```cpp
#include <SPI.h>

#include <MFRC522.h>

#include <Servo.h>

#include <Wire.h>

#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27,16,2);

#define SS_PIN 10

#define RST_PIN 9


Servo myservo;


int yellow_led = 7;

int red_led = 5;


int IR1 = 2;

int IR2 = 4;


int Slot = 1;


int flag1 = 0;

int flag2 = 0;


int Buzzer = 8;


MFRC522 rfid(SS_PIN, RST_PIN);
```

```
MFRC522::MIFARE_Key key;


void setup() {
  //put your setup code here, to run code:
  Serial.begin(9600);
  SPI.begin();
  rfid.PCD_Init();
  lcd.begin();
  lcd.backlight();
  pinMode(IR1, INPUT);
  pinMode(IR2, INPUT);
  pinMode(7, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(8, OUTPUT);
  myservo.attach(3);
  myservo.write(0);


  lcd.setCursor (0,0);
lcd.print("    VEHICLE    ");
lcd.setCursor (0,1);
lcd.print(" PARKING SYSTEM ");
delay (2000);
lcd.clear();



  Serial.println("waiting for card...");
}
```

```cpp
void loop() {

 //put your main code here, to run repeatedly:

 if (!rfid.PICC_IsNewCardPresent() || !rfid.PICC_ReadCardSerial())

 return;


 //Serial.print(F("PICC type: "));

 MFRC522::PICC_Type piccType = rfid.PICC_GetType(rfid.uid.sak);

 //Serial.println(rfid.PICC_GetTypeName(piccType));


 //Check is the PICC of Classic MIFAREtype

 if (piccType != MFRC522::PICC_TYPE_MIFARE_MINI &&

    piccType != MFRC522::PICC_TYPE_MIFARE_1K &&

    piccType != MFRC522::PICC_TYPE_MIFARE_4K) {

     Serial.println(F("Your tag is not of type MIFARE Classic."));

     return;

    }

    String strID = "";

    for (byte i = 0; i < 4; i++) {

     strID +=

     (rfid.uid.uidByte[i] < 0x10 ? "0" : "") +

     String(rfid.uid.uidByte[i], HEX)+

     (i != 3 ? ":" : "");

    }

    strID.toUpperCase();

    //Serial.print("Tap card key: ");

    //Serial.println(strID);

    delay(500);
```

```
    if(digitalRead (IR1) == LOW && flag1==0){

     if(Slot>0){flag1=1;

     if(flag2==0){strID.indexOf("F2:03:4C:1A") >= 0; Slot=Slot-1;}


     Serial.println("Authorised access");


     digitalWrite(7, HIGH);

     digitalWrite(5, LOW);

     digitalWrite(8, HIGH);

     delay(100);

     digitalWrite(8, LOW);

     myservo.write(90); // motor moves 90 degree


     }else{
lcd.setCursor (0,0);

lcd.print("   SORRY :(   ");

lcd.setCursor (0,1);

lcd.print("  Parking Full  ");

delay (3000);

lcd.clear();

}

}

   if(digitalRead (IR2) == LOW && flag2==0){flag2=1;

   delay(2000);

    if(flag1==0){myservo.write(90); Slot = Slot+1;}

}
```
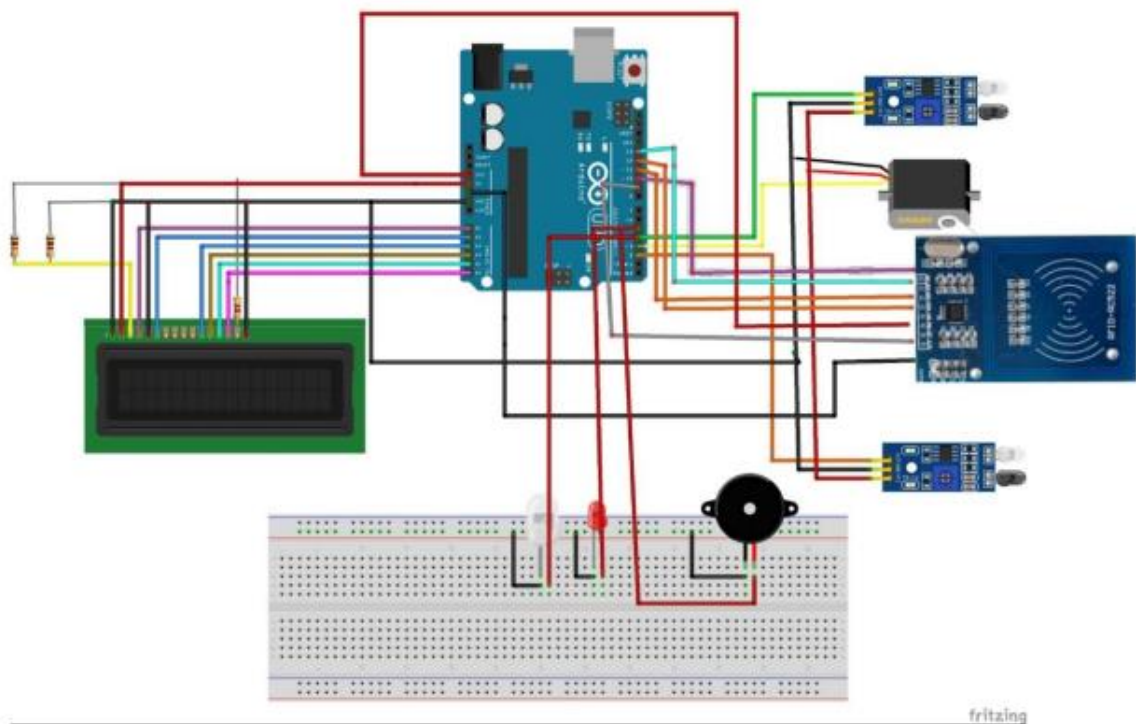
```
if(flag1==1 && flag2==1){

     digitalWrite(7, HIGH);

    digitalWrite(5, LOW);

    digitalWrite(8, HIGH);

    delay(100);

    digitalWrite(8, LOW);

delay (100);

myservo.write(0);

flag1=0, flag2=0;

}




lcd.setCursor (0,0);

lcd.print("   WELCOME!   ");

lcd.setCursor (0,1);

lcd.print("Slot Left: ");

lcd.print(Slot);

}
```

RFID Reader to Arduino uno Board

• SDA of RFID to Arduino uno Pin 10

• SCK of RFID to Arduino uno Pin 13

• MOS1 of RFID to Arduino uno Pin 11

• MOS0 of RFID to Arduino uno Pin 12

• RST of RFID to Arduino uno Pin 9

• GND of RFID to Arduino uno GND

• 3.3V of RFID to Arduino uno 3.3V

Servo Motor to Arduino uno Board

• Orange wire of servomotor to Arduino Uno Board pin 3

 • Brown wire of servomotor to Arduino Uno Board GND

• Red wire of servomotor to Arduino Uno Board 5V

Buzzer to Arduino Uno Board

• Positive to Arduino Uno Board Pin 8

• Negative to Arduino Uno Board GND

IR Sensor to Arduino Uno Board IR Sensor 1

• IR Sensor 1 GND to Arduino Uno Board GND

• IR Sensor 1 VCC to Arduino Uno Board VCC

• IR Sensor 1 Output to Arduino Uno Board Pin 2

IR Sensor 2

• IR Sensor 2 GND to Arduino Uno Board GND

• IR Sensor 2 VCC to Arduino Uno Board VCC

• IR Sensor 2 Output to Arduino Uno Board Pin 4

I2C LCD display to Arduino Uno Board •

 GND of I2C LCD display to Arduino Uno Board GND

• VCC of I2C LCD display to Arduino Uno Board 5V

• SDA of I2C LCD display to Arduino Uno Board Pin A4

• SCL of I2C LCD display to Arduino Uno Board Pin A5