PRACTICE 4: MODERN IOT

1. PIR motion sensor

In the components list there exists a PIR motion sensor component that simulates motion. The PIR sensor has three pins; a power, ground, and digital output (OUT). Triggering the sensor will drive the OUT pin high for 5 seconds, and then go low again. The sensor will ignore any further input for the next 1.2 seconds, and then start sensing for motion again. Create a project that upon detecting movement turns on a red LED.

2. Three LED

Develop a program to generate a custom LED pattern on three GPIO pins. The pattern should repeat indefinitely, cycling through turning on and off each LED in sequence (e.g., LED1 on, LED2 on, LED3 on, LED1 off, LED2 off, LED3 off, repeat).

3. Keypad

In the components, under inputs, you can find a keypad component. This is a matrix keypad, which arranges the buttons in rows and columns. Here's how it works and button presses detected:

- **Matrix Layout**: The buttons on the keypad are arranged in a grid pattern, typically with rows and columns. Each button is at the intersection of a row and a column, forming a matrix
- **Scanning Rows and Columns**: To detect button presses, the microcontroller scans each row and column of the keypad in sequence. It sets one-row line as output (high) and the rest as input. Then, it checks the status of each column line. If a column line reads low, it indicates that a button in that row is pressed.
- **Multiplexing**: The scanning process is repeated for each row, cycling through all rows one by one.
- **Button Detection and Character Mapping**: When a button on the keypad is pressed, it creates a connection between the corresponding row and column lines. Once a button press is detected, the microcontroller maps

the row and column of the pressed button to a specific character or function based on the keypad layout. For example, if row 1 and column 1 are connected, the microcontroller knows that the ”1” button is pressed.

Based on the above, create a simple application that detects the button pressed and prints it to the console.

4.

Refactor the temperature sensing application to include a flashing/blinking LED. As the temperature increases, the LED should flash faster. On the other hand, when the tempera-ture decreases the LED blinks slower.

5.

Wokwi has an analog joystick component that provides a range of analog values depending on the direction. Create an application that prints out the position of the joystick.

6.

Using the LED bar and the rotating potentiometer, create an application that replicates a volume control dial. Meaning as the potentiometer dial is rotated clockwise, an increasing amount of LEDs light up indicating higher volume. Conversely, as the dial is rotated counter-clockwise, LEDs are turned off.