KOCAELİ UNIVERSITY* FACULTY OF ENGINEERING

SMART PARK AND HEADLIGHT CONTROL SYSTEM

C# PROGRAMMING
PROJECT REPORT

Ozan Hamdi Kaplan

Department: Electronics and Communication Engineering

KOCAELİ, 2022

İNGREDİENTS

İNDEX OF SHAPES

ABBREVİATİONS

IoT	:Internet of Things
GHz	:Gigahertz
ISM	:Industrial Scientific and Medical
MBPS	:Megabit per second
API	:Application Programming İnterface
NET	:Network
FCL	:Full Container Load
RISC	:Reduced instruction set computer
I2C	:Inter-Integrated Circuit
SPI	:Serial peripheral interface
UART	:Universal Asynchronous Receiver Transmitter
USART	:Universal synchronous asynchronous receiver transmitter
ADC	:Analog Digital Converter
LCD	:Liquid-crystal display
RFID	:Radio frequency identification
IDE	:Integrated development environment
TI	:Texas Instruments
CCS	:Code Composer Studio
IR	:Infrared
PWM	:Pulse Width Modulation
A	:Amper
V	:Volt
W	:Watt
IC	:Integrated Circuit

1. LOGIN

Today, every object you can think of (phones, computers, home systems, robots, etc.) means that it somehow accesses the internet and communicates with other devices. But the Internet of Things is not just about devices that connect to the Internet. IoT devices communicate with people and let them know about any situation. The first application of the IoT concept in history has been developed until today, after the images of the coffee machine were shared over the internet by a group of academics from Cambridge University in 1991, thanks to a system with a camera. Objects can detect the system, communicate, record data, direct it to the destination address, etc. The fact that they can communicate with each other thanks to their abilities has actually been in our lives for a long time.

One of the most common application areas of IoT today is the municipal stops, metro stations, which we use almost every day, in other words Smart Stations.

It is the name of short range radio frequency technology that eliminates the Bluetooth cable connection. It was developed by Ericsson company in 1994 to wirelessly connect and communicate with mobile phones and other mobile devices. Bluetooth allows computers, peripherals and other devices to communicate with each other without a cable connection, even if they are out of line of sight. Bluetooth technology operates in the 2.4 Ghz ISM frequency band and can transmit voice and data. The effective range of Bluetooth-enabled devices capable of transmitting data up to 24 MBPS is about 10 to 100 meters.

A framework is a package of application development interfaces (APIs) and a shared library of code that programmers can call whenever they need it. In .NET Framework, the shared code library is called Framework Class Library (FCL). The codes in this library can run many kinds of functions. In this way, programmers do not have to write the necessary functions for small operations from scratch. An "application" development platform built on open Internet protocols and standards, developed by Microsoft. The scope of the application concept here is very broad. Everything from a desktop application to a web browser application has been conceived and supported within this platform.

MSP430 is a microcontroller produced by Texas Instrument company that stands out with its very low power consumption. It is a 16-bit RISC microcontroller in Von Neumann architecture. It contains many classic modules such as I2C, SPI, USART, ADC.

1.1 Literature Review

There were previous studies on IoT smart parking systems. Considering the general features, the parking lot occupancy, balance inquiry, automatic opening and closing speed controls of the obstacles, the lack of determining the number of parking spaces, the lack of balance loading, the lack of information with LCD, the design of the user interfaces, the lack of computer applications were taken into consideration and the project was designed according to these evaluations.

The goal of the project; With the rapid increase in the number of vehicles and the world population, simple tasks such as parking have become difficult. It is expected to worsen in the coming years, leading to conflicts such as lack of parking and traffic problems. With the development of technology, IoT plays a critical role in our daily life. The IoT made the idea of smart cities possible. It also plays an important role in smart parking systems. Here we propose an idea of the smart parking system using Radio Frequency Identification (RFID) technology. This system aims to replace the traditional parking system with high-tech, IoT-based smart parking using RFID. It is aimed to receive and send data using bluetooth serial port communication between the hardware and the computer, and to manage the problems of finding a place in the parking lots and loading the balance with the help of the application.

The project consists of 4 tasks and the tasks are listed below:

Task-1 : Speed control for barriers at entrances and exits
Task-2 : Determining the number of parking spaces
Task-3 : User balance check and upload
Task-4 : Adjusting the brightness of vehicle lights

1.2 Platforms Used

1.2.1 Software Platforms Used

1.2.1.1 Code Composer Studio

It is used to program the MSP430 microcontroller.

1.2.1.2 Microsoft Visual Studio

It is used for the operations to be performed according to the data coming from the integrated, sending the data from the computer to the integrated and creating the user interface.

1.2.1.3 Arduino IDE

It is used as a preliminary preparation in order to learn the functions of the sensors used.

1.2.1.4 Altium Designer

It is used for circuit schematic and pcb design.

1.2.2 Hardware Platforms Used

1.2.2.1 MSP430G2553 Launchpad

This microcontroller, which belongs to the TI (Texas Instruments) family, is used to code the G2553 IC.

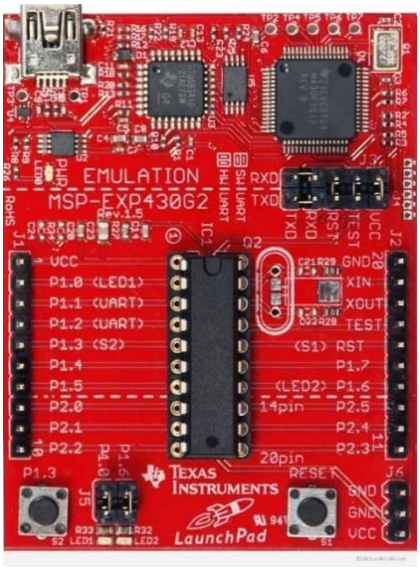

Figure 1:MSP430G2553 Launchpad

1.2.2.2 HC-05 Bluetooth Module

It is used for communication of G2553 ICs with computer and uart.

Figure 2: HC-05 Bluetooth Module

1.2.2.3 RFID Rc522 Sensor

It is used for reading user cards and balance checks.

Figure 3: RFID Rc522 Sensor

1.2.2.4 IR Obstacle Detection Sensor

It is used for vehicle status information in parking lots.

Figure 4: IR Obstacle Detection Sensor

1.2.2.5 LCD Led Display

It is used to inform the user.

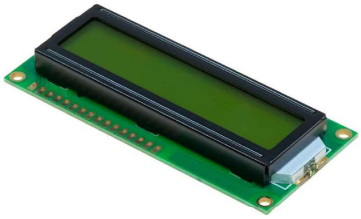

Figure 5: LCD Led Display

1.2.2.6 Servo Motor

Used as park entrance and exit barriers.

Figure 6: Servo Motor

1.2.2.7 G2553 Integrated

It is used for the assignment of the elements connected to the integrated.

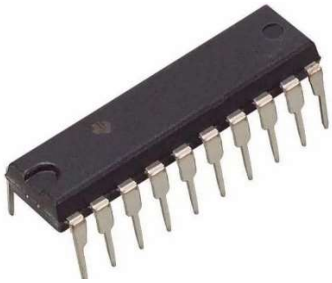

Figure 7: G2553 Integrated

1.2.2.8 LM2596 Adjustable Power supply

Used to keep 12V voltage at 5V and constant current.

Figure 8:LM2596 Adjustable Power supply

1.2.2.9 5V 3.3V Voltage Module

Used to reduce 12V voltage to 3.3V and 5V voltage.

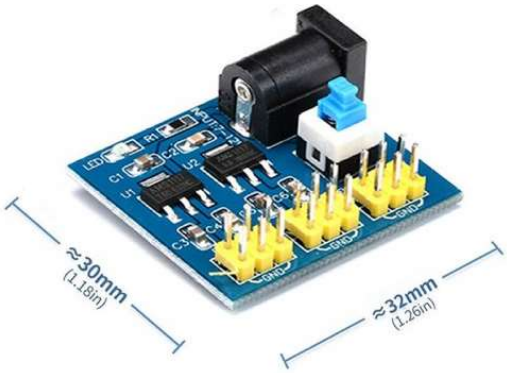

Figure 9: 5V 3.3V Voltage Module

1.2.2.10 12V Adapter

Used for the energy required for the parking system.

Figure 10: 12V Adapter

1.2.2.11 15A 400W Mosfet Switching Board

9V used to light the headlight with voltage.

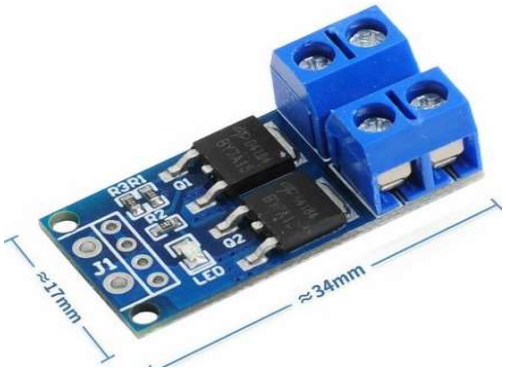

Figure 11: 15A 400W Mosfet Switching Board

2. REALIZATION OF THE PROJECT

2.1 Hardware Part

All elements used in Altium Designer application are connected as follows.

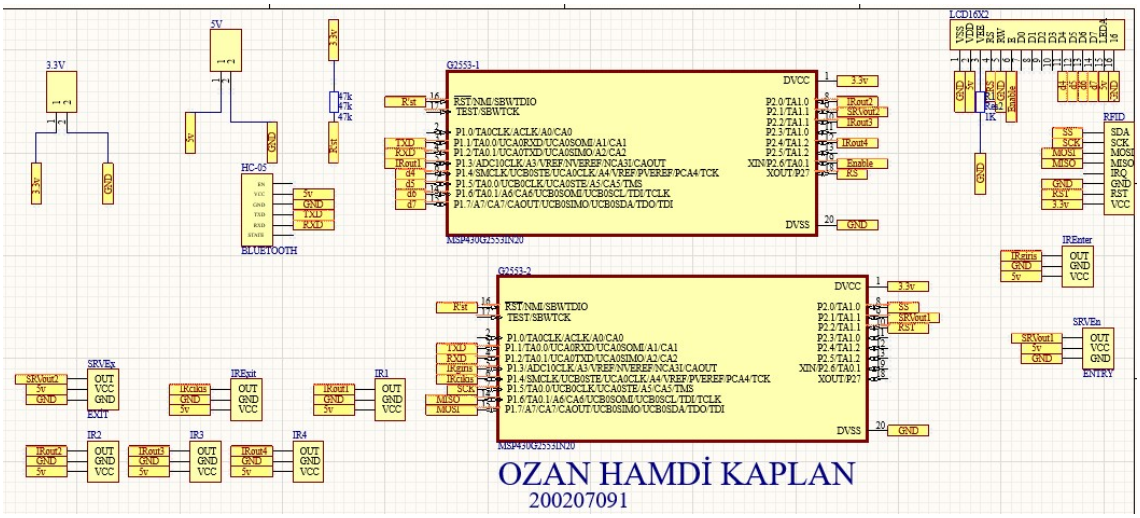

Figure 12: Circuit schematic of the parking system

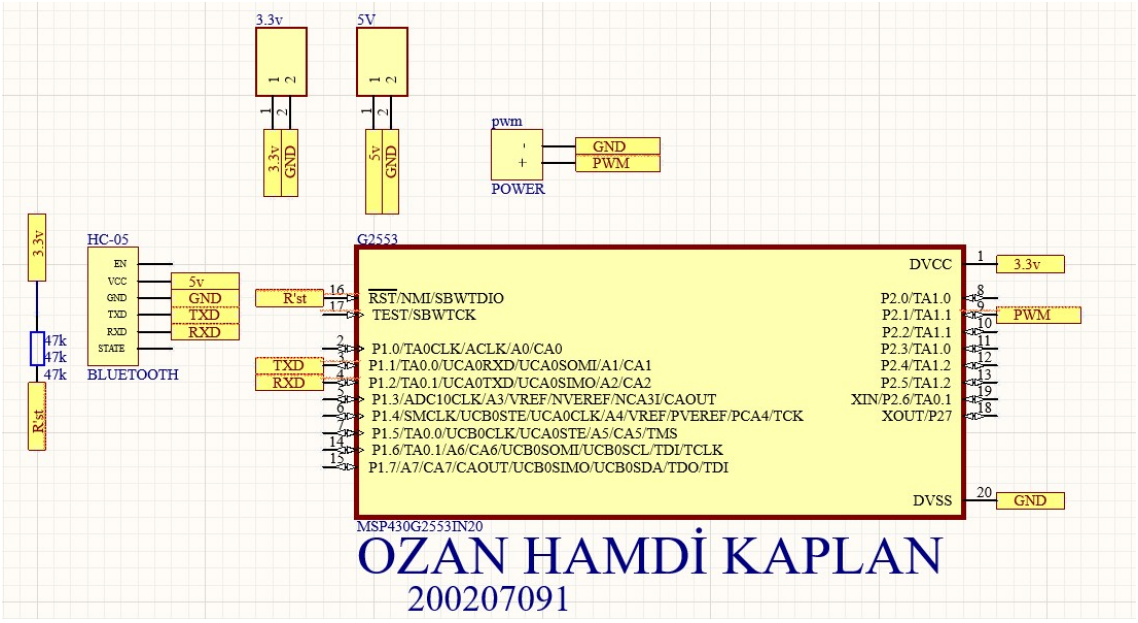

Figure 13: Circuit schematic of vehicle headlight system

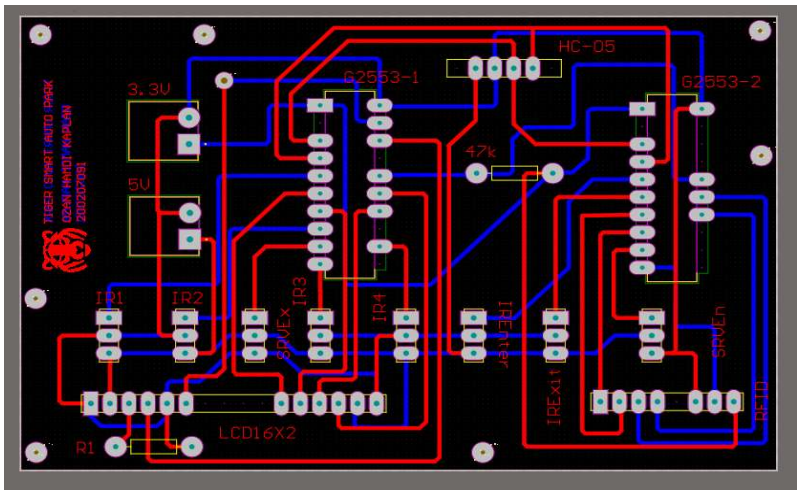

Figure 14: Pcb drawing of parking system

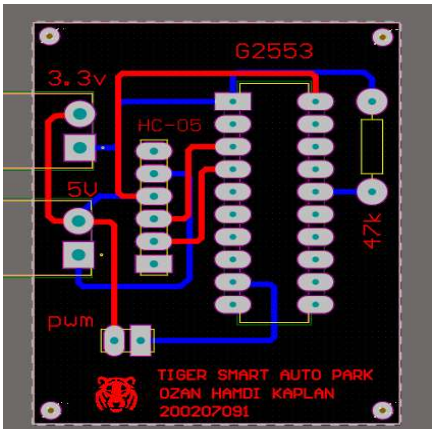

Figure 15: Pcb drawing of vehicle headlight system

.

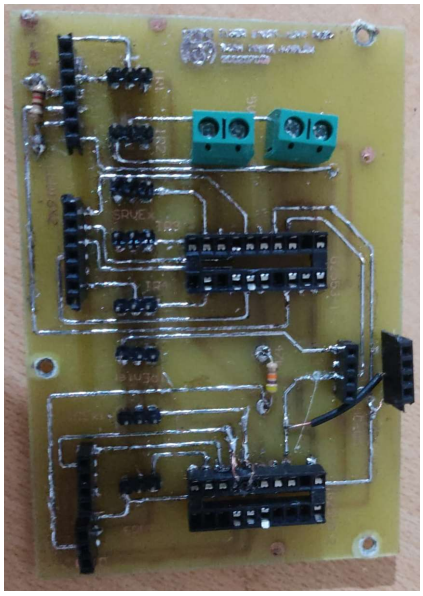

Figure 16: Circuit pressure of the parking system

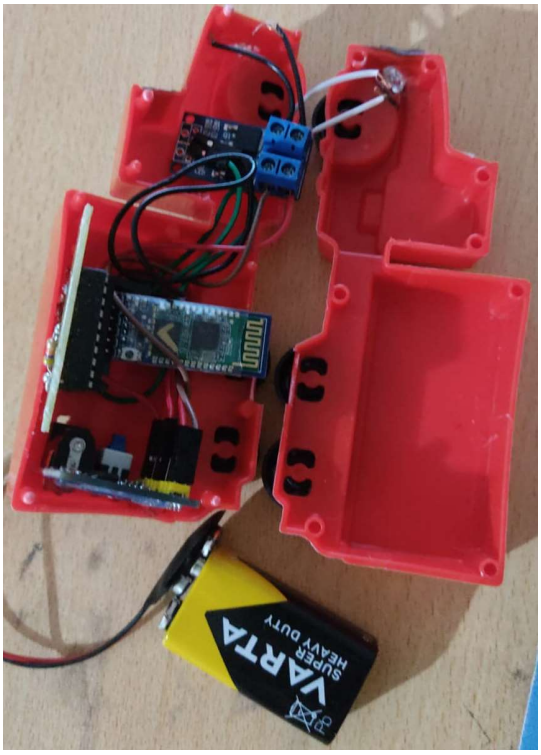

Figure 17: Modeling of vehicle headlight system

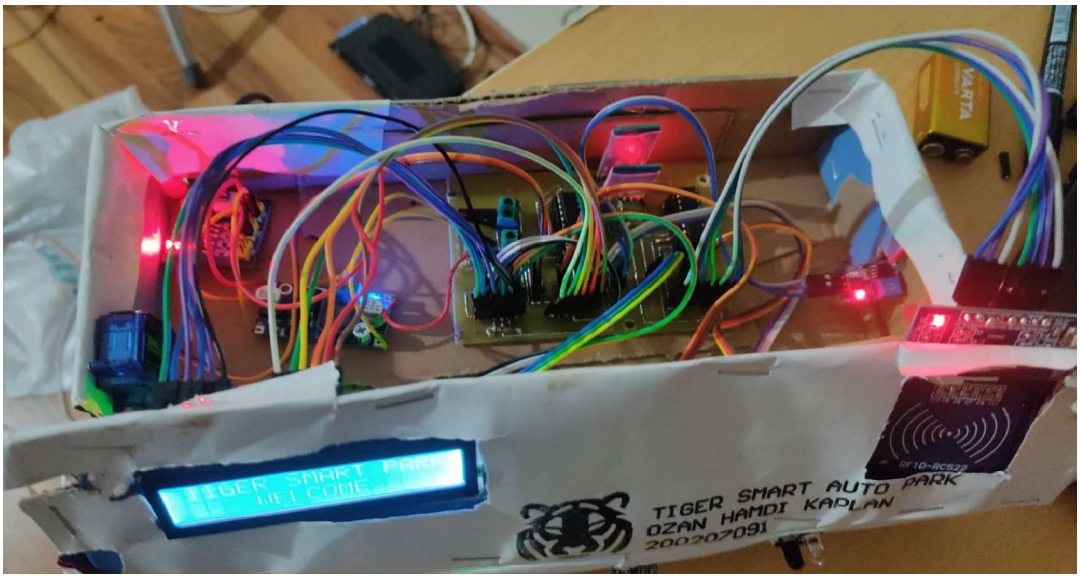

Figure 18: Modeling of the parking system

PCB drawings prepared in Altium were printed. The model has been prepared and the used elements have been placed.

2.2 Software Part

System control is provided by using the MSP430G2553 microprocessor. Data transmission to and from the computer is done by UART communication. The connection is made with the computer using the Bluetooth module HC-05. The required code written in the CCS program for the connection configuration of the HC-05 is given below.

```
P1SEL=(BIT1+BIT2);//P1.1 = RXD, P1.2=TXD
P1SEL2=(BIT1+BIT2);//P1.1 = RXD, P1.2=TXD

UCA0CTL1|=UCSSEL_2;//UART Ayarları, Saat kaynağı SMCLK//UART Settings, Clock source SMCLK
UCA0BR0=104;//1MHz 9600 bud ayarı///1MHz 9600 bud tuning
UCA0BR1=0;// 1MHz 9600 baud ayarı///1MHz 9600 bud tuning
UCA0MCTL=UCBRF_0+UCBRS_1;// UART Baud hassas ayar// UART Baud fine tuning
UCA0CTL1&=~UCSWRST;// USCI birimini hazırla// prepare the USCI volume
IE2|=UCA0RXIE;// USCI_A0 RX kesmesini aç// turn on USCI_A0 RX interrupt
```

Figure 19: Uart settings

The code written for servo motor timer settings is given below.

```
//PWM ayarları//PWM settings
TA1CCR0 = 20000;
TA1CCR1 =1600;
TA1CCTL1 = OUTMOD_7;
TA1CTL = TASSEL_2|MC_1;
```

Figure 20: Timer PWM settings

The code that allows receiving data to the computer and sending the data to the microcontroller is given below.

```
//Serial Port ayarları ve açma işlemleri yapıldı
//Serial Port settings and opening operations are done
spParkLcd.PortName = "COM13";
spRfidEntryExit.PortName = "COM3";
spCarLamp.PortName = "COM9";
try
{
    //Veri gönderip alabilmek için seri portumuzu açıyoruz
    // We open our serial port to send and receive data
    spParkLcd.Open();
    spRfidEntryExit.Open();
    spCarLamp.Open();
}
catch(Exception ex) { MessageBox.Show(ex.Message); }

//Seri port datareceived ile dinleniyor
// Listening to serial port datareceived
spParkLcd.DataReceived += new SerialDataReceivedEventHandler(SpParkLcd_DataReceived);
spRfidEntryExit.DataReceived += new SerialDataReceivedEventHandler(SpRfidEntryExit_DataReceived);
```

Figure 21: Serial Port Settings

Below is the code that allows specific users to be added and listed in the parking system.

```
//Sistem kullanıcı bilgilerii tuttuğumuz model
// Model where we keep system user information
public class Users
{
    public string UserId { get; set; }
    public string CardId { get; set; }
    public string UserName { get; set; }
    public int UserBalance { get; set; }

}
//Kullanıcıların tanımlanması
//Defining users
Users user1 = new Users() { UserId = "1", UserName = "Ozan Hamdi Kaplan", UserBalance = 50, CardId = "29" };
Users user2 = new Users() { UserId = "2", UserName = "Eray Kaplan", UserBalance = 300, CardId = "13" };
Users user3 = new Users() { UserId = "3", UserName = "Merve Şahin", UserBalance = 0 , CardId = "47" };
Users user4 = new Users() { UserId = "4", UserName = "Sibel Şafak", UserBalance = 10, CardId = "19" };
Users user5 = new Users() { UserId = "5", UserName = "Ayşe Serim", UserBalance = 30,CardId = "41" };

//Kullanıcıların saklanacağı liste oluşturuldu
// Created a list of users
List<Users> NewUsers = new List<Users>(4);
```

Figure 22: The model in which the system user information is kept

```
//Listeye kullanıcılar eklendi
// Added users to the list
NewUsers.Add(user1);
NewUsers.Add(user2);
NewUsers.Add(user3);
NewUsers.Add(user4);
NewUsers.Add(user5);

//ListView tablo ayarları yapıldı
//ListView table settings are made
lstvUser.GridLines = true;
lstvUser.FullRowSelect = true;
lstvUser.View = View.Details;

lstvUser.Columns.Add("ID");
lstvUser.Columns.Add("USER NAME");
lstvUser.Columns.Add("BALANCE");
lstvUser.Columns[0].Width = 50;
lstvUser.Columns[1].Width = 180;
lstvUser.Columns[2].Width = 110;
//Bakiye yükleme sisteminde kullanıcı seçilecek olan combobox indexlerine listelenen kullanıcıların isimleri atandı
//Names of the listed users are assigned to the combobox indexes that will be selected in the balance loading system.
foreach (Users c in NewUsers)
{
    string[] userinfo = new string[3] { c.UserId,c.UserName, c.UserBalance.ToString() };
    string[] cmb = new string[1] { c.UserName.ToString() };
    cmbUSER.Items.AddRange(cmb);
    ListViewItem lst = new ListViewItem(userinfo);
    lstvUser.Items.Add(lst);

}
```

Figure 22: Adding users to the list

The code that allows users to check balance and load transactions is given below.

```
#region Balance Check Function
public void CheckBalance(string cardID)
{
    foreach (Users c in NewUsers)
    {

        if (cardID == c.CardId.ToString())
        {
            //Yetersiz bakiye
            // Insufficient balance
            if (c.UserBalance < 10)
            {
                MessageBox.Show("Inefficient Fund Please Recharge..", "Error!!", MessageBoxButtons.OK, MessageBoxIcon.Error);
                spParkLcd.Write("d");
            }
            else
            {
                c.UserBalance -= 10;
                MessageBox.Show("Balance payment successfully completed", "İNFO", MessageBoxButtons.OK, MessageBoxIcon.Information);
                spParkLcd.Write("e");
            }
        }
    }

    lstvUser.Items.Clear();
    foreach (Users c in NewUsers)
    {
        string[] userinfo = new string[3] { c.UserId, c.UserName, c.UserBalance.ToString() };

        ListViewItem lst = new ListViewItem(userinfo);
        lstvUser.Items.Add(lst);
    }
}
#endregion
```

Figure 23: (a) User balance system

```
#region Balance Load System
private void btnLoadbalance_Click(object sender, EventArgs e)
{
        if (rdb10Balance.Checked == true)
        {
            balance = int.Parse(rdb10Balance.Text); x = 1;
        }
        else if (rdb20Balance.Checked == true)
        {
            balance = int.Parse(rdb20Balance.Text); x = 1;
        }
        else if (rdb30Balance.Checked == true)
        {
            balance = int.Parse(rdb30Balance.Text); x = 1;
        }
        else
        {
            x = 0;
        }
        foreach (Users c in NewUsers)
        {
            if (cmbUSER.SelectedItem.ToString() == c.UserName.ToString())
            {
                c.UserBalance += balance;
            }
        }
        lstvUser.Items.Clear();
        foreach (Users c in NewUsers)
        {
            string[] userinfo = new string[3] { c.UserId, c.UserName, c.UserBalance.ToString() };
            ListViewItem lst = new ListViewItem(userinfo);
            lstvUser.Items.Add(lst);
        }
        if (x == 1)
        {
            MessageBox.Show("Balance upload successfully completed", "İNFO", MessageBoxButtons.OK, MessageBoxIcon.Information);
            rdb10Balance.Checked = false;
            rdb20Balance.Checked = false;
            rdb30Balance.Checked = false;
            cmbUSER.Text = "Please Select User!!!";
        }
        else
        {
            MessageBox.Show("Please select the amount you want to top up!!!", "ERROR", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
```

Figure 24: (b) User balance system

The code written to determine the number of parking spaces is given below.

```
#region Determining The Number of Parking Spaces
private void btn4Park_Click(object sender, EventArgs e)
{
    spParkLcd.Write("A");
    btn4Park.IdleFillColor = Color.Orange;
    btn4Park.IdleForecolor = Color.Black;
    btn4Park.IdleLineColor = Color.Orange;
    btn2Park.IdleFillColor = Color.Transparent;
    btn2Park.IdleForecolor = Color.FromArgb(176, 222, 180);
    btn2Park.IdleLineColor = Color.FromArgb(176, 222, 180);
    btn3Park.IdleFillColor = Color.Transparent;
    btn3Park.IdleForecolor = Color.FromArgb(176, 222, 180);
    btn3Park.IdleLineColor = Color.FromArgb(176, 222, 180);
}

private void btn3Park_Click(object sender, EventArgs e)
{
    spParkLcd.Write("B");
    btn3Park.IdleFillColor = Color.Orange;
    btn3Park.IdleForecolor = Color.Black;
    btn3Park.IdleLineColor = Color.Orange;
    pbP1.BackColor = Color.DarkBlue;
    btn2Park.IdleFillColor = Color.Transparent;
    btn2Park.IdleForecolor = Color.FromArgb(176, 222, 180);
    btn2Park.IdleLineColor = Color.FromArgb(176, 222, 180);
    btn4Park.IdleFillColor = Color.Transparent;
    btn4Park.IdleForecolor = Color.FromArgb(176, 222, 180);
    btn4Park.IdleLineColor = Color.FromArgb(176, 222, 180);
}

private void btn2Park_Click(object sender, EventArgs e)
{
    spParkLcd.Write("C");
    btn2Park.IdleFillColor = Color.Orange;
    btn2Park.IdleForecolor = Color.Black;
    btn2Park.IdleLineColor = Color.Orange;
    pbP1.BackColor = Color.DarkBlue;
    pbP2.BackColor = Color.DarkBlue;
    btn3Park.IdleFillColor = Color.Transparent;
    btn3Park.IdleForecolor = Color.FromArgb(176, 222, 180);
    btn3Park.IdleLineColor = Color.FromArgb(176, 222, 180);
    btn4Park.IdleFillColor = Color.Transparent;
    btn4Park.IdleForecolor = Color.FromArgb(176, 222, 180);
    btn4Park.IdleLineColor = Color.FromArgb(176, 222, 180);
}
#endregion
```

Figure 25: Determining the number of parking spaces

The code that provides the barrier speed control is given below.

```
#region Barrier Control
private void btnEntryopenclose_Click(object sender, EventArgs e)
{
    if (cmbEntrySpeed.SelectedIndex == -1)
    {
        MessageBox.Show("Please select barrier speed!!!", "ERROR", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    else
    {
        spRfidEntryExit.Write(cmbEntrySpeed.SelectedIndex.ToString());
        spParkLcd.Write("x");
    }
}

private void btnExitopenclose_Click(object sender, EventArgs e)
{
    if (cmbExitSpeed.SelectedIndex == -1)
    {
        MessageBox.Show("Please select barrier speed!!!", "ERROR", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    else
    {
        spParkLcd.Write(cmbExitSpeed.SelectedIndex.ToString());
        spParkLcd.Write("x");
    }
}
#endregion
```

Figure 26: Barrier speed control

The code that provides vehicle headlight control is given below.

```
private void swCarLamp_OnValueChange_1(object sender, EventArgs e)
{
    if(swCarLamp.Value == false)
    {
        PbLampOFF.Visible = true;
        PbLampLOW.Visible = false;
        PbLampMIDDLE.Visible = false;
        PbLampHIGH.Visible = false;
        TrcBarCarLamp.Value = 0;
        spCarLamp.Write("6");
    }
    else {
    TrcBarCarLamp.Value =34;
    PbLampOFF.Visible=false;
    PbLampLOW.Visible = true;
        spCarLamp.Write("5");
    }
}

private void TrcBarCarLamp_ValueChanged(object sender, EventArgs e)
{
    if (TrcBarCarLamp.Value ==0)
    {
        swCarLamp.Value = false;
        PbLampOFF.Visible = true;
        PbLampLOW.Visible = false;
        PbLampMIDDLE.Visible = false;
        PbLampHIGH.Visible = false;
        spCarLamp.Write("6");
    }
    else if (TrcBarCarLamp.Value == 34)
    {
        swCarLamp.Value = true;
        PbLampOFF.Visible = false;
        PbLampLOW.Visible = true;
        PbLampMIDDLE.Visible = false;
        PbLampHIGH.Visible = false;
        spCarLamp.Write("5");
    }
    else if (TrcBarCarLamp.Value == 64)
    {
        PbLampOFF.Visible = false;
        PbLampLOW.Visible = false;
        PbLampMIDDLE.Visible = true;
        PbLampHIGH.Visible = false;
        spCarLamp.Write("4");
    }
    else if (TrcBarCarLamp.Value == 100)
    {
        PbLampOFF.Visible = false;
        PbLampLOW.Visible = false;
        PbLampMIDDLE.Visible = false;
        PbLampHIGH.Visible = true;
        spCarLamp.Write("3");
    }
```

Figure 27: Car headlight brightness control

The prepared form user interface is given below.

Figure 23:C# form

CONCLUSIONS AND RECOMMENDATIONS

In this study, it has been shown that current parking problems can be solved with IoT systems. The problems of not having a place in the car parks, lack of information about the place, irregular entry-exit, inability to close the parking lot (reservation system) have been taken into consideration. It has been shown that the reservation system can be formed with the determining the number of parking spaces system. It has been noticed that the location conditions can be seen remotely with c# Form, and these systems can also be applied on the web and mobile. With the user balance control system, the remaining balance information, user names were learned, and the balance was loaded.

It has been seen that many objects can be controlled remotely with barrier speed and vehicle headlight control systems. Connection is provided with 3 microcontrollers using the serial port. It has been seen that the number of these connections can be increased and more than one bluetooth can be connected to a computer.

REFERENCES

1)https://onlineplcegitimi.com/haberlesme-protokolleri-bluetouth/#:~:text=Bluetooth%20bilgisayar%2C%20%C3%A7evre%20birimleri%20ve,ses%20ve%20veri%20iletimi%20yapabilmektedir.

2)https://www.evde360.com/page/nesnelerin-interneti-ve-akilli-evler

3)https://avesis.gazi.edu.tr/resume/lessonmaterieldownload/rkasap?key=7ad1d128-99d2-44ad-b168-2e525c7ef11b

4)https://www.technopat.net/2022/08/07/microsoft-net-framework-nedir/

5)https://tr.wikipedia.org/wiki/.NET_Framework

6)https://www.mcu-turkey.com/wp-content/uploads/2011/03/msp430-programlama-notlari-uygulamalar-bilgiler.pdf

7)https://nevonprojects.com/iot-smart-parking-using-rfid-with-android-app/