CODEALPHA INTERNSHIP (10th Dec 2025 – 10th Jan 2026)

INTERNET OF THINGS DOMAIN

STUDENT ID: CA/DE1/5845

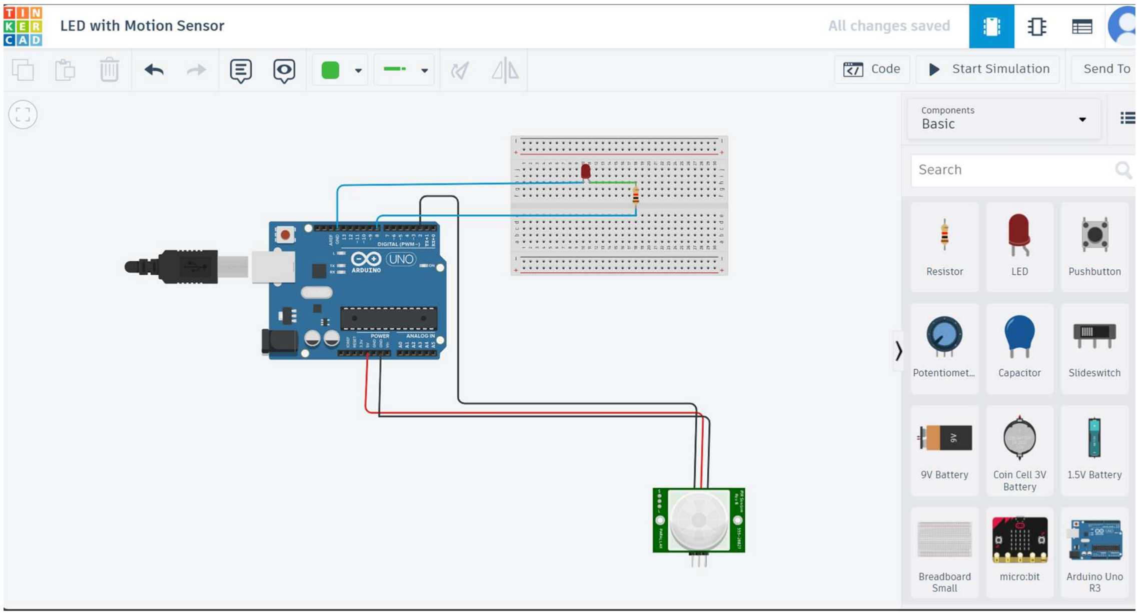
TASK 2: Sensor-Based Simulation

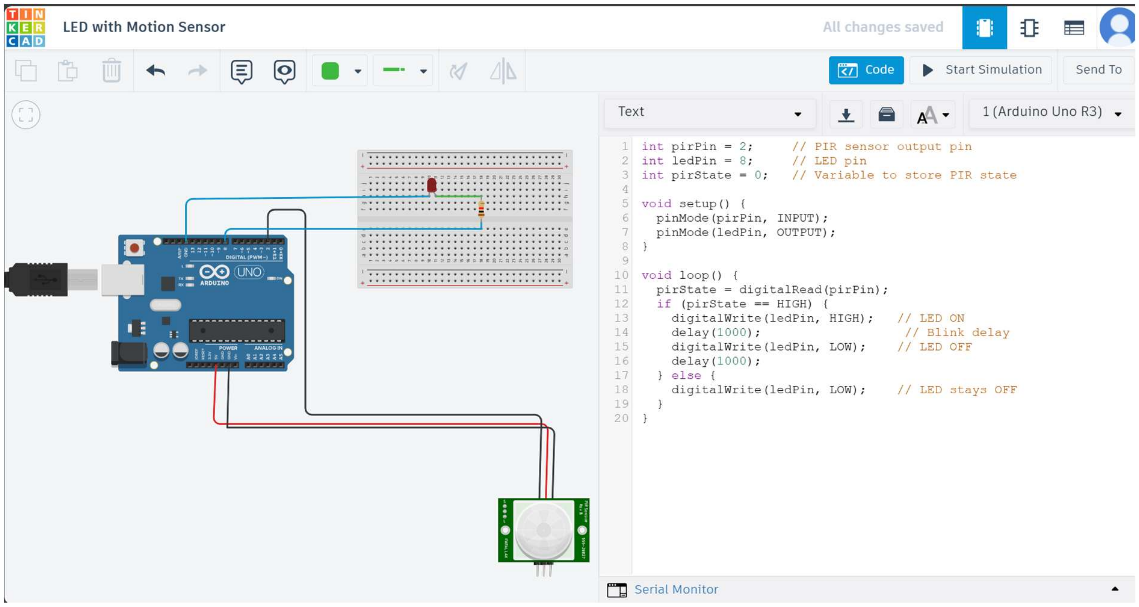
- **Simulate an IoT system using Tinkercad/Proteus (no hardware required).**
- **Example: LED on/off using a sensor (Temperature/Light/Motion).**
- **Submit simulation screenshots + code explanation.**

OBJECTIVE:

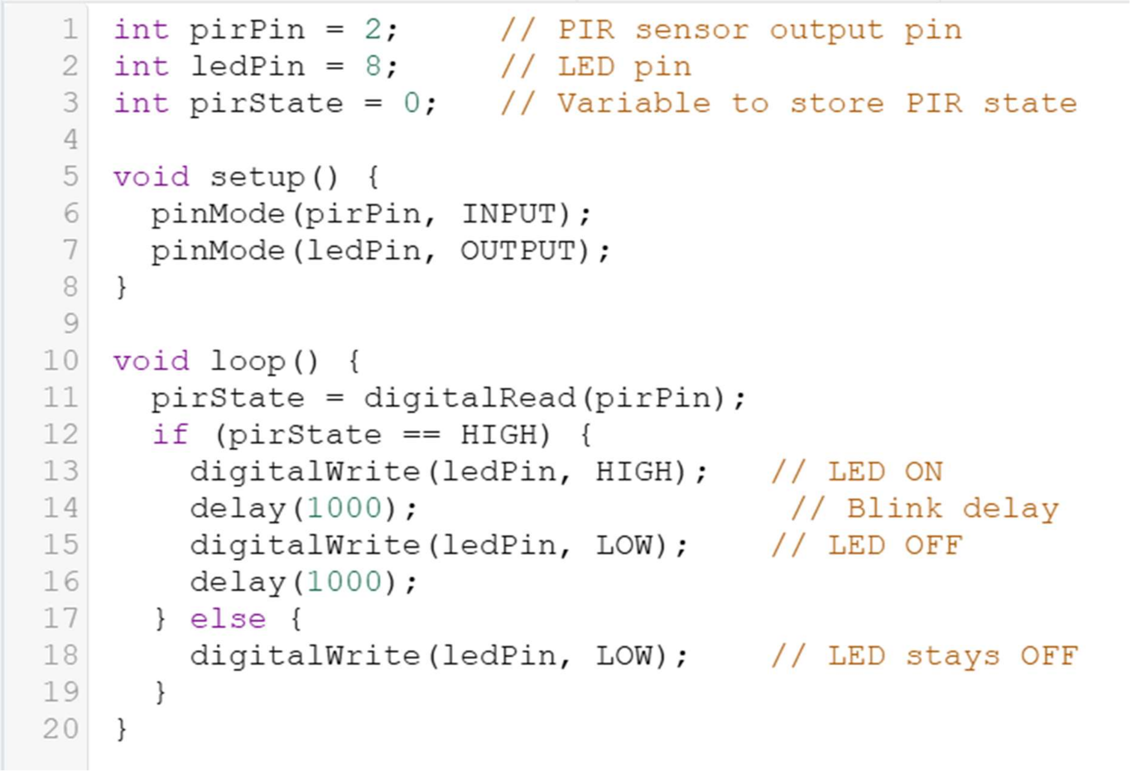
To design a simple **motion-activated LED system** where the LED serves as an indicator whenever movement is detected by the PIR sensor. This can be applied in **security systems, automatic lighting, and energy-saving devices**.

SCREENSHOTS:

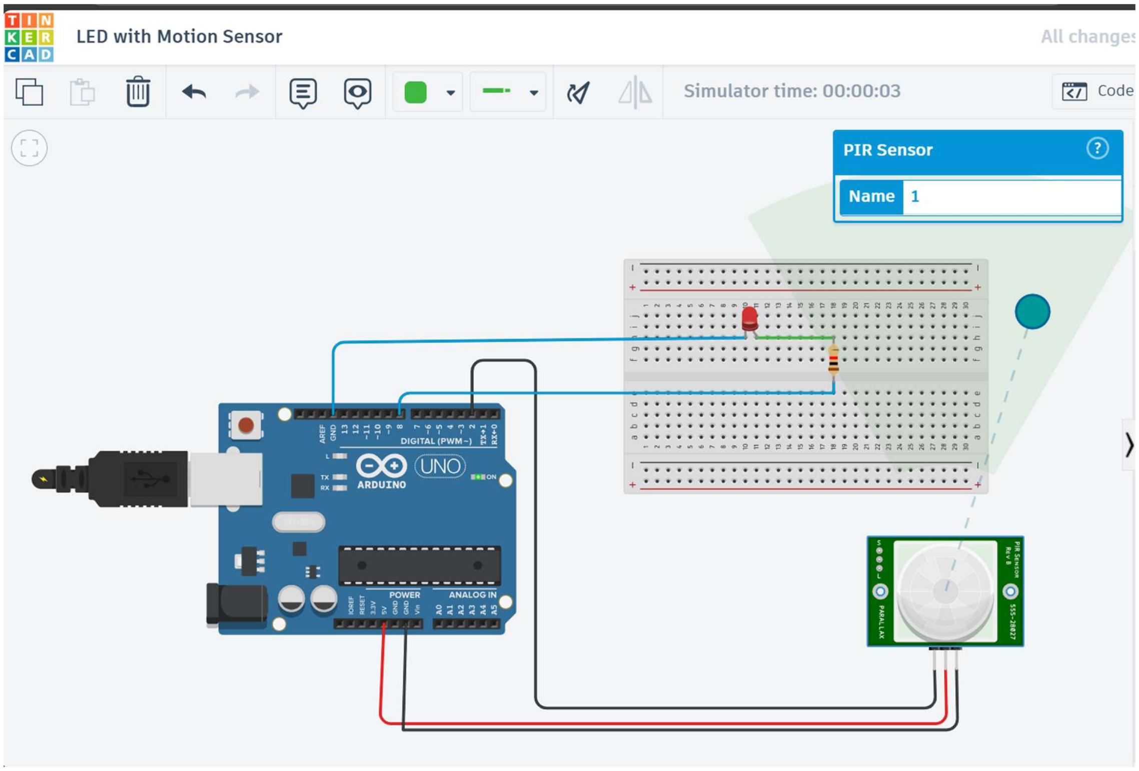




```
int pirPin = 2;     // PIR sensor output pin
int ledPin = 8;     // LED pin
int pirState = 0;   // Variable to store PIR state

void setup() {
  pinMode(pirPin, INPUT);
  pinMode(ledPin, OUTPUT);
}

void loop() {
  pirState = digitalRead(pirPin);
  if (pirState == HIGH) {
    digitalWrite(ledPin, HIGH);   // LED ON
    delay(1000);                   // Blink delay
    digitalWrite(ledPin, LOW);    // LED OFF
    delay(1000);
  } else {
    digitalWrite(ledPin, LOW);    // LED stays OFF
  }
}
```



```
int pirPin = 2;     // PIR sensor output pin
int ledPin = 8;     // LED pin
int pirState = 0;   // Variable to store PIR state

void setup() {
  pinMode(pirPin, INPUT);
  pinMode(ledPin, OUTPUT);
}

void loop() {
  pirState = digitalRead(pirPin);
  if (pirState == HIGH) {
    digitalWrite(ledPin, HIGH);   // LED ON
    delay(1000);                   // Blink delay
    digitalWrite(ledPin, LOW);    // LED OFF
    delay(1000);
  } else {
    digitalWrite(ledPin, LOW);    // LED stays OFF
  }
}
```



Name 1

CODE:

```
int pirPin = 2;     // PIR sensor output pin
int ledPin = 8;     // LED pin
int pirState = 0;   // Variable to store PIR state

void setup() {
  pinMode(pirPin, INPUT);
  pinMode(ledPin, OUTPUT);
}

void loop() {
  pirState = digitalRead(pirPin);
  if (pirState == HIGH) {
    digitalWrite(ledPin, HIGH);   // LED ON
    delay(10000);                 // Blink delay
    digitalWrite(ledPin, LOW);    // LED OFF
    delay(10000);
  } else {
    digitalWrite(ledPin, LOW);    // LED stays OFF
  }
}
```

CODE EXPLANATION:

This project demonstrates the use of a **PIR (Passive Infrared) motion sensor** to control an **LED** using Arduino programming. The PIR sensor detects human movement, and the LED responds accordingly.

The **PIR sensor** acts like a *watchman* that continuously checks for movement.

- When motion is detected, the sensor sends a signal (HIGH) to the Arduino.
- The Arduino then turns the **LED ON for 10 seconds**, followed by turning it **OFF for 10 seconds**, creating a blinking effect.
- If no motion is detected, the LED stays OFF.

The program begins by declaring three variables: `pirPin = 2` assigns the PIR sensor's output to digital pin 2, `ledPin = 8` assigns the LED to digital pin 8, and `pirState = 0` initializes a variable to store whether motion is detected or not. In the `setup()` function, the PIR pin is configured as an **input** so the Arduino can read signals from the sensor, while the LED pin is set as an **output** so the Arduino can control the LED.

Inside the continuously running `loop()`, the Arduino reads the PIR sensor's state using `digitalRead(pirPin)` and stores the result in `pirState`. If the sensor reports `HIGH`, meaning motion is detected, the LED is turned ON (`digitalWrite(ledPin, HIGH)`), kept ON for 10 seconds (`delay(10000)`), then turned OFF (`digitalWrite(ledPin, LOW)`) and kept OFF for another 10 seconds. This creates a blinking cycle whenever motion is sensed. If the sensor reports `LOW`, meaning no motion is detected, the LED remains OFF (`digitalWrite(ledPin, LOW)`).

Thus, the code implements a simple motion-activated LED system: the PIR sensor acts like a watchman detecting movement, and the LED serves as a visual indicator, switching ON and OFF in timed intervals when motion is present, while staying OFF when no motion is detected.