INDUSTRIAL FIRE SAFETY

DESCRIPTION: Develop a prototype system which can generate and evacuate the workers under fire accident.

PROBLEM STATEMENT: There is a serious safety risk because there is currently no effective, automated system in place to plan and produce worker evacuations in the event of a fire. Planning an evacuation manually takes time and can cause delays, which makes it more difficult to respond quickly. By creating a prototype system that automates evacuation planning and execution, this project seeks to overcome these issues and improve worker safety and efficiency during fire situations.

SCOPE OF SOLUTION: The scope of the solution includes creating a prototype system that tackles important elements of worker evacuation in the event of a fire:

Risk assessment and facility monitoring: Examine the building in detail to find any possible fire dangers and gauge the level of risk. Install monitoring systems to assess the severity and spread of flames and to detect them in real time.

Creation of Evacuation Plans: Provide an algorithm or software module to create the best possible evacuation plans based on real-time data, taking into account safe zones, worker positions, evacuation routes, and fire locations. Plans must to put safety first, cut down on evacuation time, and take into consideration the different abilities and preferences of the workers.

Route Optimisation and Guidance: Create a function that allows evacuation routes to be dynamically optimised in response to shifting fire conditions and traffic volumes. Using physical devices or mobile applications, give employees real-time direction so they may take the quickest and safest routes to departure points.

Establish a communication system to notify employees about the fire, start the evacuation process, and give them ongoing instructions and updates. To effectively contact employees, make use of a variety of communication channels, including SMS, smartphone apps, PA systems, and alarms.

Integration with Safety Systems: Link the evacuation system to the current safety framework, which consists of emergency response plans, access controls, and fire detection systems. Make sure that the evacuation system and these safety precautions work together seamlessly and share information.

Testing and Simulation: Use simulated fire scenarios and, if feasible, actual fire drills to conduct thorough testing of the prototype. Analyse the system's overall performance and the efficacy, correctness, and dependability of the evacuation plans in various fire disaster situations.

User Training and Documentation: Create thorough training manuals and hold seminars to instruct facility personnel on the proper operation of the evacuation system. To help users comprehend the capabilities and functions of the system, provide user manuals and documentation.

Future Enhancements and Scalability: Build the system with future growth, more features, and integration with developing technologies in mind. Think about upcoming improvements like AI-based evacuation analytics and predictive modelling for fire behaviour for ongoing improvement.

REQUIRED COMPONENTS:

Hardware Components:

1. Sensors and Detectors: Smoke/Gas Sensor
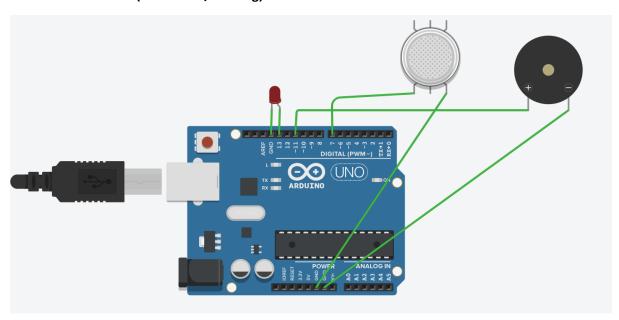
2. Microcontroller: Arduino UNO R3

3. Alert Systems: LED, Buzzer

4. Power Supply

Software Components:

1. Arduino IDE

2. C/C++ for microcontroller programming

3. TinkerCAD

SIMULATED CIRCUIT (TinkerCAD/ Fritzing)

CODE FOR THE SOLUTION

```
const int flameSensorPin = 7;  // Pin connected to Flame Sensor

const int ledPin = 13;       // Pin connected to LED

const int buzzerPin = 11;     // Pin connected to Buzzer

void setup() {

  pinMode(flameSensorPin, INPUT);

  pinMode(ledPin, OUTPUT);

  pinMode(buzzerPin, OUTPUT);

}
```

```
void loop() {
 int flameValue = digitalRead(flameSensorPin);
 // Flame sensor is active LOW, so LOW means flame detected
 if (flameValue == LOW) {
  digitalWrite(ledPin, HIGH);    // Turn on LED
  tone(buzzerPin, 1000);        // Play a tone on the buzzer
  delay(500);                    // Delay for half a second
  digitalWrite(ledPin, LOW);    // Turn off LED
  noTone(buzzerPin);            // Stop the buzzer tone
  delay(500);                    // Delay for half a second
 } else {
  digitalWrite(ledPin, LOW);    // Turn off LED
  noTone(buzzerPin);            // Stop the buzzer tone
 }
}
```