**TECHNICAL UNIVERSITY OF MOLDOVA**

**FACULTY OF COMPUTERS, INFORMATICS AND MICROELECTRONICS**

**DEPARTMENT OF SOFTWARE ENGINEERING AND AUTOMATION**

**Laboratory Work No. 1.2**

**User interaction – LCD and Keypad**

**Realized by: Iațco Sorin**

**st. gr. FAF-213**

**Checked by: Moraru Dumitru**

**univ. lect.**

**Chișinău, 2024**

# 1 THE TASKS OF LABORATORY WORK

Configure an application to work with the STDIO library through the serial interface for text exchange via LCD & Keypad. To design a MCU-based application to detect a code from a 4x4 keypad, verify the code and display a message on an LCD.

- for a valid code, a green LED should light up, for an invalid code, a red LED.

- use STDIO to scan the keypad and display on the LCD.

# 2 IMPLEMENTATION

**2.1 Main program**

Constants are defined for the I2C address of the LCD, LCD dimensions, maximum length of the code, pins for LEDs, keypad layout, and keypad pins.

Global Variables, a constant array keys[][] defines the layout of the keypad. Arrays rowPins[] and colPins[] specify the pins connected to the rows and columns of the keypad matrix. An instance of the Keypad class is created using the makeKeymap() function to map keys to their corresponding positions on the keypad. An instance of the LiquidCrystal\_I2C class is created for controlling the LCD. Arrays and variables are declared for storing the entered code, tracking the code index, and indicating whether the code should be checked.

In the setup() function, the LCD is initialized, its backlight is turned on, and the initial message is displayed on it. Additionally, the pins for the LEDs are set as outputs, and serial communication is initialized for debugging purposes.

The loop() function continuously runs and checks for key presses from the keypad.

Keypad Input Handling, when a key is pressed, it is checked if it's not equal to NO\_KEY (a constant representing no key being pressed). If the key pressed is '#', it signifies the end of the code input. The system then checks whether the entered code matches the desired code. If the key pressed is not '#', it's added to the enteredCode array if the maximum code length has not been reached, and it's displayed on the LCD [1].

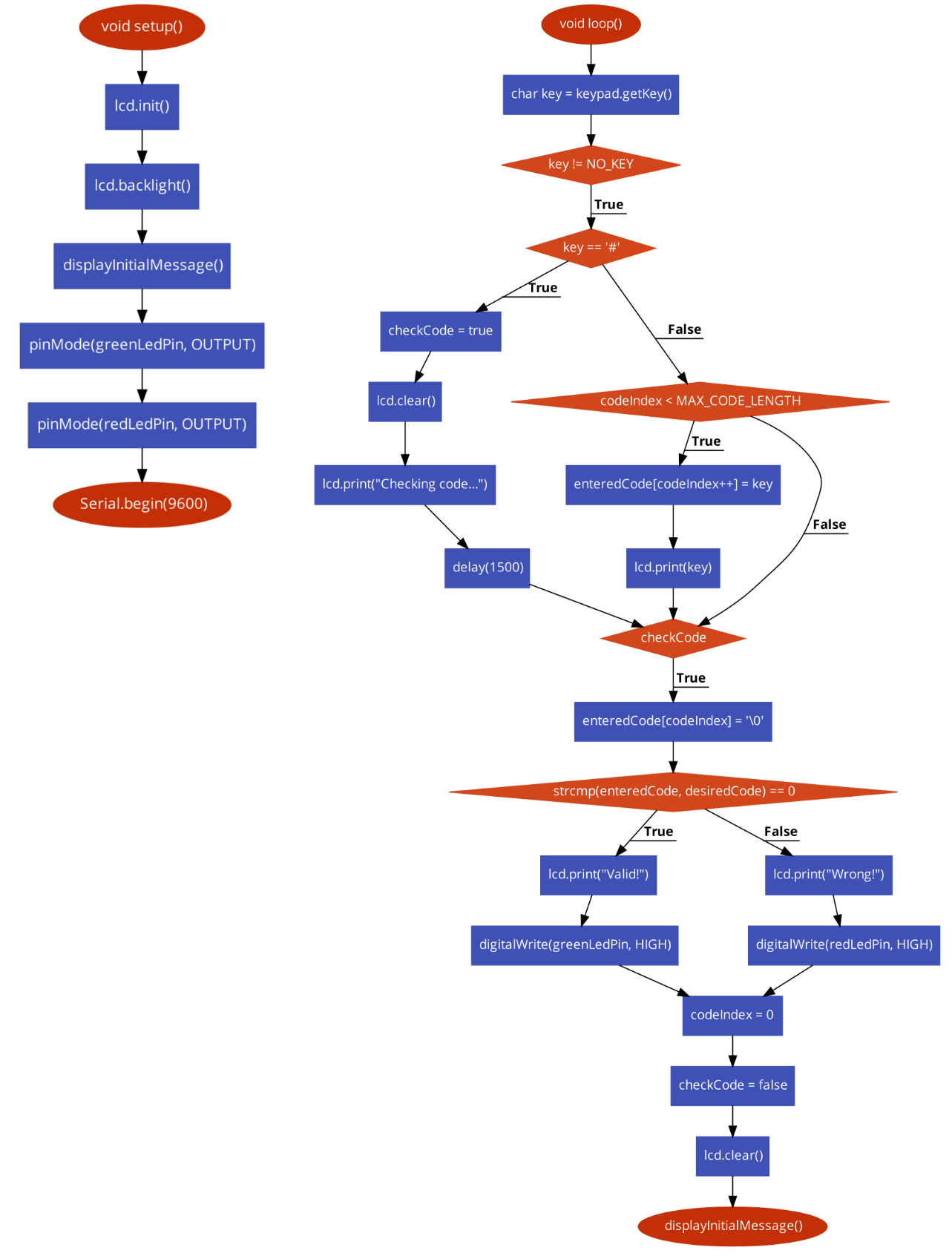
Code Verification, if checkCode is true (indicating the user has pressed '#'), the entered code is compared with the desired code using strcmp. If the entered code matches the desired code, "Valid!" is displayed on the LCD, and the green LED is turned on for 3 seconds. If the entered code doesn't match the desired code, "Wrong!" is displayed on the LCD, and the red LED is turned on for 3 seconds.

Resetting State, after checking the code, codeIndex (tracking the index of the entered code) is reset to 0, checkCode is set to false, and the LCD is cleared to display the initial message again.

Display Initial Message: The displayInitialMessage() function sets the cursor position on the LCD and prints the initial instructions for entering the code [2].

**2.2 Block diagram**

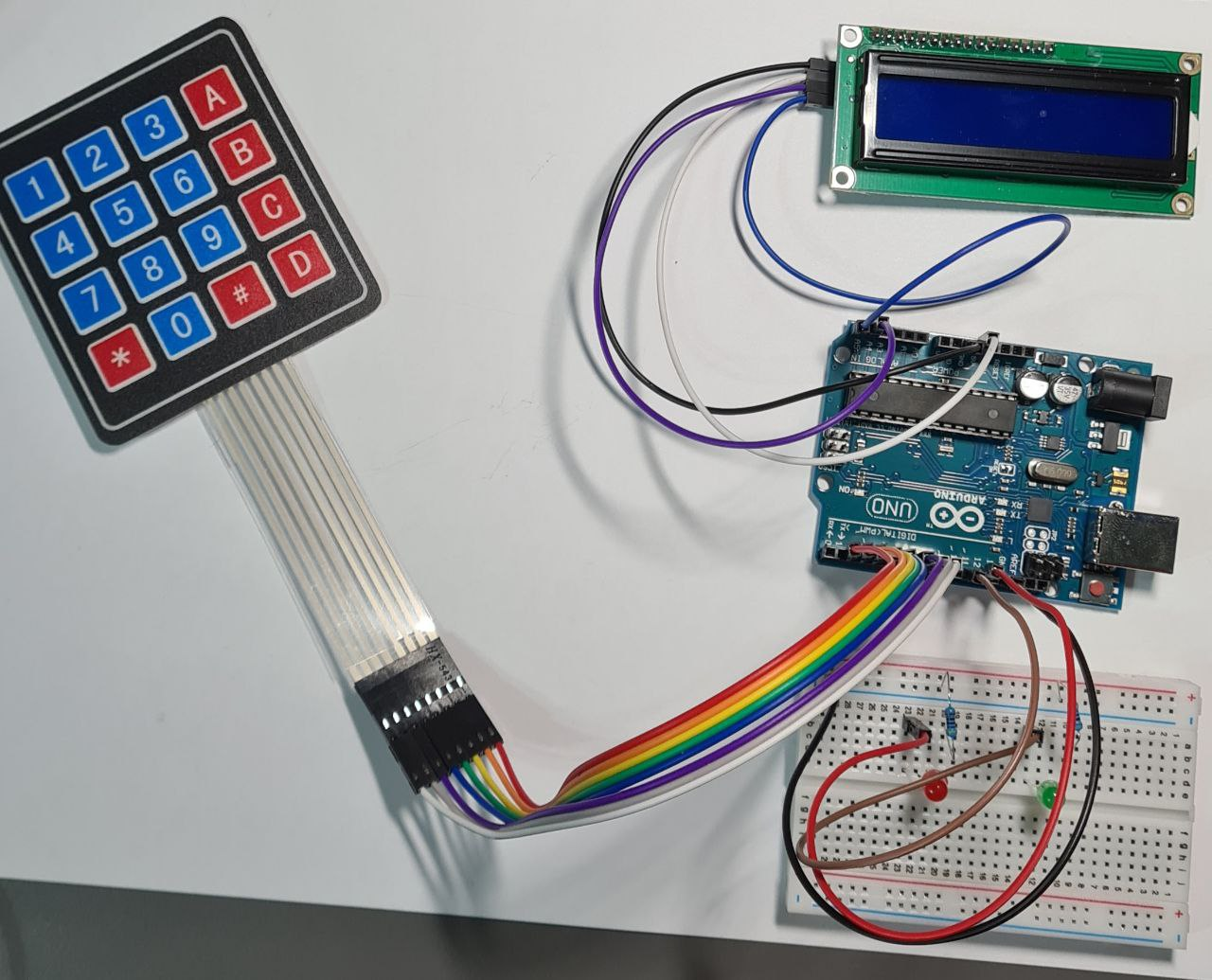
Block diagram illustrating the main components and interactions in the provided Arduino code for the keypad-based code verification system using online tool [3].



**Figure 1. Block diagram**

**2.3 Circuit electrical schematic**

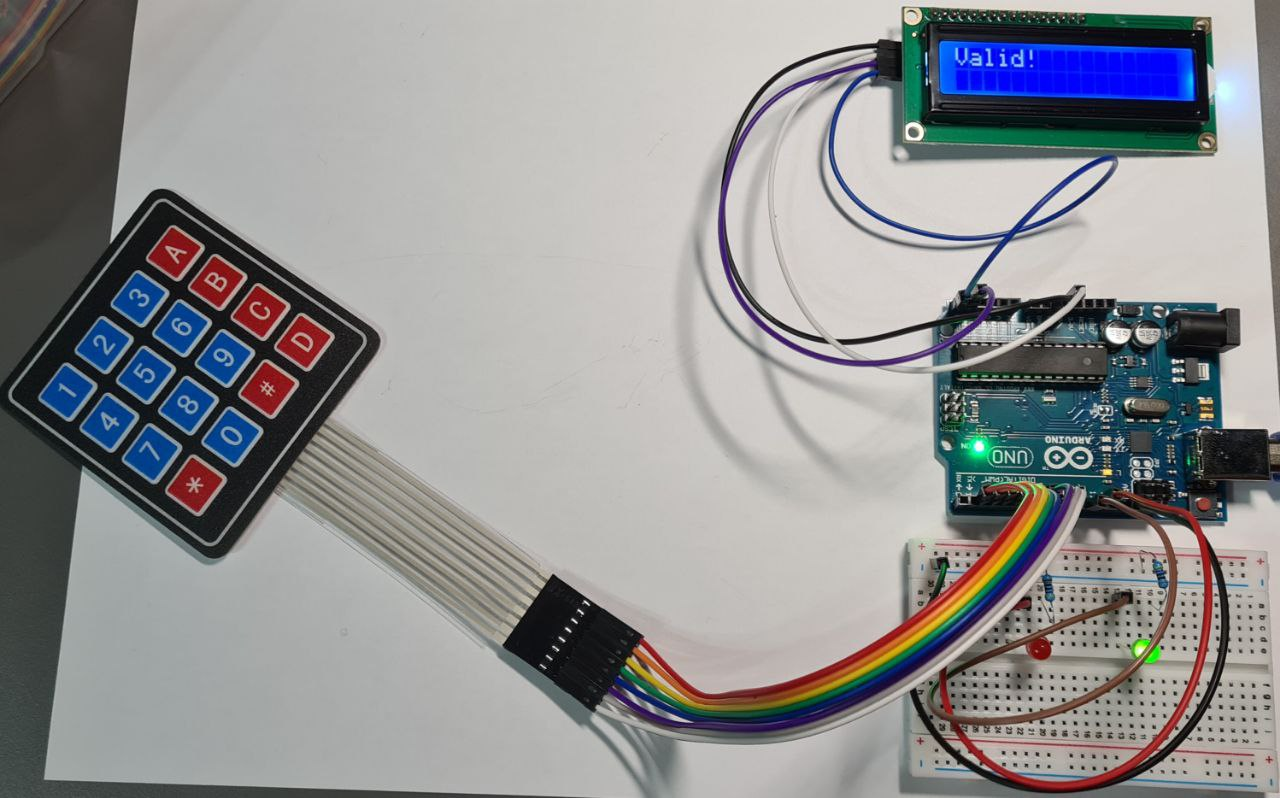
Schematic representation of the circuit illustrating how the various components of the keypad-based code validating system are connected to the Arduino board and to each other, enabling the system to function as intended.



**Figure 2. Electrical schematic**

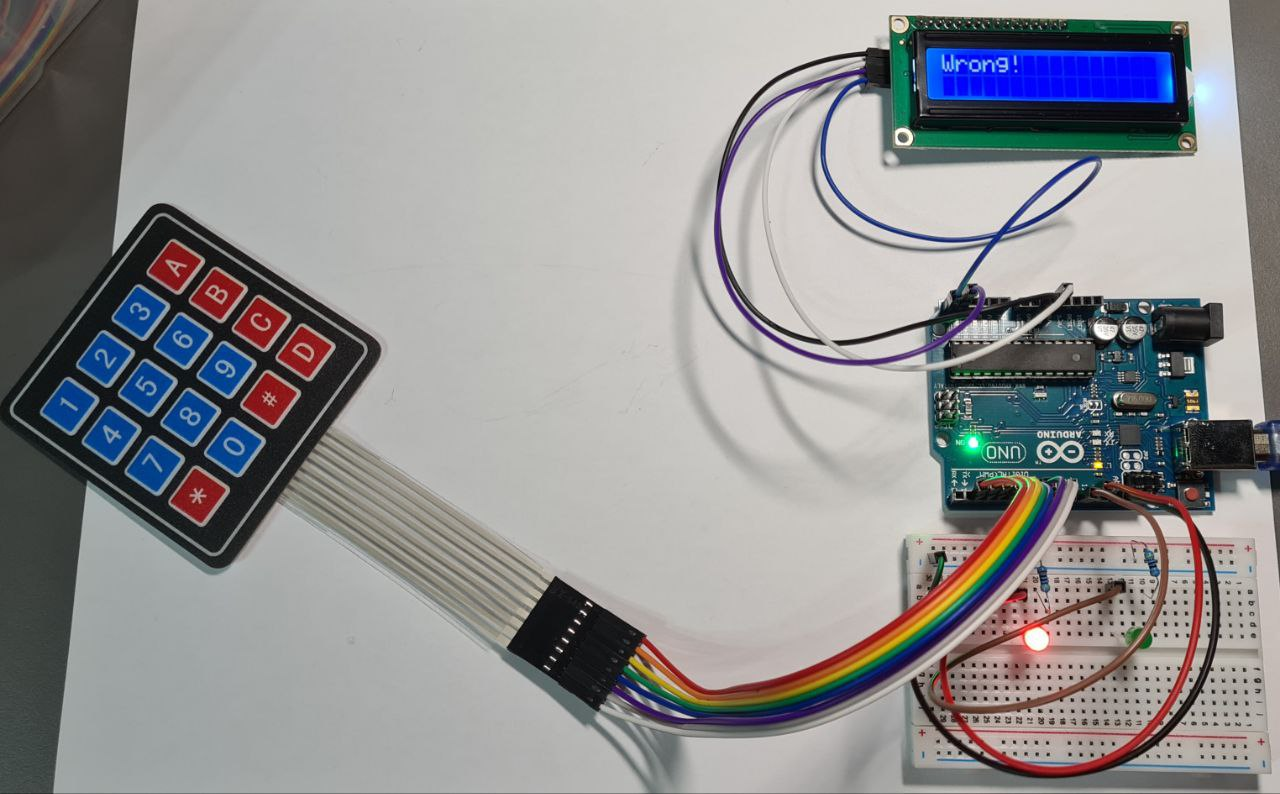
**2.4 Photos of running simulation**

Program simulation involving keypad input “1234” which is the expected code turning the green LED on and displaying text “Valid!” on the LCD display.



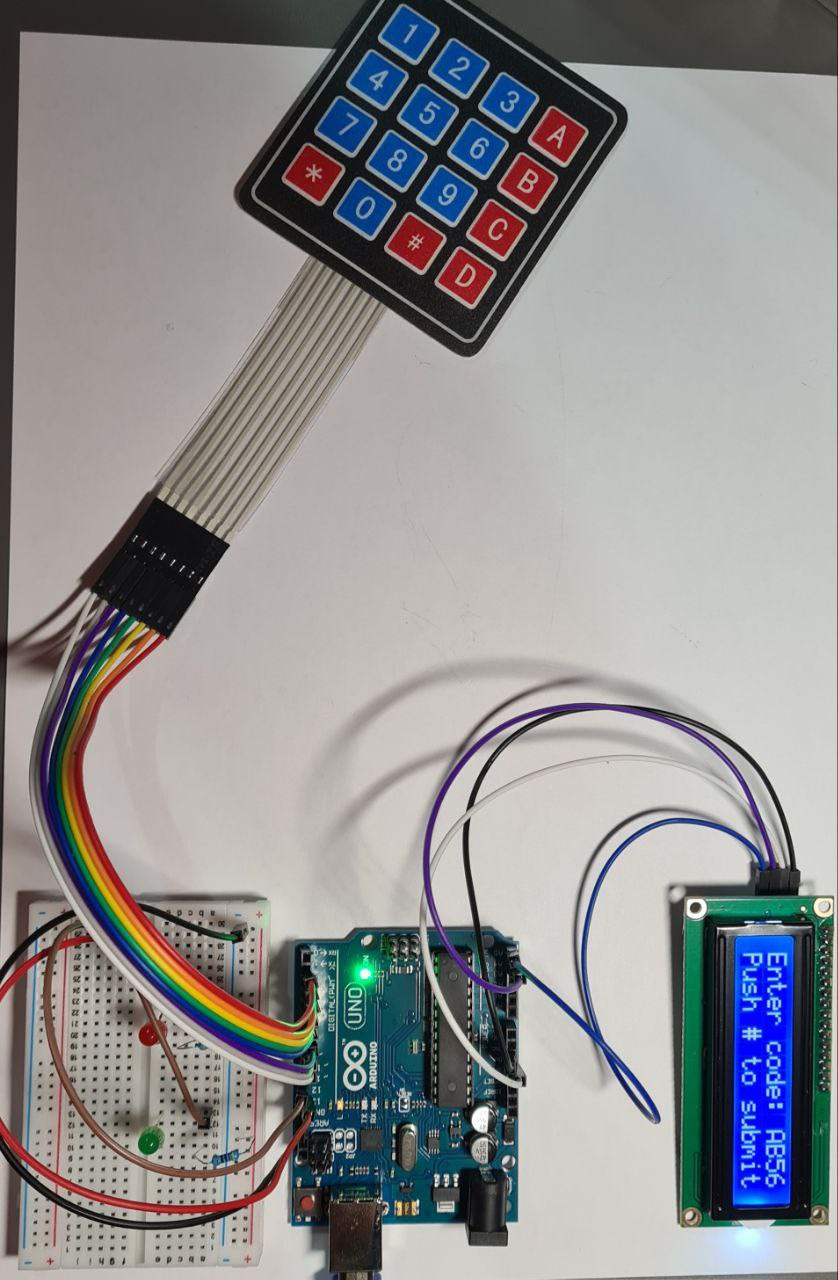
**Figure 3. Program simulation (keypad input “1234”)**

Program simulation involving keypad input “A21B” which is incorrect turning the red LED on and displaying “Wrong!” text on the LCD display.



**Figure 4. Program simulation (keypad input “A21B”)**

The default state of the circuit where the LCD displays “Enter code:” on the first row and “Push # to submit” on the second row, expecting the user to press on the keypad.



**Figure 5. Program simulation (keypad input “AB56”)**

# CONCLUSION

In summary, this lab work has improved my knowledge of embedded systems development, especially with regard to projects involving Arduino. Additionally, it has given me hands-on experience troubleshooting and debugging problems that could come up during the development process.

Future research into embedded systems and Internet of Things applications can be facilitated by utilizing the information and abilities obtained from this project, which can also be used to other projects involving Arduino-based systems.

# BIBLIOGRAPHY

1. ARDUINOGETSTARTED: Keypad documentation for arduino. [online], [accesed 11.02.2024]. Available at: <https://arduinogetstarted.com/tutorials/arduino-keypad>
2. PROJECTHUB.ARDUINO.CC: LCD documentation for arduino 16x2. [online], [accesed at 12.02.2024]. Available at: <https://projecthub.arduino.cc/arduino_uno_guy/i2c-liquid-crystal-displays-5eb615>
3. CODE2FLOW: Diagram tool. [online], [accesed at 10.02.2024]. Available at: <https://app.code2flow.com>