Assignment No. 05

Title - Write a Program to design and develop a user interface for monitoring and controlling CPS system

Problem Statement - Design and develop a user interface for monitoring and controlling CPS system on Tinkercad.

Prerequisite – C/C++ programming, basic understanding of conditionals, loops, and Tinkercad environment.

Software Requirements - Tinkercad simulation platform, internet connection, computer or compatible device.

Hardware Requirements - Arduino Uno R3, Resistor, LED, Breadboard Small, Push Buttons, Servo Motor**,** Jumper Wires, potentiometer

Learning Objectives - Implement Smart parking system

Outcomes - After Completion of this assignment students

Theory - The Industrial Internet of Things (IIoT) represents a significant advancement in connecting and managing a wide range of devices and sensors to enhance operational efficiency and data collection. However, with these advancements come substantial security concerns, as interconnected systems are vulnerable to unauthorized access, data breaches, and other cyber threats. Implementing robust security measures is essential for protecting these systems. This experiment demonstrates how to design and implement security measures in an IIoT system using Arduino, breadboard, various sensors (gas and temperature), resistors, LEDs, LCDs, a potentiometer, a piezo buzzer, and jumper wires, all within the Tinkercad simulation environment.

Components and Their Functions

1. **Arduino Microcontroller:**

Function: Acts as the central processing unit of the IIoT system. It handles data from sensors, processes security measures, and controls outputs based on programmed security protocols.

2. Breadboard:

Function: Provides a platform for prototyping the circuit without soldering. It allows for easy connections and modifications of the components.

3.Resistors:

Function: Control the flow of current through various components, protecting sensitive parts from damage and ensuring proper operation.

4.Push Buttons:

The push button switch is usually used to turn on and off the control circuit, and it is a kind of control switch appliance that is widely used. It is used in electrical automatic control circuits to manually send control signals to control contactors, relays, electromagnetic starters, etc .

1. Servo Motor**:**

   Function: A servomotor is a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity and acceleration. It consists of a suitable motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller (Arduino UNO in our case), often a dedicated module designed specifically for use with servomotors.
2. Jumper Wires:

Function: Connect various components on the breadboard and to the Arduino, ensuring proper circuit connectivity.

7. LCD Display**: LCD (Liquid Crystal Display) is a type of flat panel display which uses liquid crystals in its primary form of operation. LEDs have a large and varying set of use cases for consumers and businesses, as they can be commonly found in smartphones, televisions, computer monitors and instrument panel.**

Circuit Design and Implementation

3.Building the Circuit:

Breadboard Setup: Arrange the Arduino, sensors, and other components on the breadboard. Connect the gas sensor, temperature sensor, and potentiometer to the Arduino using jumper wires.

Wiring: Connect the sensors' output pins to Arduino analog or digital input pins. Connect the LCD, LEDs, and piezo buzzer to appropriate output pins on the Arduino.

Power Supply: Ensure that the power supply to the Arduino and components is stable and within the required voltage range.

4.Testing and Calibration:

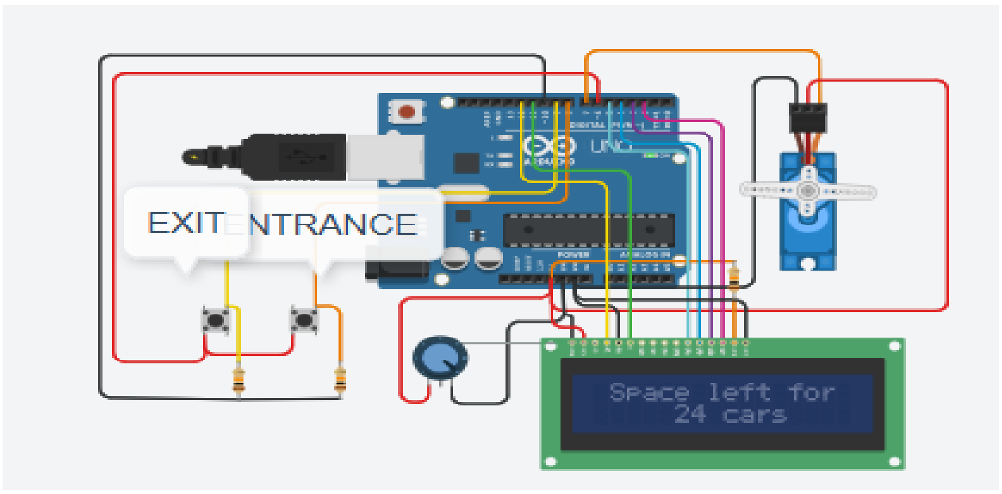
Component Testing: Test each component individually to ensure proper operation. Check sensor readings, authentication processes, and alert mechanisms.

Source Code –

```
#include <Servo.h>
#include<LiquidCrystal.h>
LiquidCrystal lcd(12,11,5,4,3,2);
Servo myservo;

#define ServoM    7       //Connected to the servo motor.
#define Exit      9       //Pin connected to the EXIT sensor.
#define In        8       //Pin connected to the IN sensor.
#define Pwr  6            //Extra power pin for sensors(Don't connect servo's power to this!)
#define Gnd  10           //Extra groung pin for sensors(Don't connect servo's power to this!)
#define BarLow    90      //Low position of the barrier.
#define BarUp     177     //Up position of the barrier.
#define CAPACITY  24      //Capacity of the parking lot.

void setup(){
  myservo.attach(ServoM);
  lcd.begin(16,2);
  lcd.print("  BatStateU Parking!");
  pinMode(Gnd, OUTPUT);
  pinMode(Pwr, OUTPUT);
  pinMode(Exit, INPUT);
  pinMode(In, INPUT);
  digitalWrite(Gnd, LOW);
  digitalWrite(Pwr, HIGH);
  myservo.write(BarLow);
 delay(1000);
}
```

```
int  Available= 24;

void loop(){

if (Available == 1){
  lcd.clear();
lcd.setCursor(1,0);
lcd.print("Space left for");
lcd.setCursor(0,1);
lcd.print(" ");
lcd.print(" ");
lcd.print(" ");
lcd.print(" ");
lcd.print(" ");
lcd.print(Available);
lcd.print(" car");
delay(1000);
}else{
if (Available >= 1){
lcd.clear();
lcd.setCursor(1,0);
lcd.print("Space left for");
lcd.setCursor(0,1);
lcd.print(" ");
lcd.print(" ");
lcd.print(" ");
lcd.print(" ");
lcd.print(" ");
lcd.print(Available);
lcd.print(" cars");
}else{
  lcd.clear();
    lcd.setCursor(1,0);
    lcd.print("    Sorry!");
    lcd.setCursor(0,1);
              lcd.print(" No Place left!");
              delay(1000);
}
}
if(digitalRead(In))
{
  if(Available != 0){
    Available--;
    myservo.write(BarUp);
    delay(3000);
    myservo.write(BarLow);
    }
  }
if(digitalRead(Exit))
{
  if(Available != CAPACITY){
    Available++;
    myservo.write(BarUp);
    delay(3000);
    myservo.write(BarLow);
    }
}
delay(80);
}
```

Circuit Diagram –



Conclusion - This experiment demonstrates the implementation of car Parking system in an IIoT system using Arduino and various components within Tinkercad.