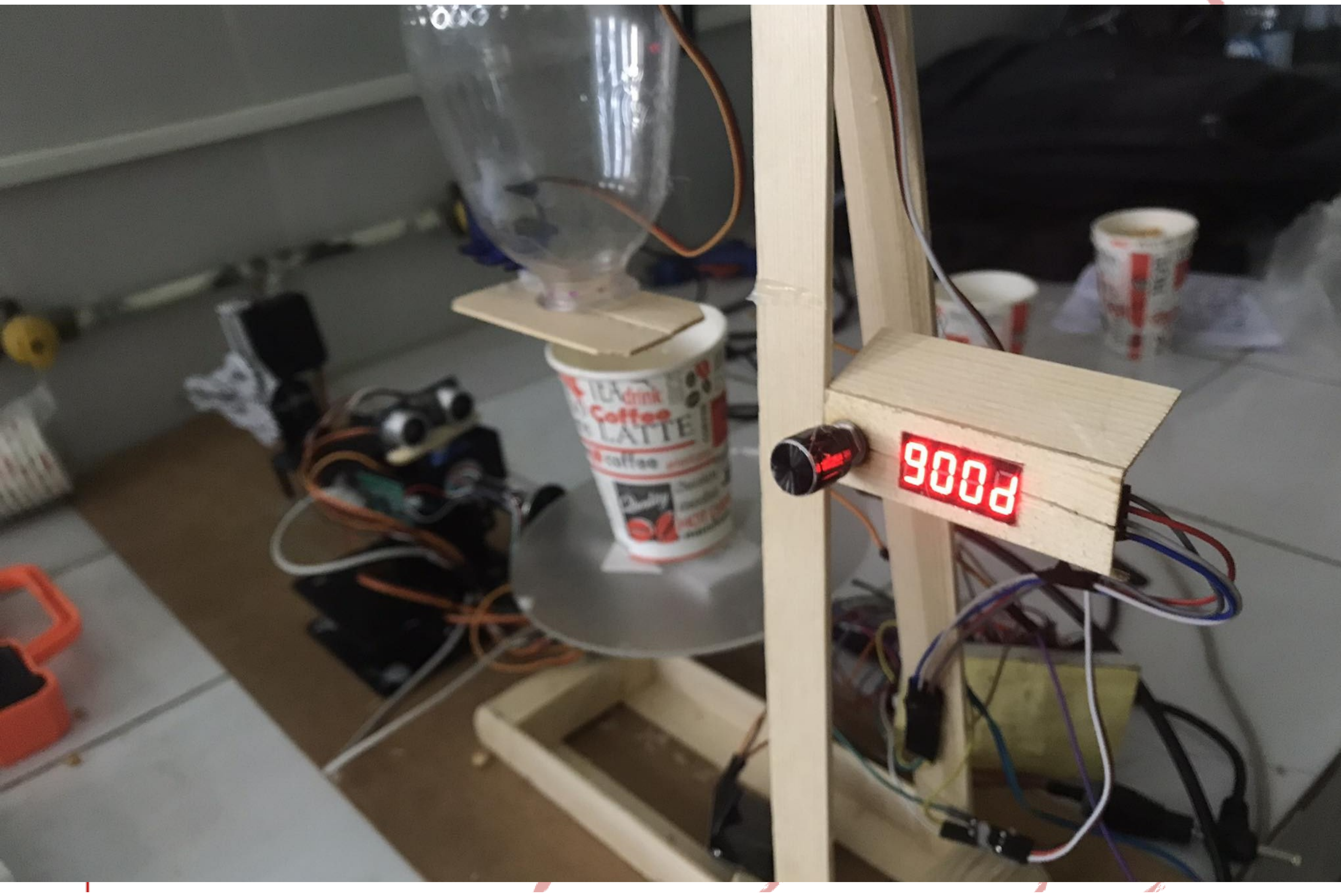


*Figure 2: Pictures of realization*

MOHAMAD LOUNN

# Chapter 02: Electronic part

## 1) Introduction

In this chapter, we explore the essential electronic components that make up the core of our cup handling and material dispensing system. These components include the TM1637 Display, the HX711 ADC, the ultrasonic sensor, the Arduino Uno, and the ESP8266 microcontroller. Each component plays a crucial role in enabling the functionality, control, and monitoring of our system.

## 2) used components

### 2.1 TM1637 Display

The TM1637 Display serves as a vital visual interface, providing clear and concise information to the user. We will discuss its integration and utilization within our system, highlighting how it enhances the user experience by displaying relevant data such as system status, measurements, and prompts.


*Figure 3: TM1637 Display*

### 2.2 HX711 ADC

The HX711 ADC (Analog-to-Digital Converter) module is responsible for accurately measuring the weight on the scale. We will delve into its operation and calibration process, emphasizing how it interfaces with the scale and communicates weight data to the microcontroller. Understanding the HX711 ADC's role will allow us to ensure precise and reliable weight measurements.

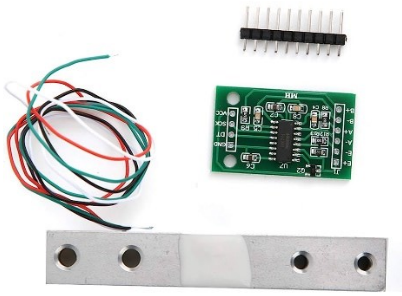

*Figure 4: HX711*

### 2.3 ultrasonic sensor

The ultrasonic sensor, another key component, enables the system to detect the presence of cups on the scale. We will explain its principles of operation, focusing on how it uses sound waves to measure distances and detect objects. By comprehending the ultrasonic sensor's functionality, we can ensure efficient cup detection and enable appropriate actions within the system.


*Figure 5: ultrasonic sensor*

### 2.4 Arduino Uno

The Arduino Uno microcontroller serves as a fundamental building block, responsible for controlling the various electronic components and executing the system's logic. We will discuss its capabilities, programming environment, and how it interfaces with the other components to coordinate the cup handling and material dispensing operations. Understanding the Arduino Uno's role will provide insight into the system's control and coordination mechanisms.


*Figure 6: Arduino Uno*

## 2.5 ESP8266 microcontroller

Lastly, the ESP8266 microcontroller empowers our system with network connectivity and remote control capabilities. We will explore its features, programming options, and how it facilitates communication between the system and external devices or user interfaces. By utilizing the ESP8266, we can control and monitor our system remotely, enhancing its accessibility and usability.[1]

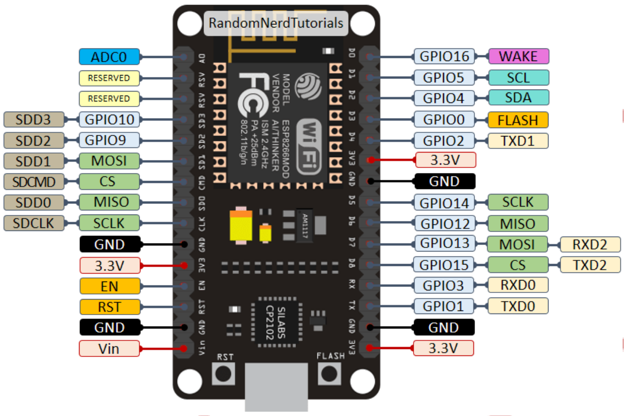


*Figure 7: ESP8266*

## 2.6 Other helpers components

We used a bench with push buttons, resistors, wires, a breadboard, and other basic components.

---

[1] https://arduino-esp8266.readthedocs.io/en/2.5.2/reference.html

## 3) Schematic Diagrams

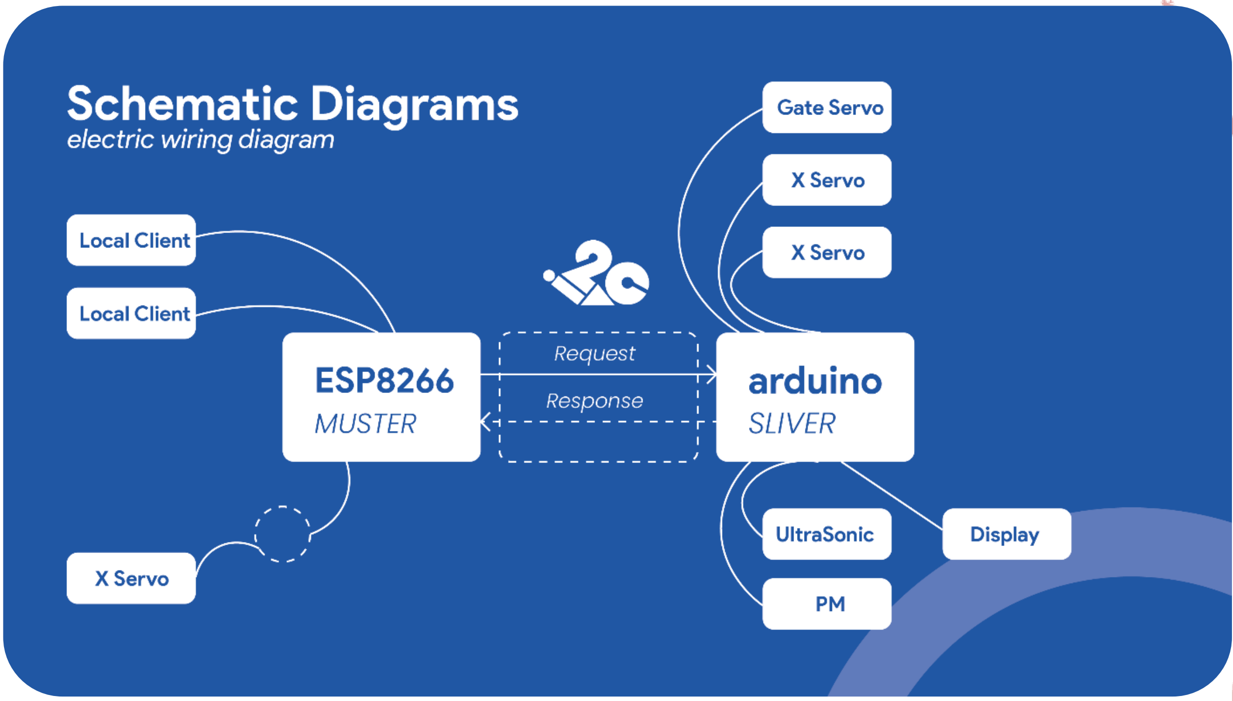


*Figure 8: Schematic Diagrams*

## 4) conclusion

Through the examination of these electronic components - the TM1637 Display, HX711 ADC, ultrasonic sensor, Arduino Uno, and ESP8266 microcontroller - this chapter aims to provide a comprehensive understanding of their roles, interactions, and contributions to our cup handling and material dispensing system. By grasping the functionalities of each component, readers can gain insights into the electronic part and appreciate its significance in the overall system's operation.

# Chapitre03 : Coding part

## 1) introduction

We used many techniques during the build of this system, starting from basic components, to the communication and monitoring state of our system.

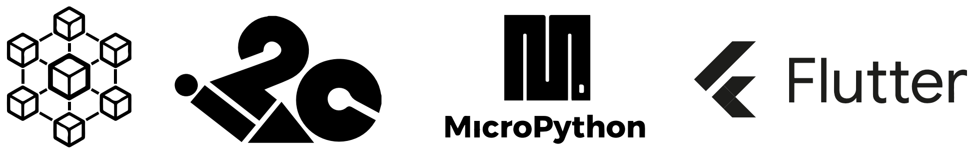


*Figure 9: techs we used*

## 2) techs used

### 2.1 MicroPython

**MicroPython** is a compact version of Python designed for microcontrollers and embedded systems, making it ideal.[1]

### 2.2 I2C (Inter-Integrated Circuit)

I2C (Inter-Integrated Circuit) is a communication protocol commonly used in embedded systems and electronic devices. It enables communication between microcontrollers and various peripherals such as sensors, displays, and memory modules. I2C uses a two-wire serial interface, consisting of a clock line (SCL) and a data line (SDA), allowing multiple devices to be connected on the same bus. It provides a simple and efficient method for data exchange, making it popular for interconnecting components in a wide range of applications.[2]

### 2.3 Flutter (Mobile apps freamwork)

Flutter is a Google-developed open-source framework for building high-performance apps on multiple platforms from a single codebase. It offers fast development, visually appealing interfaces, and cross-platform compatibility.[3]

---

[1] https://micropython.org/
[2] https://www.i2c-bus.org
[3] https://flutter.dev/

we use it to as commend interface.

## 2.4 MicroServices

are small, independent services that make up an application. They are developed, deployed, and scaled independently, communicating with each other through APIs. This architectural approach offers flexibility and modularity in building and maintaining applications.[1]

## 2.5 Design patterns

Design patterns are reusable solutions to common software design problems. They help improve code organization, maintainability, and scalability in software systems. By applying design patterns, developers can solve problems efficiently and create more robust and flexible software.[2]

## 2.6 No delay

we use milis() to excute daily functions.



[1] https://microservices.io/
[2] https://refactoring.guru/design-patterns

## 3) LOGIG DAIGRAM

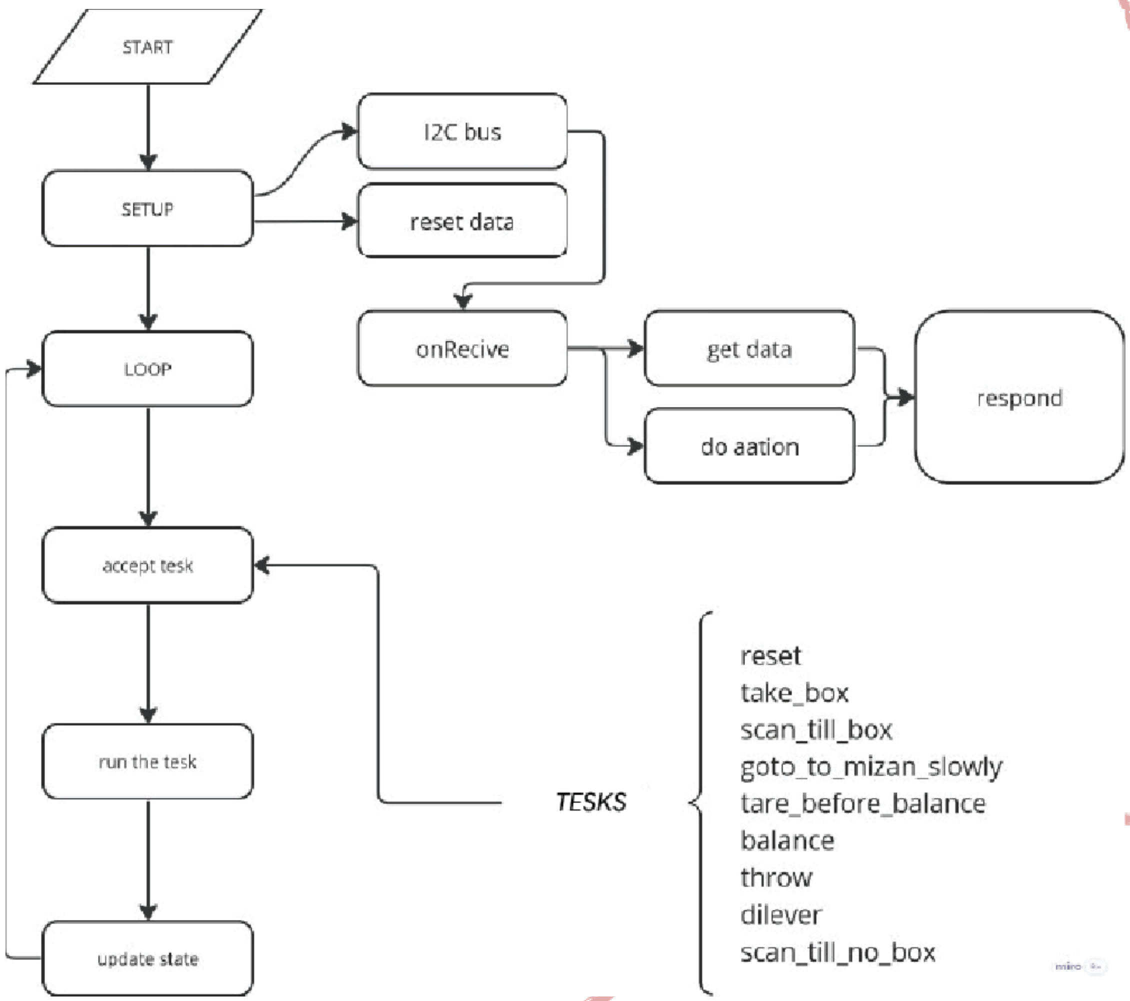


*Figure 10: LOGIG DAIGRAM*

## 4) Conclusion

In conclusion, the coding phase of our project involved the utilization of various techniques and technologies. We employed **MicroPython** for efficient coding on microcontrollers, utilized the **I2C** communication protocol for seamless data exchange, utilized Flutter for developing a command interface, adopted the microservices architecture for flexibility, implemented design patterns for improved code organization, and employed the "**no delay**" approach using **millis()** for executing daily functions. These techniques and technologies played a crucial role in achieving our project goals and ensuring the effectiveness and efficiency of our system.

# conclusion

by the end we get

In conclusion, several limitations and challenges have been identified in the given context. Firstly, I2C communication limitations have hindered the effective utilization of interrupt functions. This constraint may require alternative communication protocols or workarounds to overcome the issue.

Secondly, power supply instability due to voltage fluctuations caused by large servo motors has been observed. This instability may impact the overall performance and reliability of the system. Implementing measures such as voltage regulators or power conditioning techniques could help mitigate this problem.

Another challenge highlighted is the lengthy code compilation times, which can be a significant annoyance during the development process. Exploring options to optimize the code compilation process, such as using more efficient compilers or development tools, could enhance the development workflow.

The presence of arm vibrations has been identified as a mechanical and stability issue during operation. These vibrations can affect the precision and accuracy of the system. Conducting a thorough analysis of the mechanical design, exploring damping techniques, or implementing control algorithms to compensate for vibrations could improve the system's stability.

The use of low-quality components in the breadboard and component setup has caused various problems. Upgrading to higher-quality components or using more reliable circuitry solutions can help mitigate these issues and improve the overall performance and reliability of the system.

Furthermore, algorithm refinement is required, specifically in optimizing the opening gate and material dispensing speed. This optimization process can improve the efficiency and effectiveness of the system, leading to better performance and resource utilization.

In light of the identified challenges, future improvements can be made to address these limitations. This may include exploring alternative communication protocols, implementing measures to stabilize the power supply, optimizing the code compilation process, improving the mechanical design to reduce vibrations, upgrading components, and refining algorithms for better efficiency.

Overall, by addressing these challenges and making the necessary improvements, the system's performance, stability, and reliability can be significantly enhanced, enabling a more efficient and effective operation in the future.

# Footnotes

https://arduino-esp8266.readthedocs.io/en/2.5.2/reference.html

https://micropython.org/

https://www.i2c-bus.org

https://flutter.dev/

https://microservices.io/

https://refactoring.guru/design-patterns