**Automatic Street Lights and Parking System with Voice or Remote Controlling Car**

**CODE for Automatic Street Lights and Parking System**

#include <Servo.h>

Servo myservo;

int pos = 0;

#include <LiquidCrystal.h>

LiquidCrystal lcd(13,12,11,10,9,8);

char user1[12]={"$0001242566"};

char blankarr[12];

int count = 0;

const int entry=27;

const int slot1= 22;

const int slot2= 23;

const int slot3= 24;

const int slot4= 25;

const int slot5= 26;

char result=0,n=0,d1,state,e,s1,s2,s3,s4,s5;

char f=0,g=0,h=0,i=0,j=0;

char compare( char \*b1)

{ char b2,b3,b5,b6,b7,b8,b9,b10;

b5=0;b6=0;

for(b2=0;b2<11;b2++)

{if(\*b1==user1[b2])

{ b5++; }

b1++; }

if(b5>9)

{ b5=0;

b3='A';

n=1; }

else

{ b5=0;

b3=1; }

return (b3); }

void slot\_check()

{ if(s1==1)

{ lcd.setCursor(6,1);

lcd.print('1');

f=1; }

if(s1==0)

{ lcd.setCursor(6,1);

lcd.print(' ');

f=0; }

if(s2==1)

{ lcd.setCursor(8,1);

lcd.print('2');

g=2; }

if(s2==0)

{ lcd.setCursor(8,1);

lcd.print(' ');

g=0; }

if(s3==1)

{ lcd.setCursor(10,1);

lcd.print('3');

h=3; }

if(s3==0)

{ lcd.setCursor(10,1);

lcd.print(' ');

h=0; }

if(s4==1)

{ lcd.setCursor(12,1);

lcd.print('4');

i=4; }

if(s4==0)

{ lcd.setCursor(12,1);

lcd.print(' ');

i=0; }

if(s5==1)

{ lcd.setCursor(14,1);

lcd.print('5');

j=5; }

if(s5==0)

{ lcd.setCursor(14,1);

lcd.print(' ');

j=0; } }

void setup()

{ Serial.begin(9600);

pinMode(entry,INPUT);

pinMode(slot1,INPUT);

pinMode(slot2,INPUT);

pinMode(slot3,INPUT);

pinMode(slot4,INPUT);

pinMode(slot5,INPUT);

myservo.attach(6);

myservo.write(0);

lcd.begin(16, 2);

// Print a message to the LCD.

lcd.print(" CAR PARKING ");

lcd.setCursor(0, 1);

lcd.print(" SYSTEM ");

delay(4000);

lcd.clear(); }

void loop()

{ slot\_check();

e=digitalRead(entry);

s1=digitalRead(slot1);

s2=digitalRead(slot2);

s3=digitalRead(slot3);

s4=digitalRead(slot4);

s5=digitalRead(slot5);

slot\_check();

if((s1==0)&&(s2==0)&&(s3==0)&&(s4==0)&&(s5==0))

{ lcd.setCursor(0, 0);

lcd.print("PARKING FULL...."); }

if((s1==1)||(s2==1)||(s3==1)||(s4==1)||(s5==1))

{ lcd.setCursor(0, 0);

lcd.print(" "); }

if(e==0)

{ lcd.clear();

lcd.setCursor(0, 0);

lcd.print("SCAN CARD......");

delay(4000);

lcd.clear();

slot\_check();

if(Serial.available())

{ count = 0;

while(Serial.available() && count < 12) // Read 12 characters and store them in input array

{ blankarr[count] = Serial.read();

//Serial.print(blankarr[count]);

count++;

delay(5);

result=1; } }

if(result==1)

{ result=0;

d1=compare(&blankarr[0]);

if(d1==1)

{ d1=0; }

else

{ if(n==1)

{ switch(d1)

{ case 'A':

{ slot\_check();

if((f>g)&&(f>h)&&(f>i)&&(f>j))

{ lcd.clear();

lcd.setCursor(0,0);

lcd.print(" SLOT ASSIGNED: ");

lcd.setCursor(0,1);

lcd.print(" NUMBER 1 ");

delay(5000);

lcd.clear(); }

if((g>f)&&(g>h)&&(g>i)&&(g>j))

{ lcd.clear();

lcd.setCursor(0,0);

lcd.print(" SLOT ASSIGNED: ");

lcd.setCursor(0,1);

lcd.print(" NUMBER 2 ");

delay(5000);

lcd.clear(); }

if((h>g)&&(h>f)&&(h>i)&&(h>j))

{ lcd.clear();

lcd.setCursor(0,0);

lcd.print(" SLOT ASSIGNED: ");

lcd.setCursor(0,1);

lcd.print(" NUMBER 3 ");

delay(5000);

lcd.clear(); }

if((i>g)&&(i>h)&&(i>f)&&(i>j))

{ lcd.clear();

lcd.setCursor(0,0);

lcd.print(" SLOT ASSIGNED: ");

lcd.setCursor(0,1);

lcd.print(" NUMBER 4 ");

delay(5000);

lcd.clear(); }

if((j>g)&&(j>h)&&(j>i)&&(j>f))

{ lcd.clear();

lcd.setCursor(0,0);

lcd.print(" SLOT ASSIGNED: ");

lcd.setCursor(0,1);

lcd.print(" NUMBER 5 ");

delay(5000);

lcd.clear(); }

for (pos = 0; pos <= 90; pos += 1)

{ myservo.write(pos);

delay(15); }

delay(5000);

for (pos = 90; pos >= 0; pos -= 1)

{ myservo.write(pos);

delay(15); }

break; } } } } } } }

**CODE for Voice or Remote Controlling Car**

int incomingByte=0;

const int motora=2;

const int motorb=3;

const int motorc=4;

const int motord=5;

const int in1=8;

const int in2=9;

const int in3=10;

const int in4=11;

void setup() {

pinMode(motora, OUTPUT);

pinMode(motorb, OUTPUT);

pinMode(motorc, OUTPUT);

pinMode(motord, OUTPUT);

pinMode(in1, INPUT);

pinMode(in2, INPUT);

pinMode(in3, INPUT);

pinMode(in4, INPUT);

digitalWrite (motora,LOW);

digitalWrite (motorb,LOW);

digitalWrite (motorc,LOW);

digitalWrite (motord,LOW);

Serial.begin(9600); }

void loop() {

int a=digitalRead(in1);

int b=digitalRead(in2);

int c=digitalRead(in3);

int d=digitalRead(in4);

if (Serial.available() > 0) {

int a=digitalRead(in1);

int b=digitalRead(in2);

int c=digitalRead(in3);

int d=digitalRead(in4);

// read the incoming byte:

incomingByte = Serial.read();

if(incomingByte=='S') ///// go forward

{ digitalWrite (motora,LOW);

digitalWrite (motorb,LOW);

digitalWrite (motorc,LOW);

digitalWrite (motord,LOW); }

if(incomingByte=='R')///// take back

{ digitalWrite (motora,HIGH);

digitalWrite (motorb,LOW);

digitalWrite (motorc,LOW);

digitalWrite (motord,LOW); }

if(incomingByte=='L') ///// go forward

{ digitalWrite (motora,LOW);

digitalWrite (motorb,LOW);

digitalWrite (motorc,HIGH);

digitalWrite (motord,LOW); }

if(incomingByte=='F')///// take back

{ digitalWrite (motora,HIGH);

digitalWrite (motorb,LOW);

digitalWrite (motorc,HIGH);

digitalWrite (motord,LOW); }

if(incomingByte=='B') ///// go forward

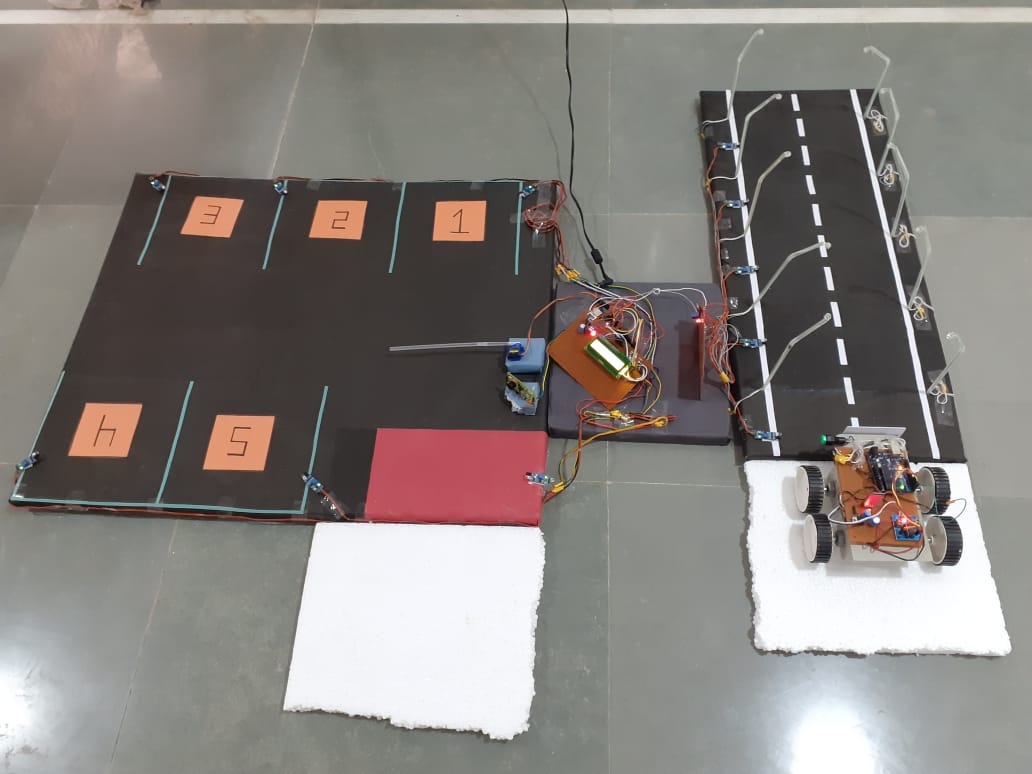
{ digitalWrite (motora,LOW);

digitalWrite (motorb,HIGH);

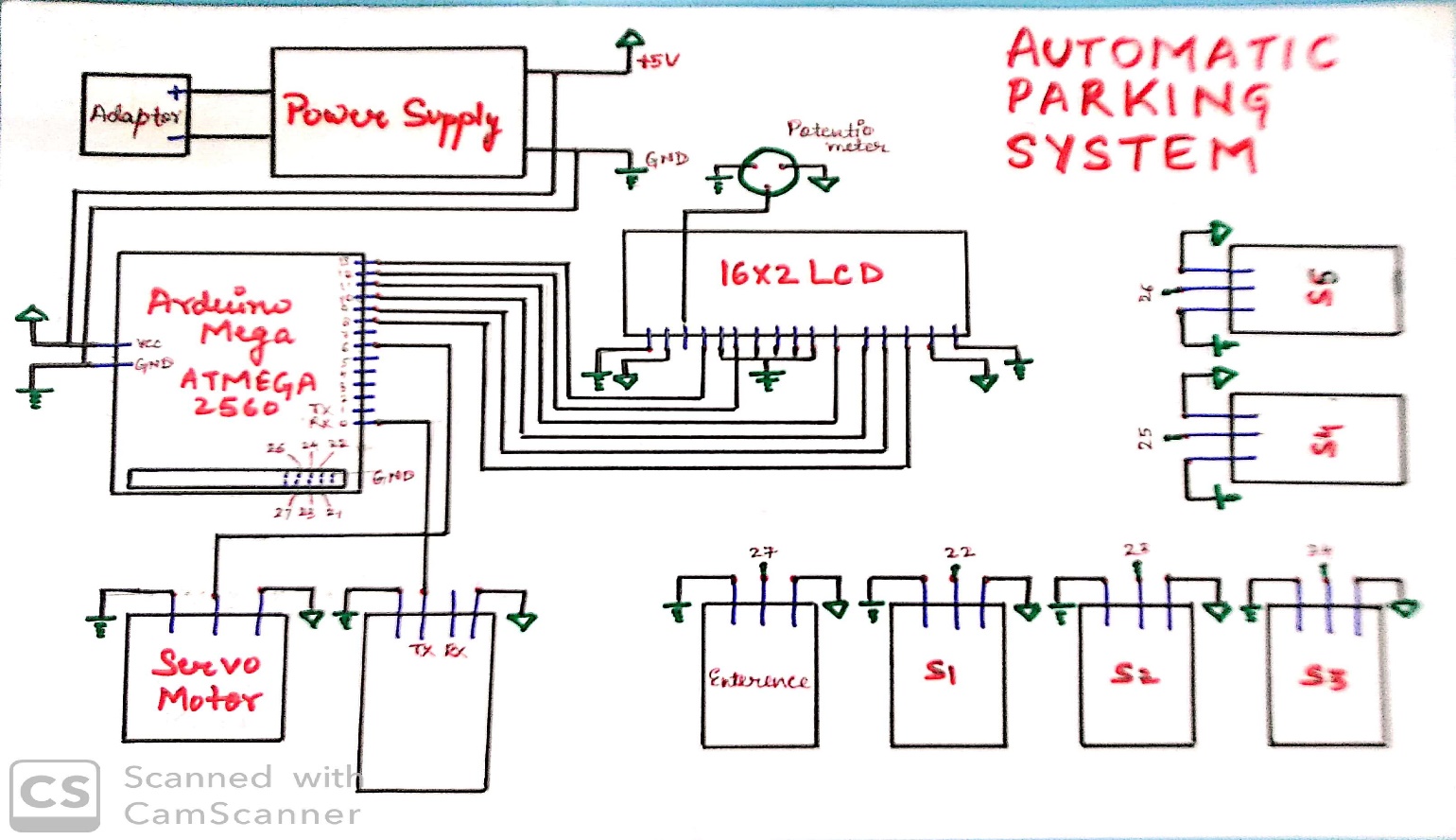
digitalWrite (motorc,LOW);

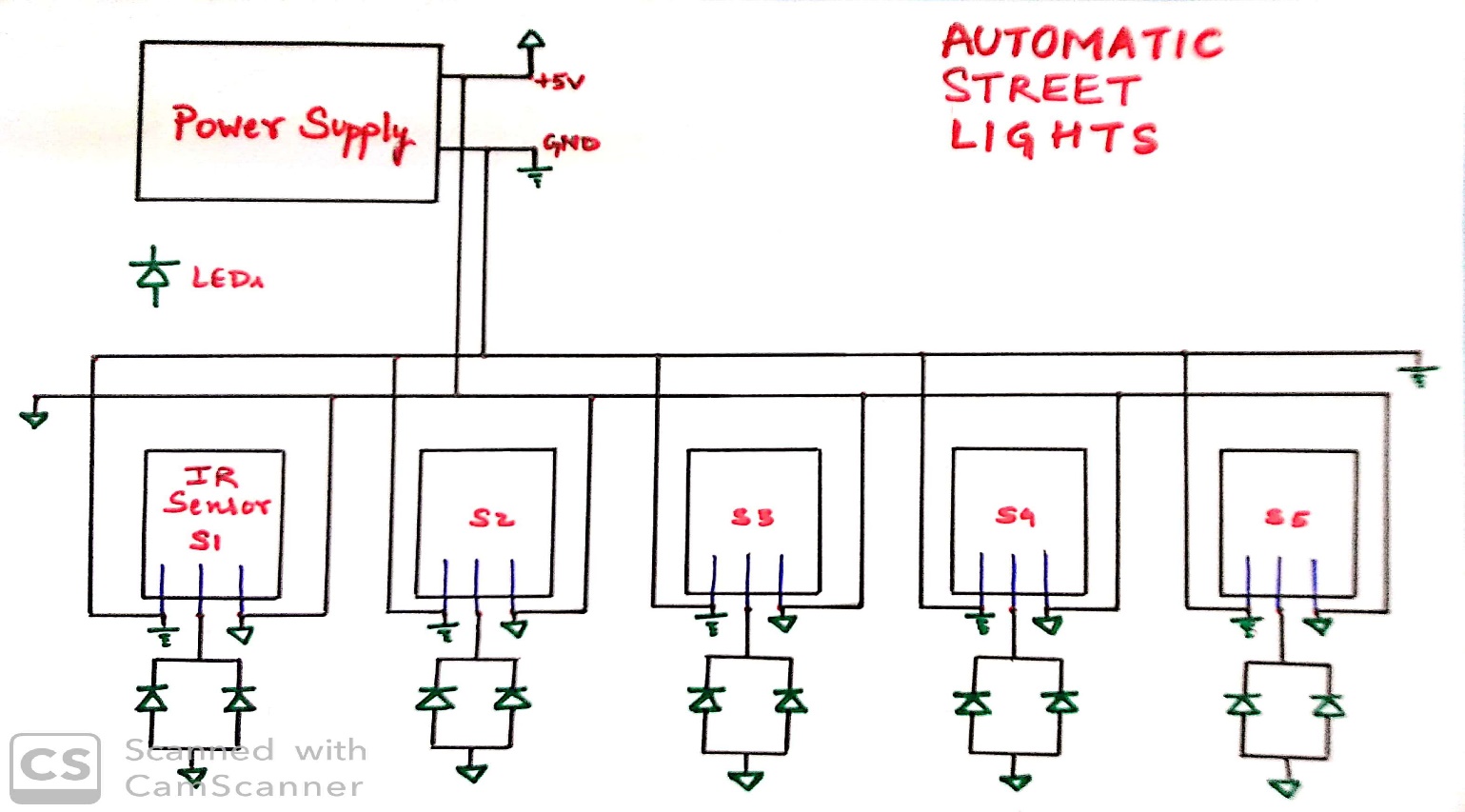
digitalWrite (motord,HIGH); } } }

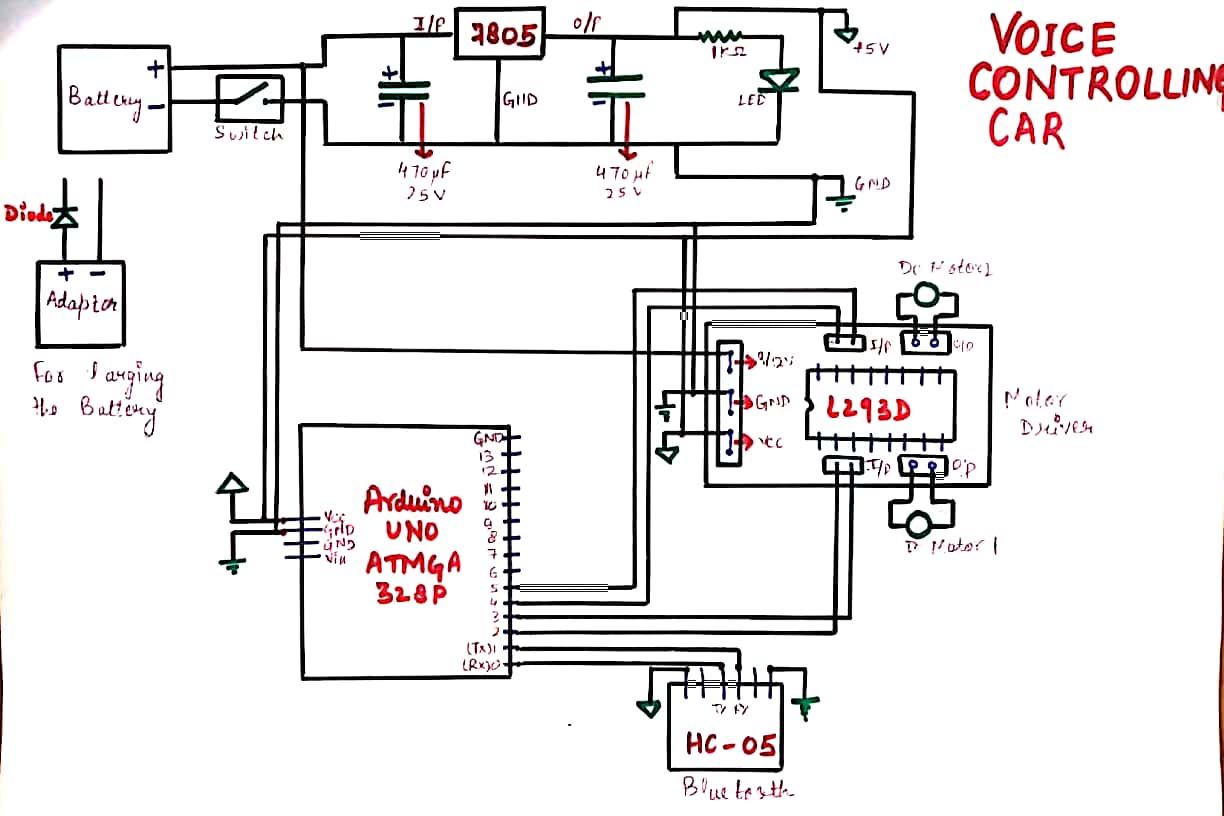
**MODEL PICTURE**

****

**CIRCUIT DIAGRAM**

****



****