Homework no. 1

Smart School

Ciorâțanu Maria, Pâncă Aida-Gabriela, Varzar Alina-Miruna
year III, group A5

Content:

1. Applicability domains
    a. use-cases (minim 4, maxim 8);
    b. one of the use-cases is chosen and the system is defined in details;
2. System definition:
    a. components that enter the system infrastructure;
    b. protocols used in communication processes;
    c. process automation;
    d. estimated costs;
3. Cisco Packet Tracer:
    a. add a print screen of the designed architecture;
    b. explain the functionalities implemented in Cisco Packet Tracer;
4. TinkerCad:
    a. add a print screen of the designed architecture;
    b. explain the functionalities implemented in TinkerCad.

1. Applicability domains

In a smart school, IoT technology can solve a few major problems:

- Fire safety detecting flames and automatically extinguishing them before the fire spreads.
- Energy efficiency autonomously adjusting temperature and lighting so that energy is used only when needed.
- Access control opening doors and turning on lights only when students or staff are present, maintaining security when the space is empty.
- Visual feedback during lessons LEDs programmed to change color or intensity to mark different moments of the class (break, experiment, group work time).

a. Use-cases

- Fire alarm with automatic sprinklers, motorised window, siren, and an LCD warning message.
- Smart temperature control that automatically switches between heating and cooling and shows the current status on a display.
- Intelligent door that opens only when the motion sensor detects movement; simultaneously, the lamp and video camera turn on, and the screen greets the visitor.
- Educational LED system that runs colour and message sequences to provide students with clear visual cues.
- Smart Attendance & Seat‑Map – RFID or BLE tags in student ID cards are read when pupils enter the classroom; a live seating map shows which desks are occupied so the teacher can start the lesson without roll‑call.
- Connected Science‑Lab Benches – Gas taps and 220 V sockets on each lab table are interlocked with presence sensors; if no student is detected for 2 minutes, supplies shut off automatically, preventing accidents and saving energy.
- Intelligent Lockers & Reminder Alerts – Hallway lockers unlock via NFC on a phone; if a locker remains open or a PE kit is left inside over the weekend, the student receives a push‑notification to collect belongings.
- School‑Bus Geo‑Fence & Arrival Display – GPS‑equipped buses publish their position to a campus dashboard; entrance screens count down minutes to arrival and

send an SMS to parents if a bus is delayed by more than ten minutes.

b. Fire Alarm

This program is designed to enhance the safety of students and staff in a smart school. In the event of a fire, the sensor quickly detects the flame, and the controller automatically activates protective measures: the sprinkler system starts to extinguish the fire, the window opens to let out the smoke, and the Piezo Speaker emits an audible alarm for warning. At the same time, the LCD screen displays the message "FIRE DETECTED," informing students and teachers about the danger.

Once the fire is extinguished, the smoke is cleared, and the area becomes safe, the system automatically returns to its initial state. The alarm stops, the window closes, and the LCD shows the message "SAFE MODE." In this way, students and teachers can clearly see that the danger has passed and can safely return to the building.

When the flame is near the Fire Motion sensor, the controller performs the following actions:

- activates the sprinkler, and water starts flowing;
- opens the window for smoke evacuation;
- powers the Piezo Speaker, which emits an intermittent sound alarm;
- displays the message "FIRE DETECTED" on the LCD.

As soon as the flame disappears:

- the sprinkler is turned off;
- the window closes;
- the Piezo Speaker stops sounding;
- the LCD returns to the "SAFE MODE" text, confirming the system is back to normal.

The MCU board is connected to five devices using wires that link the digital pins D0–D4 of the board to the D0 pin of each device.
A program runs on the MCU board, which performs specific actions for each pin D0–D4. Thus, when the Fire Monitor detects a fire, it activates the Fire Sprinkler, triggers the Piezo Speaker sound, opens the window, and displays the message "FIRE DETECTED."

2. System definition

a. Components that enter the system infrastructure;

- Fire-Safety – one flame sensor (KY-026), one Arduino Uno R3, one SG90 micro-servo to open the window, one solenoid valve with ceiling sprinkler, one active piezo buzzer, and one 16×2 LCD with I²C interface.
- Smart Temperature – one DS18B20 temperature probe, one Raspberry Pi Pico W (also acts as a local MQTT broker), two 5 V relays that switch a ceramic heater element or a portable AC / fan-coil, plus one 16×2 LCD.
- Access Control – one HC-SR505 PIR sensor, one smart-lock or servo latch, one 12 V LED lamp for the doorway, one HD USB/Wi-Fi webcam, one Pico W and one 16×2 LCD.
- Educational LED System – one metre of WS2812B LED strip (≈ 30 pixels), one dimmable E27 smart bulb for general room light, one discrete RGB 5 mm LED for demonstrations, one Pico W and one 16×2 LCD.
- Network backbone – a DLC100 IoT Home Gateway supporting 2.4 GHz Wi-Fi and Zigbee 3.0, a cable modem, a coax splitter, an 8-port Fast-Ethernet switch, an 802.11 n access-point, and a tablet PC that runs the "IoT Monitor" dashboard.

b. Protocols used in communication processes

- Physical / MAC layer – 802.11 n Wi-Fi at 2.4 GHz and Zigbee 3.0 link the IoT nodes to the gateway.
- Personal-area communications – Bluetooth-Classic 2.1 pairs a portable music player with a classroom Bluetooth speaker.
- Network / Transport layer – IPv4 with DHCP and NAT; the gateway hands out addresses in the 192.168.10.0/24 range and keeps all IoT devices in VLAN 20.
- Application layer – MQTT v3.1.1 on port 1883 for telemetry and commands, HTTP REST with WebSocket for the teacher dashboard, and local I²C, 1-Wire or bare GPIO buses on every microcontroller.

  Security – WPA2-PSK encrypts the Wi-Fi traffic; optional TLS 1.2 secures any cloud-MQTT uplink.

c. Process automation

- On‑device real‑time logic runs in C/C++ on each MCU or SBC, reacting in under 20 ms to critical events such as flame detection or door motion.
- Edge orchestration is handled by Node‑RED on the Pico W of the temperature node; it subscribes to every MQTT topic, writes logs into a local InfluxDB instance and forwards data to the cloud when the Internet link is up.
- Key rules include:
  - If a flame is detected → turn the sprinkler on, open the window, sound the buzzer and show "FIRE DETECTED" on the LCD; otherwise restore the "SAFE MODE" state.
  - If the room stays above 24 °C for three minutes → switch the AC on; if it drops below 19 °C → switch the heater on.
  - If the PIR senses motion → unlock the door, start video recording, turn the lamp on and greet the visitor on the LCD.
- Run a classroom LED cycle (green, yellow, blue, red) every ten seconds, with matching messages on the LCD, to signal lesson stages.
- Manual override – the teacher's tablet sends REST calls to the gateway, which republishes them as MQTT commands.
- Fail‑safe behaviour – all local rules continue to work without Internet; devices fall back to a safe state after a power loss.

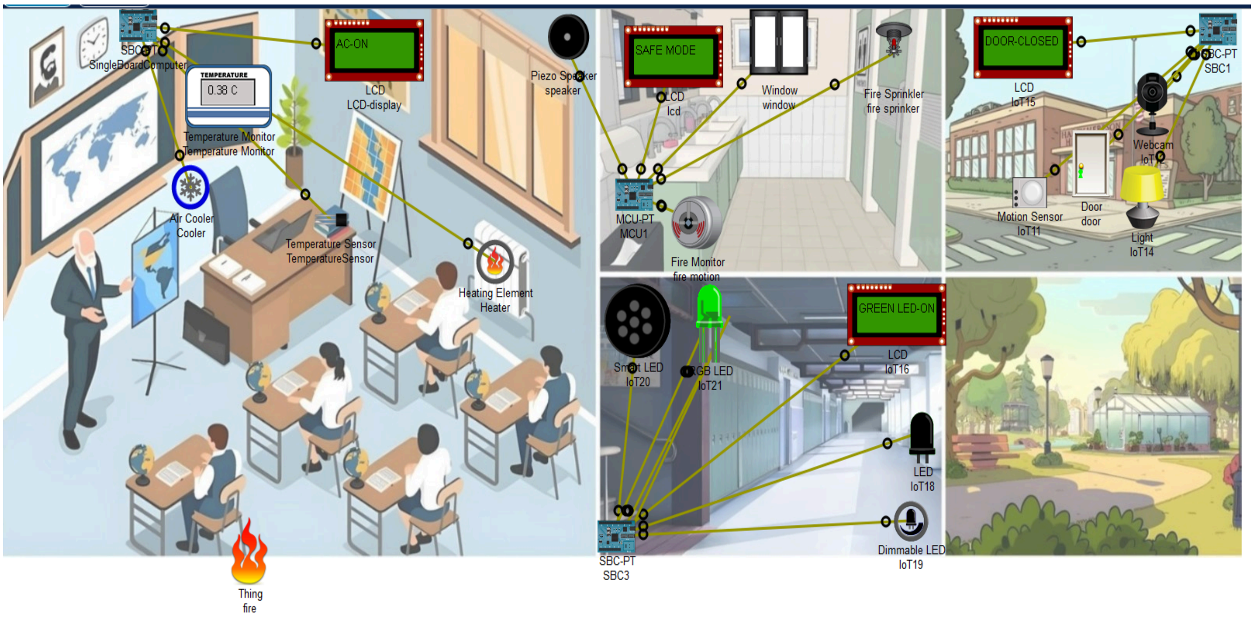
d. Estimated costs

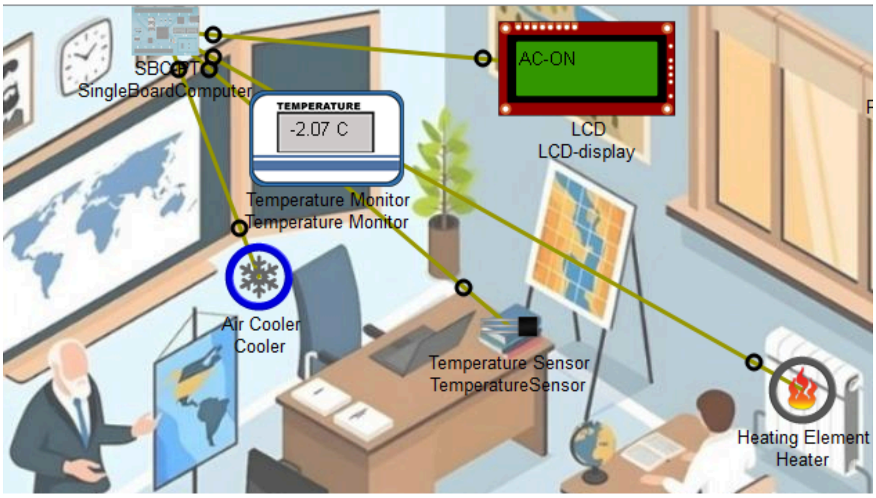
- Fire‑Safety hardware totals roughly 45 € (Arduino Uno R3 ~ 24 €, flame sensor 2.5 €, SG90 servo 4 €, solenoid valve 8 €, piezo 2 €, LCD 5 €).
- Smart Temperature comes to about 92 € (Pico W 6 €, DS18B20 2 €, two relays 4 €, heater element 15 €, portable AC or fan unit 60 €, LCD 5 €).
- Access Control is around 67 € (PIR 2 €, smart‑lock servo 4 €, smart lamp 15 €, HD webcam 40 €, Pico W 6 €).
- Educational LED System is approximately 28 € (WS2812B strip 8 €, smart bulb 8 €, RGB LED 1 €, Pico W 6 €, LCD 5 €).
- Network backbone costs about 215 € in total (DLC100 gateway 50 €, Wi‑Fi AP with switch 35 €, cable modem 30 €, teacher tablet 100 €).

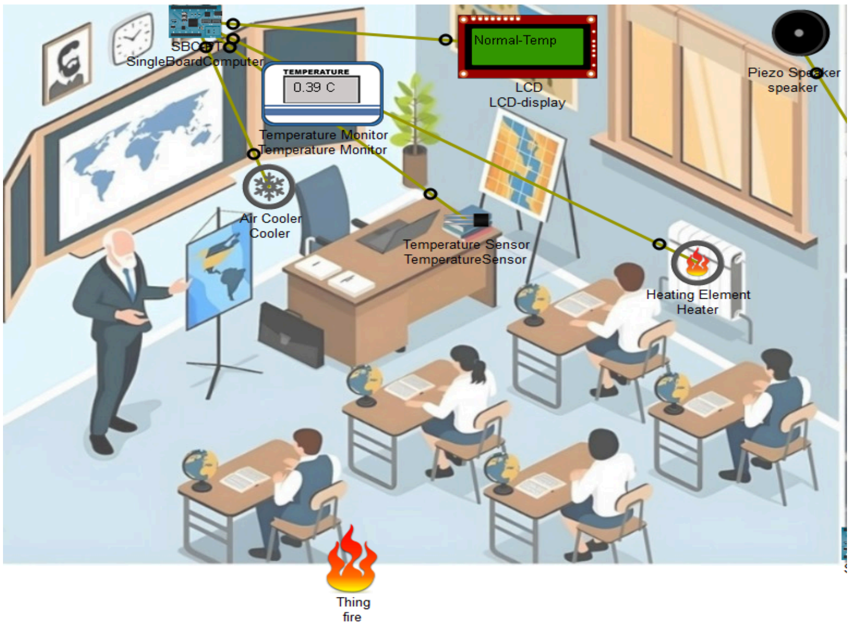
- The complete pilot installation therefore requires ≈ 450 € in hardware. Actual procurement may vary by plus or minus fifteen percent.

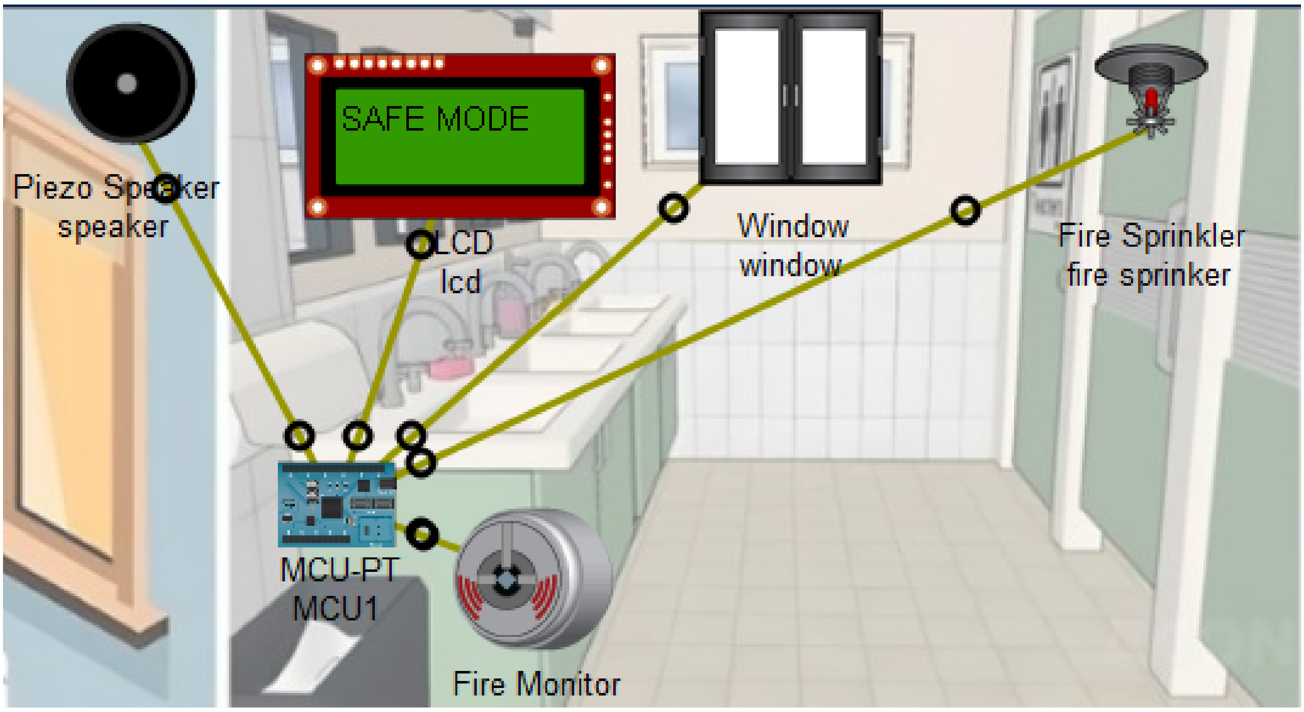
3. Cisco Packet Tracer

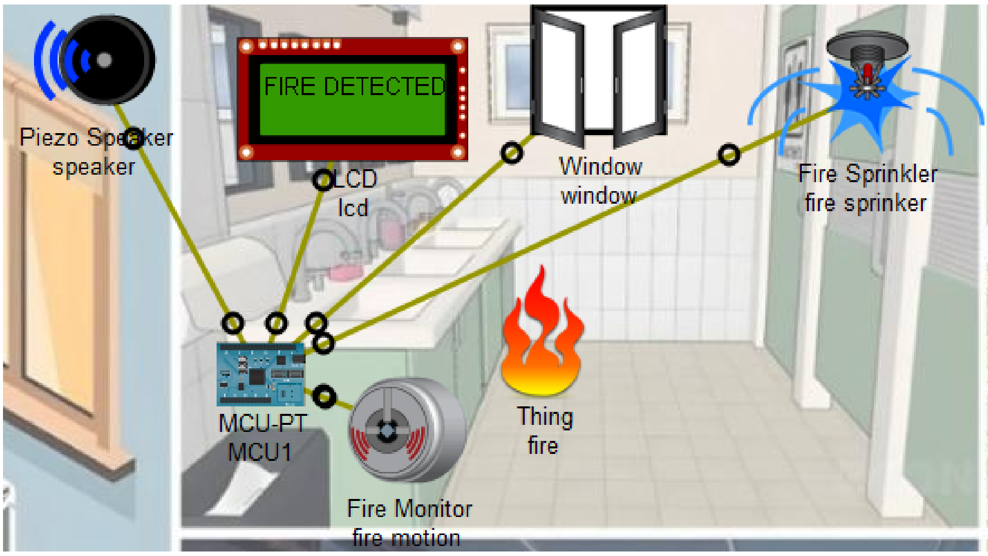
a. Designed architecture

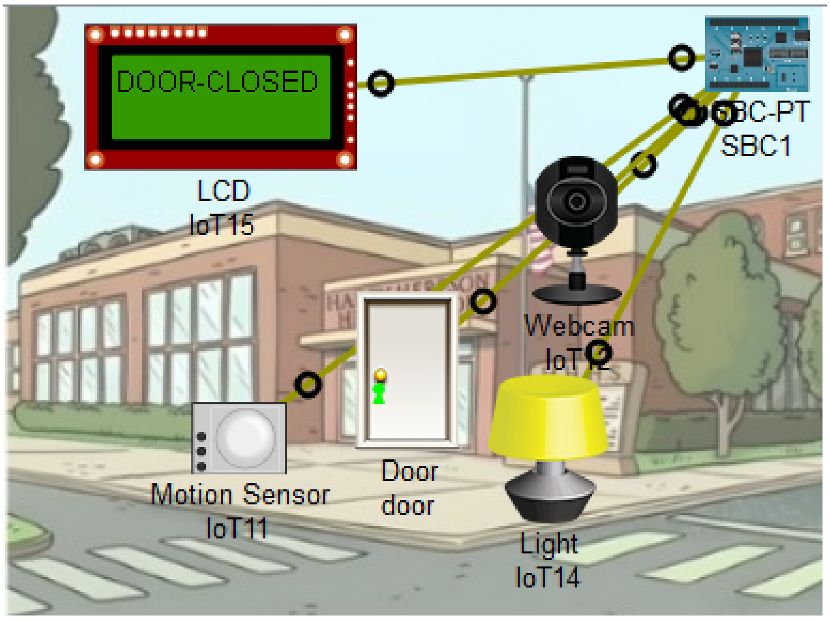


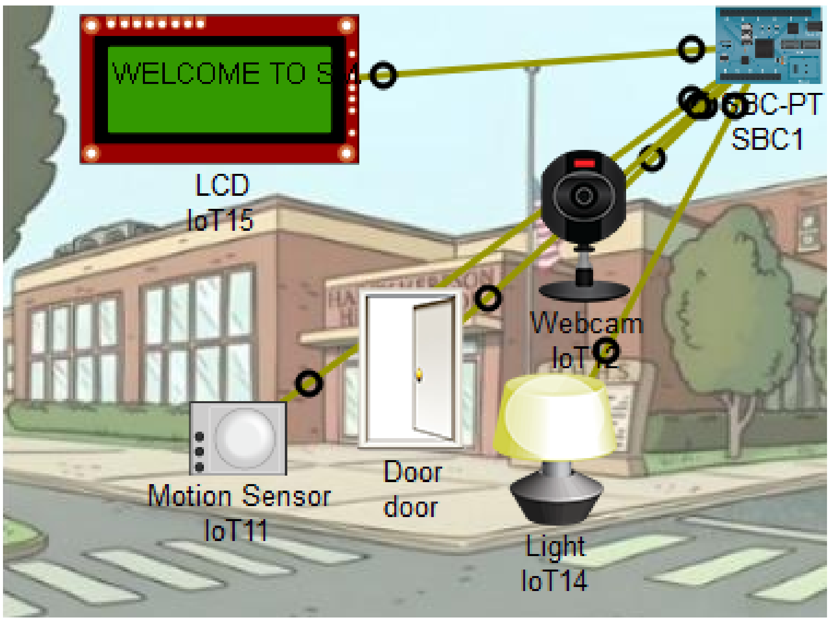


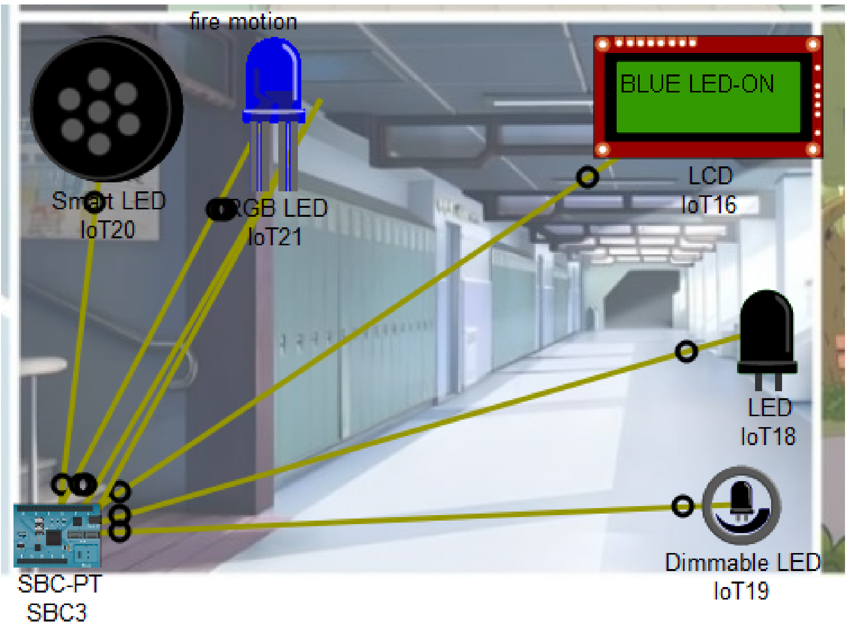

SBC-PT
SingleBoardComputer
Normal-Temp
LCD
LCD-display
Piezo Speaker
speaker
TEMPERATURE
0.39 C
Temperature Monitor
Temperature Monitor
Air Cooler
Cooler
Temperature Sensor
TemperatureSensor
Heating Element
Heater
Thing
fire


Piezo Speaker
speaker
SAFE MODE
LCD
lcd
Window
window
Fire Sprinkler
fire sprinkler
MCU-PT
MCU1
Fire Monitor


FIRE DETECTED
Piezo Speaker
speaker
LCD
lcd
Window
window
Fire Sprinkler
fire sprinkler
MCU-PT
MCU1
Thing
fire
Fire Monitor
fire motion


DOOR-CLOSED
SBC-PT
SBC1
LCD
IoT15
Webcam
Motion Sensor
IoT11
Door
door
Light
IoT14


SBC-PT
SBC1
LCD
IoT15
Webcam
Motion Sensor
IoT11
Door
door
Light
IoT14



b. Functionalities implemented in Cisco Packet Tracer

Fire Alarm

When a flame is near the Fire Motion sensor, the controller performs the following actions:

- activates the sprinkler, and water starts to flow;
- opens the window to evacuate smoke;
- powers the Piezo Speaker, which emits an intermittent sound alarm;
- displays the message "FIRE DETECTED" on the LCD screen.

As soon as the flame disappears:

- the sprinkler is turned off;
- the window closes;
- the Piezo Speaker stops sounding;
- the LCD screen returns to the message "SAFE MODE," confirming the system has returned to normal.

The MCU board is connected to five devices using wires that link digital pins D0–D4 of the board to pin D0 of each device.

A program runs on the MCU board, which performs specific actions for each pin D0–D4. Thus, when the Fire Monitor detects fire, the Fire Sprinkler is activated, the Piezo Speaker is triggered, the window opens, and the message "FIRE DETECTED" is displayed

```
from gpio import *
from time import *

def main():
    pinMode(0,INPUT)
    pinMode(1,OUT)
    print("Fire Alarm System");
    while True:
        fire=digitalRead(0);
        print(fire);
        if(fire==1023):
            customWrite(1,'1');
            customWrite(4,"FIRE DETECTED");
            customWrite(3,HIGH);
            digitalWrite(2,HIGH);
        else:
            customWrite(1, '0');
            customWrite(4, "SAFE MODE");
            customWrite(3,LOW);
            digitalWrite(2,LOW);


if __name__ == "__main__":
    main()

```

Smart Temperature Control

Energy efficiency – autonomous regulation of temperature and lighting ensures that energy is used only when needed.

An appropriate temperature is maintained during class hours. Teachers and students enjoy a comfortable environment, while energy consumption is reduced since the system runs only when necessary.

If the temperature rises:

- the heater is turned on;
- the LCD screen displays the message "HEATER ON".

If the temperature drops:

- the cooler is turned on;
- the LCD screen displays the message "AC ON".

If the temperature is normal:

- neither the cooler nor the heater is turned on;
- the LCD screen displays the message "NORMAL TEMP".

The SBC board is connected to four devices: a temperature sensor, a cooler, a heater, and an LCD screen. A thermometer is also used to show the exact temperature.

A program runs on the SBC board, which performs specific actions for each pin D0–D3. Thus, when the temperature sensor detects excessive heat, the cooler is activated, and the message "AC ON" is displayed on the LCD.

```
from gpio import *
from time import *

def main():
    pinMode(0,INPUT)
    pinMode(1, OUT)
    pinMode(2, OUT)
    pinMode(3, OUT)
    print("SMART ROOM TEMPERATURE")
    while True:
        temp = digitalRead(0);
        print("Temperature", temp);
        if (temp>=510):
            digitalWrite(1,HIGH);
            digitalWrite(2,LOW);
            customWrite(3, "AC-ON");
        elif(temp<500):
            digitalWrite(2,HIGH);
            digitalWrite(1,LOW);
            customWrite(3, "HEATER-ON");
        else:
            digitalWrite(2,LOW);
            digitalWrite(1,LOW);
            customWrite(3, "Normal-Temp");

if __name__ == "__main__":
    main()
```

Access Control

This automated system enhances both security and efficiency in a smart school by allowing access only when presence is detected. When the motion sensor identifies students or staff nearby, the system activates:

- automatic door opening;
- video camera activation for monitoring;
- lamp activation to illuminate the area;
- display of the message "Welcome to School" on the LCD screen.

In the absence of any detected motion, all devices automatically shut down:

- the camera is deactivated;

- the door closes;
- the light turns off;
- the LCD screen displays the message "DOOR CLOSED".

This ensures the area remains secure when unoccupied, while also saving energy efficiently.

A program runs on the SBC board, which performs specific actions for each pin D0–D4. When the motion sensor detects presence, the camera turns on, the door opens automatically, the lamp turns on, and the LCD displays the message "WELCOME TO SMART SCHOOL". In the absence of motion, all three devices are turned off, and the LCD displays "DOOR CLOSED". The code ensures the automated operation of the access system, contributing to the school's security and energy efficiency.

```
from gpio import *
from time import *

def main():
    pinMode(0,INPUT)
    pinMode(1, OUT)
    pinMode(2, OUT)
    pinMode(3, OUT)
    pinMode(4, OUT)
    print("SMART DOOR SYSTEM")
    while True:
        d=digitalRead(0);
        print(d);
        if(d==1023):
            customWrite(1,HIGH);
            customWrite(2,1,0);
            customWrite(3,2);
            customWrite(4,"WELCOME TO SMART SCHOOL");
        else:
            customWrite(1,LOW);
            customWrite(2,0,0);
            customWrite(3,0);
            customWrite(4, "DOOR-CLOSED");


        delay(1000);

if __name__ == "__main__":
    main()
```

Educational LED System

The programmable LED system contributes to an interactive and organized educational environment. The LEDs change color or intensity depending on the moments of the lesson, providing students with clear visual cues. For example:

- Green can signal the start of a break;
- Yellow can mark the time allocated for experiments;

- Blue can indicate the team work period;
- Red can warn that the end of the class is approaching.

These visual cues help students follow the flow of classroom activities more easily, without verbal interruptions, supporting a smooth and well-structured working rhythm.

A program runs on the SBC board that controls multiple pins (D0–D6), each associated with an LED. The program lights up the LEDs in a sequence of different colors (green, blue, red, etc.), each accompanied by a message displayed on the LCD screen indicating the current activity: break, team work, experiment, and so on.

In this way, the system helps students navigate the lesson using clear visual signals without disrupting the teaching flow.

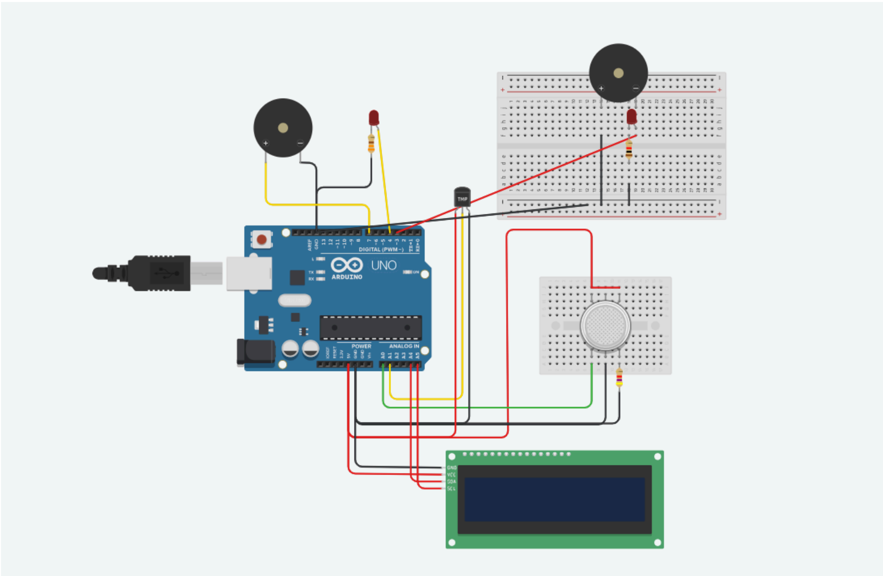
```
from gpio import *
from time import *

def main():
    pinMode(0, OUT)
    pinMode(1, OUT)
    pinMode(2, OUT)
    pinMode(3, OUT)
    pinMode(4, OUT)
    pinMode(5, OUT)
    pinMode(6, OUT)
    print("LEDs - Blinking")
    while True:
        digitalWrite(0, HIGH);
        customWrite(1,"GREEN LED-ON");
        delay(1000)
        digitalWrite(0, LOW);
        customWrite(1,"GREEN LED-OFF");
        delay(1000)

        digitalWrite(2, HIGH);
        customWrite(1,"BLUE LED-ON");
        delay(1000)
        digitalWrite(2, LOW);
        customWrite(1,"BLUE LED-OFF");
        delay(1000)

        digitalWrite(3, HIGH);
        customWrite(1,"SMART LED-ON");
        delay(1000)
        digitalWrite(3, LOW);
        customWrite(1,"SMART LED-OFF");
        delay(1000)
```

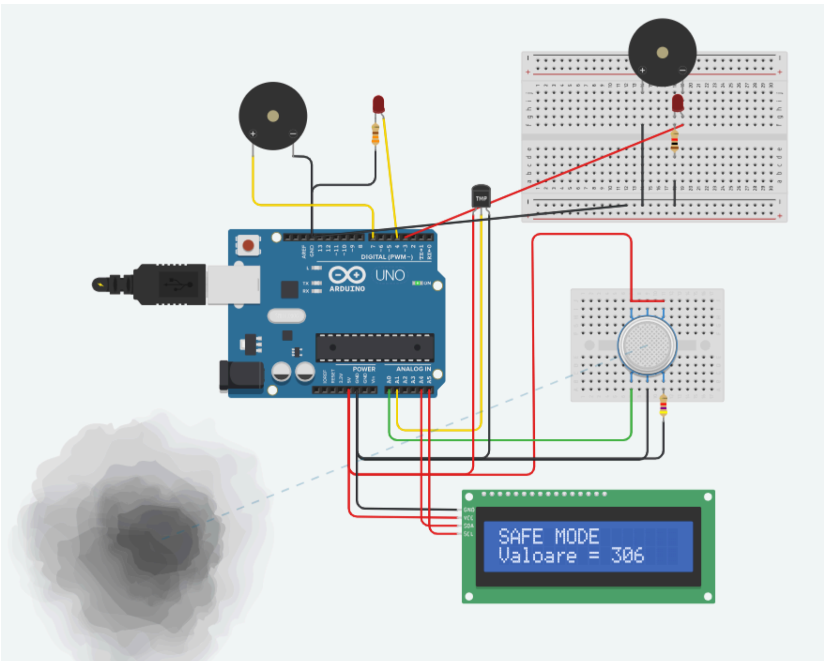
```
        delay(1000)



        digitalWrite(4, HIGH);
        customWrite(1,"RED LED-ON");
        delay(1000)
        digitalWrite(4, LOW);
        customWrite(1,"RED LED-OFF");
        delay(1000)

        digitalWrite(5, HIGH);
        customWrite(1,"GREEN LED-ON");
        delay(1000)
        digitalWrite(5, LOW);
        customWrite(1,"GREEEN LED-OFF");
        delay(1000)

        digitalWrite(6, HIGH);
        customWrite(1,"BLUE LED-ON");
        delay(1000)
        digitalWrite(6, LOW);
        customWrite(1,"BLUE LED-OFF");
        delay(1000)

if __name__ == "__main__":
    main()
```

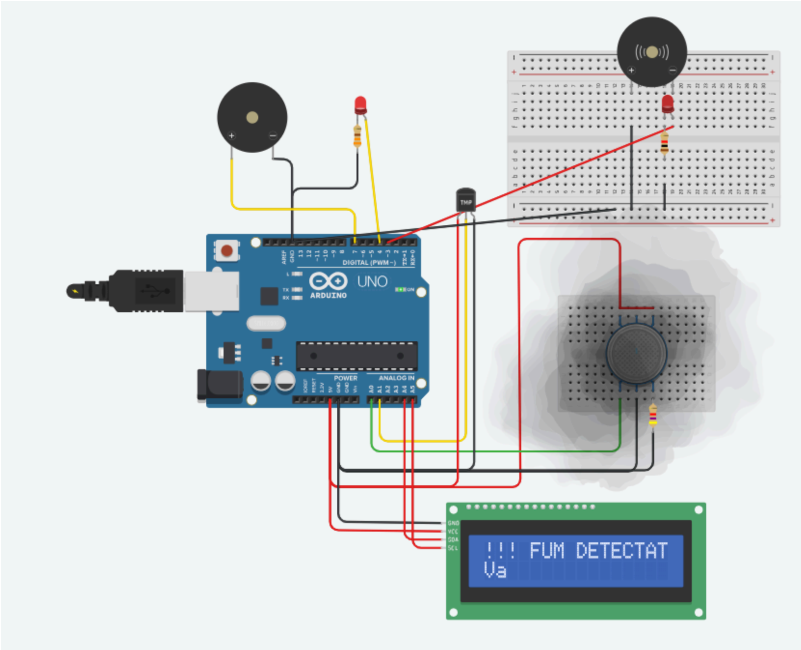
4. TinkerCad

https://www.tinkercad.com/things/fSNu3kiOizv-fire-alarm-system/editel?returnTo=https%3A%2F%2Fwww.tinkercad.com%2Fdashboard&sharecode=s8Se5dgA1QNV9rqZ4FfH4xLcF2Sgxt8FCYpXJtSsJpM

a. Designed architecture







b. Functionalities implemented in TinkerCad

Powering the Board and Breadboard

The Arduino is connected to the USB port for power.

From the Arduino, two wires are connected: a red one to the positive rail of the breadboard and a black one to the ground rail. All other components are powered from these two lines.

Gas/Smoke Sensor

It receives power from the positive rail and ground from the ground rail.

It sends an analog signal to the Arduino that varies based on the smoke/gas concentration.

Piezo Buzzer

Connected to a digital pin on the Arduino and to ground.

When the sensor detects smoke above a certain level, the buzzer emits a loud sound alert.

Warning LED

Mounted with a resistor and connected to a digital pin and to ground.

It lights up at the same time as the buzzer, providing an additional visual cue when danger is detected.

I2C LCD Display

Connected with four wires: power, ground, data line, and clock line.

It shows either the message “SMOKE DETECTED” along with the measured value or “SAFE MODE” when there is no danger.

Operational Flow

The Arduino continuously reads the value from the sensor in a loop.

If the value exceeds the preset threshold, it activates the buzzer and LED, and changes the LCD message to “SMOKE DETECTED”.

If the value is below the threshold, it turns off the sound and LED, and displays “SAFE MODE” on the LCD.

After a short delay, the cycle repeats automatically.