**BOLU ABANT İZZET BAYSAL ÜNİVERSİTESİ**

**BOLU TEKNİK BİLİMLER MESLEK YÜKSEKOKULU**

**BİLGİSAYAR TEKNOLOJİLERİ BÖLÜMÜ**

**VISUAL STUDIO ile YAZILIM**

**PROJE RAPORU**

**Zeynel ÇİFTSÜREN**

**183301018**

**Öğretim Görevlisi Nermin OKUR**

Eylül 2020

İÇİNDEKİLER

1.ÖzetIII

2.GirişV

2.1 Konunun Problemi V

2.2 Park Otomasyonu V

3.Kurumsal Bilgiler VIII

4.Sonuç IX

5.KaynaklarX

6.ÖzgeçmişXI

7.EklerXII

VISUAL STUDIO ile YAZILIM

(1.ÖZET)

Yazılım, elektronik parçaların birbiri ile iletişime geçerek amaçlanan işi yapmasını sağlayan kodlar bütünlüğüdür. Yazılımın ilk çıktığı zamanlarda insanlar bunu sadece karmaşık matematiksel problemleri çözmek için kullansa da günümüzde hayatımızın en önemli konumuna gelmiş bulunmakta. Günlük hayatımızda yazılımın girmediği tek bir anımız bile bulunmamakta; kullandığımız toplu taşıma araçlarında, iletişime geçmemizi sağlayan akılı telefonlarda hatta en basitinden bir kahve makinesinde bile bir yazılım mevcuttur.  
  
 Visual Studio, Microsoft tarafından geliştirilen tümleşik geliştirme ortamıdır. Genellikle Windows işletim sistemi tabanlı bilgisayarlar için bir uygulama geliştirecekseniz akla gelen ilk programdır. Basit bir arayüze sahip olması ile beraber kolay kullanımı ve tabiki de birçok nedenle beraber bu programı tercih edilmesine sebep olmakta.

**Anahtar kelimeler:**

Yazılım, Visual Studio

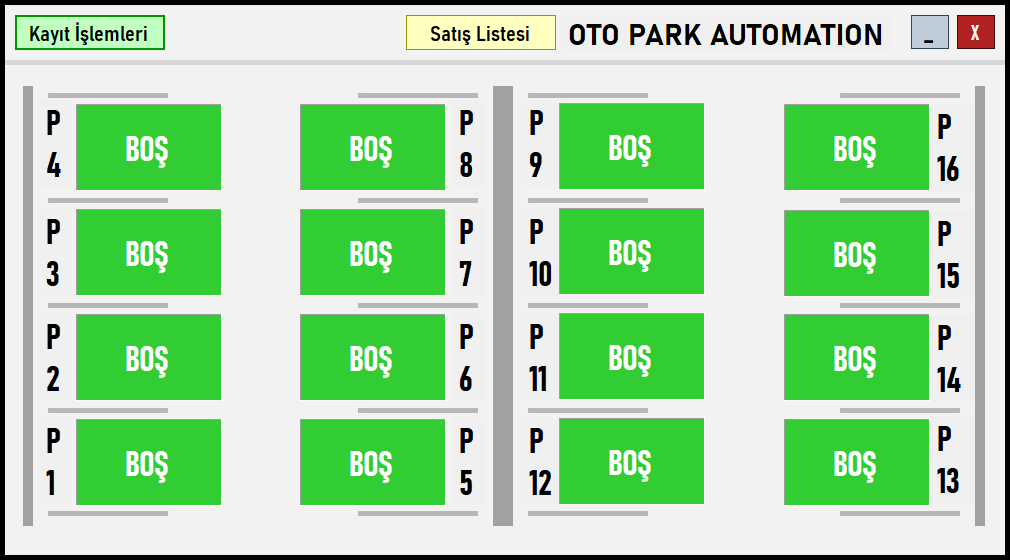
2. GİRİŞ

2.1. Konunun Problemi

Günümüzde kişisel araç sahibi olan insan sayısını artıkça araçlar için park alanlarına ihtiyaç aynı şekilde artmaktadır. Otopark giriş ve çıkış sırasında gerçekleşen işlemler bazen müşteriyi bekletmesine sebep olmak ile beraber iş için vakit kaybı olabiliyor.

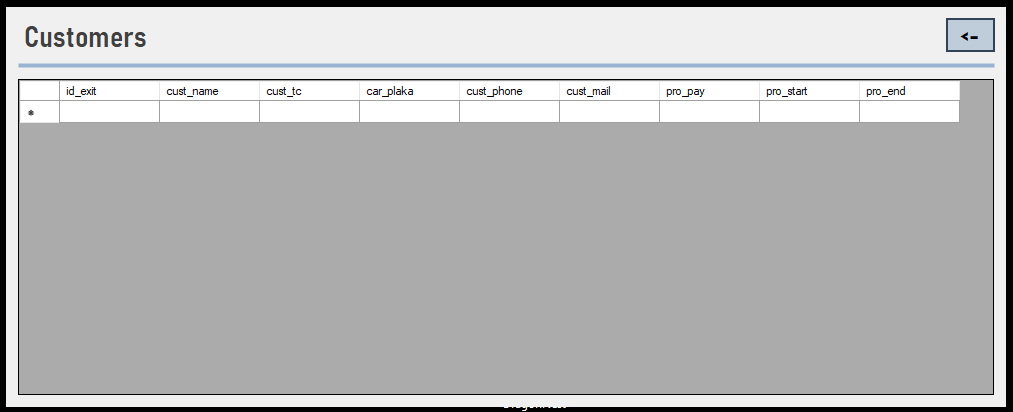
2.2. Park Otomasyonu

Park giriş ve çıkışlarda yaşanan zaman kaybı için Visual Studio da tasarlanmış bir otomasyon işimizi görür. Programda boş park alanları, dolu park alanlarındaki araç bilgileri, araç kaydı ve program kullanıldığından beridir gelen araç bilgilerini Access veri tabanı sayesinde görüntüleyebilmekteyiz.  
  
 Programın genel olarak arayüzü şekil 2.2.1 de gözüktüğü gibidir. Arayüzde toplam 33 Label 4 button yer almaktadır.



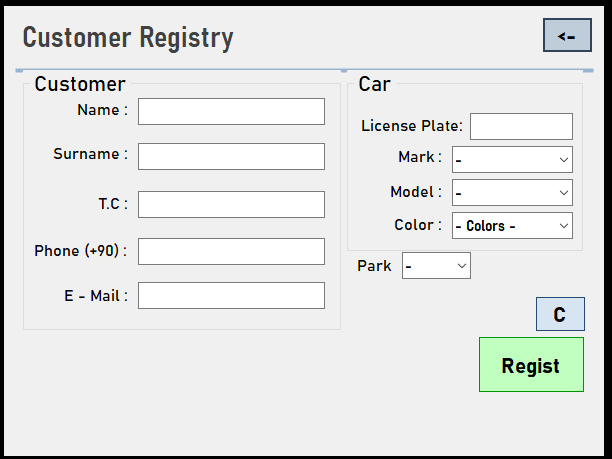
Şekil 2.2.1 Arayüz

Satış Listesi butonuna tıklandığında karşımıza şekil 2.2.2’de gözüktüğü gibi 1 label 1 button ve 1 datagridview dan oluşan basit bir arayüz bizi karşılamakta. Burada genel olarak tüm zaman boyunca otoparka gelen müşterilerin bilgileri yer almaktadır.



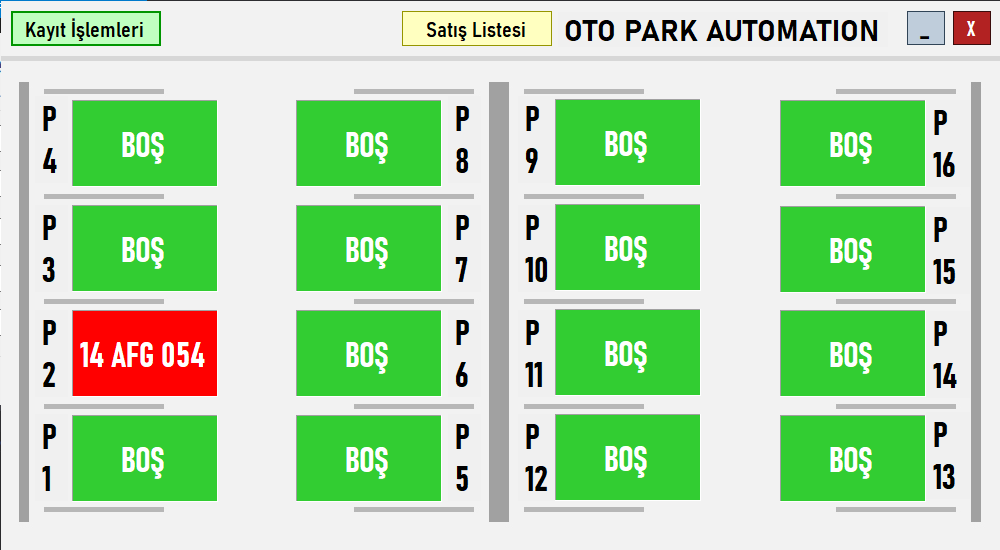
Şekil 2.2.2 Satış Listeleme

Kayıt İşlemleri butonuna tıkladığında karşımıza şekil 2.2.3’te gözüktüğü gibi bir arayüz çıkmakta. Burada otoparka giriş yapacak aracın bilgileri girilip aracın park edeceği park alanı seçip işlem yapmakta. Toplam 11 Label, 6 TextBox, 4 ComboBox, 2 GroupBox ve 3 Button kullanılmıştır.



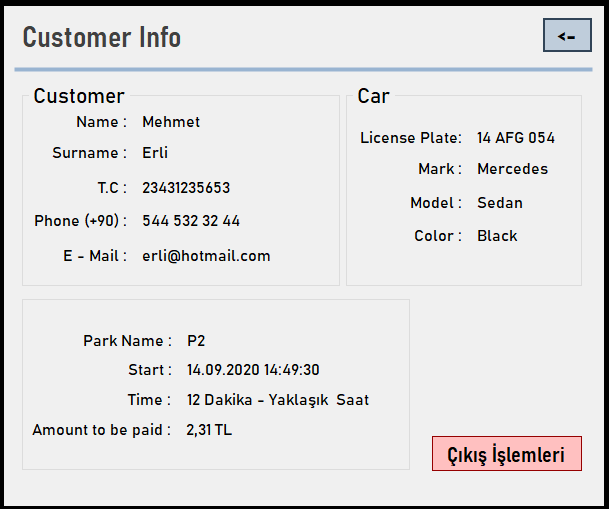
Şekil 2.2.3 Kayıt İşlemleri

Otoparka bir kayıt yapıldığı zaman şekil 2.2.4’te gözüktüğü gibi olmakta. Görülen kırmızı alana tıklanırsa o park halindeki araç ile ilgili kişisel bilgiler ekranımıza şekil 2.2.5’te gözüktüğü gibi gelmekte burada bizi aracın ne zaman başladığı ve ne kadar ücret ödemesi gerektiği ile ilgili bilgi vermekte.



Şekil 2.2.4 Otomasyon

Müşteri bilgi kısmında 27 Label, 3 groupBox, 2 button kullanılmıştır



Şekil 2.2.5 Müşteri Bilgi

Çıkış İşlemleri butonuna tıkladığında aracın çıkış işlemleri gerçekleşir ve veri tabanına şekil 2.2.2’de görüntülene bilmesi için kayıt etmekte.

1. KURAMSAL BİLGİLER

Microsoft Visual Studio, Microsoft tarafından geliştirilen bir tümleşik geliştirme ortamıdır (IDE). [Microsoft Windows](https://tr.wikipedia.org/wiki/Microsoft_Windows), [Windows Mobile](https://tr.wikipedia.org/wiki/Windows_Mobile), [Windows CE](https://tr.wikipedia.org/wiki/Windows_CE), [.NET Framework](https://tr.wikipedia.org/wiki/.NET_Framework), [.NET Compact Framework](https://tr.wikipedia.org/wiki/.NET_Compact_Framework) ve [Microsoft Silverlight](https://tr.wikipedia.org/wiki/Microsoft_Silverlight) tarafından desteklenen tüm platformlar için yönetilen kod ile birlikte yerel kod ve Windows Forms uygulamaları, web siteleri, web uygulamaları ve web servisleri ile birlikte konsol ve grafiksel kullanıcı arayüzü uygulamaları geliştirmek için kullanılır.

Visual Studio [IntelliSense](https://tr.wikipedia.org/w/index.php?title=IntelliSense&action=edit&redlink=1" \o "IntelliSense (sayfa mevcut değil))'in yanı sıra "code refactoring" destekleyen bir kod editörü içerir. Entegre hata ayıklayıcı, hem kaynak-seviyesinde hem de makine-seviyesinde çalışır. Diğer yerleşik araçlar, [GUI](https://tr.wikipedia.org/wiki/GUI" \o "GUI) uygulamaları, web tasarımcısı, sınıf tasarımcısı ve [veritabanı](https://tr.wikipedia.org/wiki/Veritaban%C4%B1" \o "Veritabanı) şema tasarımcısı yaratabilmek için bir form tasarımcısı içerir. Hemen hemen her işlevsellik düzeyinde dahil olmak üzere, kaynak kontrol sistemleri için destek (Subversion ve Visual SourceSafe gibi) sunan eklentileri kabul eder.[1]

Visual Studio, değişik programlama dillerini destekler, bu da kod editörü ve hata ayıklayıcısının neredeyse tüm programlama dillerini desteklemesini sağlamaktadır. Dahili diller [C](https://tr.wikipedia.org/wiki/C_(programlama_dili))/[C++](https://tr.wikipedia.org/wiki/C%2B%2B) (Görsel yoluyla C++), [VB.NET](https://tr.wikipedia.org/wiki/Visual_Basic_.NET) (Visual Basic .NET üzerinden), [C#](https://tr.wikipedia.org/wiki/C_Sharp) (Visual C# ile), ve [F#](https://tr.wikipedia.org/wiki/F_Sharp) (Visual Studio 2010 itibarıyla) içermektedir.[2]

C#’tan bahsetmem gerekir ise, C#; Microsoft tarafından .NET Teknolojisi için geliştirilen modern bir programlama dilidir. Sözdizimi C-like bir deneyim sunar. Microsoft tarafından geliştirilmiş olsa da ECMA ve ISO standartları altına alınmıştır. C programlama dilinde bir tam sayı değişkeni 1 artırmak için değişkenden sonra "++" eki kullanılır.[2]

1. SONUÇ

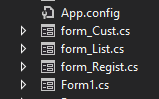
Artık park görevlisi bir kâğıt üzerinde giriş çıkış bilgilerini tutmak zorunda değil, işlemler daha kısa sürecek, işlem sonrası ücretlendirmeler daha keskin sonuçlar olacak, geçmiş park bilgileri giriş çıkışları kaydı veri tabanı üzerinden olacak, hangi park alanları boş veya dolu olduğu daha iyi görülecek ve park alanlarında bulunan müşteriler hakkında bilgi daha verimli bir şekilde gözüke bilecek.

1. KAYNAKLAR
2. Wikipedia, “Microsoft Visual Studio”, 14.09.2020, <https://tr.wikipedia.org/wiki/Microsoft_Visual_Studio>
3. Wikipedia, “C Sharp”, 2.07.20202”, <https://tr.wikipedia.org/wiki/C_Sharp>
4. **ÖZGEÇMİŞ**

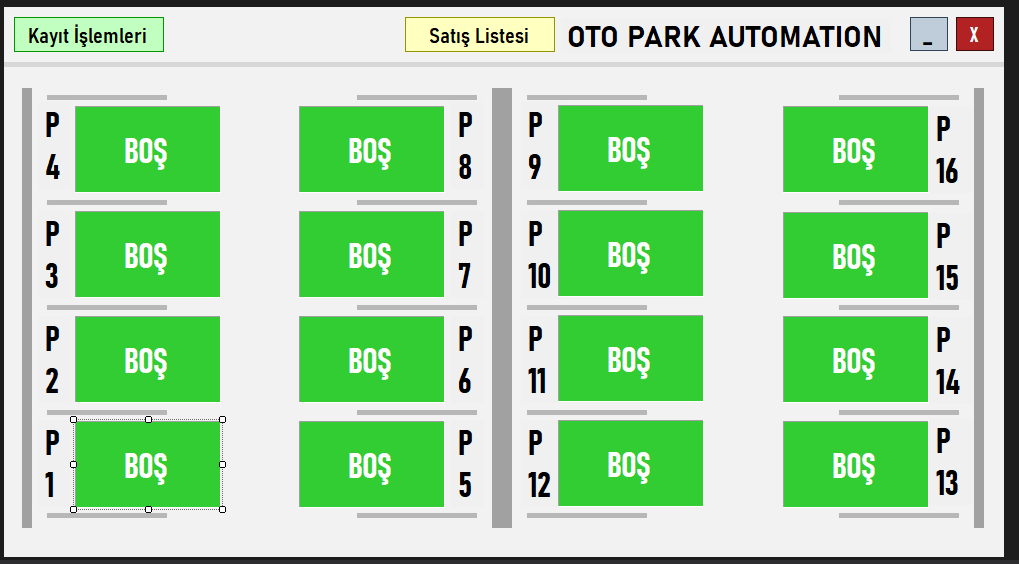
2000 yılında Ukrayna’nın Nikolaev şehrinde doğdu. 2011 de Türkiye’ye geldi, Abulkadir Tutaşı İlk Öğretim Okulu’nda orta öğretime başladı. TEOG sınavına girdikten sonra 227 puan ile Artuklu Mesleki ve Teknik Anadolu Lisesi’ne yerleşti. Bir sene bu okulda okuduktan okulunu değiştirerek Mardin Mesleki ve Teknik Anadolu Lisesi’ne geçiş yaptı ve Teknik Programında diploma notu 81,35 ile buradan mezun oldu. YKS sınavından TYT türünden 267 puan çıkararak Bolu Abant İzzet Baysal Üniversitesi’nde Bilişim Teknolojileri dalında Bilgisayar Programcılı bölümününe geçti ve hala okuyor.

1. **EKLER**

Programımızda toplam 4 form bulunmakta. Form1 bizim ana formumuzdur.



**Form1.cs**



using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

namespace otoparkOtomasyonProje

{

public partial class form\_Ana : Form

{

public form\_Ana()

{

InitializeComponent();

}

public static string custInfo;

//veri tabanı değişkenleri

OleDbConnection connect = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=otoPark.mdb;");

OleDbCommand command = new OleDbCommand();

OleDbDataAdapter adap = new OleDbDataAdapter();

OleDbDataReader dr;

DataSet dataSet = new DataSet();

string com = "";

//veri tabanına bağlanır

public void conOpen()

{

try

{

connect.Close();

connect.Open();

}

catch (Exception a)

{

MessageBox.Show("Bağlantı açılamadı");

MessageBox.Show(a.ToString());

}

}

//tabloyu getirir

public void veriDoldur(string tablo)

{

conOpen();

command.CommandText = "SELECT \* FROM " + tablo;

command.Connection = connect;

dr = command.ExecuteReader();

labelClear();

while (dr.Read())

{

foreach (Control item in Controls)

{

if (item is Label)

{

string parkYeri = "label\_" + dr["park\_name"].ToString();

if (item.Name == parkYeri)

{

item.Text = dr["car\_plaka"].ToString();

item.BackColor = Color.Red;

}

}

}

}

command.Dispose();

dr.Dispose();

connect.Close();

}

//label sıfırlar

public void labelClear()

{

int i = 1;

string labelPark = "label\_P" + i.ToString();

foreach (Control item in Controls)

{

if (item is Label)

{

if (item.Name == labelPark)

{

item.Text = "BOŞ";

item.BackColor = Color.LimeGreen;

}

}

i++;

}

}

private void form\_Ana\_Load(object sender, EventArgs e)

{

veriDoldur("proccessing");

}

private void button\_Close\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void button\_Min\_Click(object sender, EventArgs e)

{

this.WindowState = FormWindowState.Minimized;

}

private void button\_Registry\_Click(object sender, EventArgs e)

{

form\_Regist formRegist = new form\_Regist();

formRegist.Show();

this.Hide();

}

private void button\_List\_Click(object sender, EventArgs e)

{

form\_List formList = new form\_List();

formList.Show();

this.Hide();

}

form\_Cust formCust = new form\_Cust();

// Park label ları tek tek kontrol edilip eğer içerisinde bir araç park edilmişse label click ile custInfo formu açılır

private void label\_P1\_Click(object sender, EventArgs e)

{

if (label\_P1.Text != "BOŞ")

{

connect.Close();

custInfo = label\_P1.Text;

formCust.Show();

this.Hide();

}

}

private void label\_P2\_Click(object sender, EventArgs e)

{

if (label\_P2.Text != "BOŞ")

{

connect.Close();

custInfo = label\_P2.Text;

formCust.Show();

this.Hide();

}

}

private void label\_P3\_Click(object sender, EventArgs e)

{

if (label\_P3.Text != "BOŞ")

{

connect.Close();

custInfo = label\_P3.Text;

formCust.Show();

this.Hide();

}

}

private void label\_P4\_Click(object sender, EventArgs e)

{

if (label\_P4.Text != "BOŞ")

{

connect.Close();

custInfo = label\_P4.Text;

formCust.Show();

this.Hide();

}

}

private void label\_P5\_Click(object sender, EventArgs e)

{

if (label\_P5.Text != "BOŞ")

{

connect.Close();

custInfo = label\_P5.Text;

formCust.Show();

this.Hide();

}

}

private void label\_P6\_Click(object sender, EventArgs e)

{

if (label\_P6.Text != "BOŞ")

{

connect.Close();

custInfo = label\_P6.Text;

formCust.Show();

this.Hide();

}

}

private void label\_P7\_Click(object sender, EventArgs e)

{

if (label\_P7.Text != "BOŞ")

{

connect.Close();

custInfo = label\_P7.Text;

formCust.Show();

this.Hide();

}

}

private void label\_P8\_Click(object sender, EventArgs e)

{

if (label\_P8.Text != "BOŞ")

{

connect.Close();

custInfo = label\_P8.Text;

formCust.Show();

this.Hide();

}

}

private void label\_P12\_Click(object sender, EventArgs e)

{

if (label\_P12.Text != "BOŞ")

{

connect.Close();

custInfo = label\_P12.Text;

formCust.Show();

this.Hide();

}

}

private void label\_P11\_Click(object sender, EventArgs e)

{

if (label\_P11.Text != "BOŞ")

{

connect.Close();

custInfo = label\_P11.Text;

formCust.Show();

this.Hide();

}

}

private void label\_P10\_Click(object sender, EventArgs e)

{

if (label\_P10.Text != "BOŞ")

{

connect.Close();

custInfo = label\_P10.Text;

formCust.Show();

this.Hide();

}

}

private void label\_P9\_Click(object sender, EventArgs e)

{

if (label\_P9.Text != "BOŞ")

{

connect.Close();

custInfo = label\_P9.Text;

formCust.Show();

this.Hide();

}

}

private void label\_P13\_Click(object sender, EventArgs e)

{

if (label\_P13.Text != "BOŞ")

{

connect.Close();

custInfo = label\_P13.Text;

formCust.Show();

this.Hide();

}

}

private void label\_P14\_Click(object sender, EventArgs e)

{

if (label\_P14.Text != "BOŞ")

{

connect.Close();

custInfo = label\_P14.Text;

formCust.Show();

this.Hide();

}

}

private void label\_P15\_Click(object sender, EventArgs e)

{

if (label\_P15.Text != "BOŞ")

{

connect.Close();

custInfo = label\_P15.Text;

formCust.Show();

this.Hide();

}

}

private void label\_P16\_Click(object sender, EventArgs e)

{

if (label\_P16.Text != "BOŞ")

{

connect.Close();

custInfo = label\_P16.Text;

formCust.Show();

this.Hide();

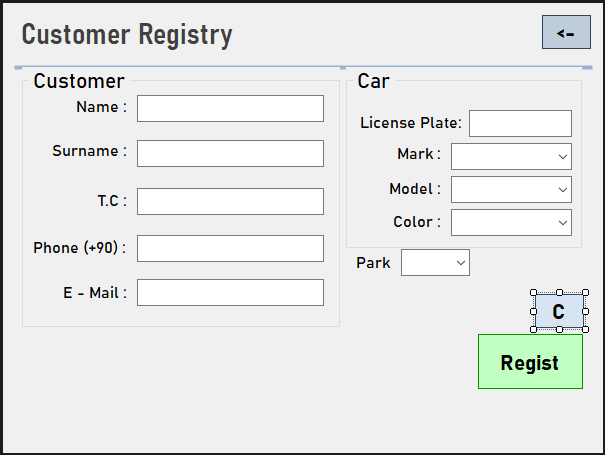
}

}

}

}

**Form\_Regist.cs**



using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

namespace otoparkOtomasyonProje

{

public partial class form\_Regist : Form

{

public form\_Regist()

{

InitializeComponent();

}

string[] parklar = new string[16];

//veri tabanı değişkenleri

public OleDbConnection connect = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=otoPark.mdb;");

public OleDbCommand command = new OleDbCommand();

public OleDbDataAdapter adap = new OleDbDataAdapter();

public OleDbDataReader dr;

public DataSet dataSet = new DataSet();

public string com = "";

//veri tabanına bağlanır

public void conOpen()

{

try

{

connect.Close();

connect.Open();

}

catch (Exception a)

{

MessageBox.Show("Bağlantı açılamadı");

MessageBox.Show(a.ToString());

}

}

//Bağlantıyı kapatıp komut değişkenlerini dispose ediyoruz

public void conClose()

{

this.dr.Dispose();

this.command.Dispose();

this.com = "";

this.connect.Close();

}

//Form açılışta bir sorun yaşanmaması ve clear butonuna tıkladığında formun temizlenmesi için.

public void clearData()

{

textBox1.Clear();

textBox2.Clear();

textBox3.Clear();

textBox4.Clear();

textBox5.Clear();

textBox6.Clear();

comboBox1.SelectedIndex = 0;

comboBox2.Items.Clear();

comboBox2.Text = "";

comboBox2.Items.Add("-");

comboBox2.SelectedIndex = 0;

comboBox3.SelectedIndex = 0;

comboBox4.SelectedIndex = 0;

}

//Veri tabanındaki proccessing tablosundan veriler kontrol edilip boş olan park alanlarını combobx’a yazılır.

public void doldurPark()

{

comboBox4.Items.Clear();

comboBox4.Items.Add("-");

comboBox4.SelectedIndex = 0;

int j = 0;

bool varMi = false;

for (int i = 0; i < parklar.Length; i++)

{

parklar[i] = " ";

}

string parkName = "P";

com = "SELECT park\_name FROM proccessing";

conOpen();

command.CommandText = com;

command.Connection = connect;

dr = command.ExecuteReader();

while (dr.Read())

{

parklar[j] = dr["park\_name"].ToString();

j++;

}

for (int i = 1; i <= 16; i++)

{

parkName += i.ToString();

foreach (string item in parklar)

{

if (parkName == item)

{

varMi = true;

}

}

if (varMi != true)

{

comboBox4.Items.Add(parkName);

}

varMi = false;

parkName = "P";

}

conClose();

}

//formun load kod satırı clearData() ve doldurPark() metodları çalıştırılır.

private void form\_Regist\_Load(object sender, EventArgs e)

{

clearData();

doldurPark();

}

//Clear butonu na tıkladığında clearData() metodu çalıştırılıp form temizlenir.

private void button\_Clear\_Click(object sender, EventArgs e)

{

clearData();

}

//formdan ana forma geçiş işlemi gerçekleşir

private void button\_Min\_Click(object sender, EventArgs e)

{

clearData();

form\_Ana formAna = new form\_Ana();

formAna.Show();

this.Hide();

}

//Araç markası seçildiğinde o araç markası ile ilgili araç modeleri ekrana gelmesi için if lerle kontrol sağlanır

private void comboBox1\_SelectedIndexChanged\_1(object sender, EventArgs e)

{

if (comboBox1.SelectedIndex == 1)

{

comboBox2.Items.Clear();

comboBox2.Items.Add("-");

comboBox2.Items.Add("Passat");

comboBox2.Items.Add("Golf");

comboBox2.Items.Add("Polo");

comboBox2.Items.Add("Arteon");

comboBox2.Text = "";

comboBox2.SelectedIndex = 0;

}

else if (comboBox1.SelectedIndex == 2)

{

comboBox2.Items.Clear();

comboBox2.Items.Add("-");

comboBox2.Items.Add("Corsa");

comboBox2.Items.Add("Astra");

comboBox2.Items.Add("Grandland");

comboBox2.Items.Add("Corssland");

comboBox2.Text = "";

comboBox2.SelectedIndex = 0;

}

else if (comboBox1.SelectedIndex == 3)

{

comboBox2.Items.Clear();

comboBox2.Items.Add("-");

comboBox2.Items.Add("Egea");

comboBox2.Items.Add("Linea");

comboBox2.Items.Add("Punto");

comboBox2.Items.Add("Panda");

comboBox2.Text = "";

comboBox2.SelectedIndex = 0;

}

else if (comboBox1.SelectedIndex == 4)

{

comboBox2.Items.Clear();

comboBox2.Items.Add("-");

comboBox2.Items.Add("Mercedes-Benz");

comboBox2.Items.Add("Sedan");

comboBox2.Items.Add("Estate");

comboBox2.Items.Add("Coupé");

comboBox2.Text = "";

comboBox2.SelectedIndex = 0;

}

}

//Regist butonuna tıklandığında kayıt işlemi gerçekleşir burada Bilgiler parametrews olarak kayıt edilip SQL INSERT komutu ile bu parametreler proccesing, customers, cars tablolarına ayrı ayrı kayıt işlemi gerçekleştirir.

private void button\_Regist\_Click(object sender, EventArgs e)

{

DialogResult Secim = new DialogResult();

Secim = MessageBox.Show("Kayıt işlemi gerçekleşsin mi", "REGIST", MessageBoxButtons.YesNo, MessageBoxIcon.Question, MessageBoxDefaultButton.Button2);

if (Secim == DialogResult.Yes)

{

com = "INSERT INTO customers(cost\_tc,cost\_name,cost\_surname,cost\_phone,cost\_mail) VALUES(@cost\_tc,@cost\_name,@cost\_surname,@cost\_phone,@cost\_mail)";

conOpen();

command.Connection = connect;

command.CommandText = com;

command.Parameters.AddWithValue("@cost\_tc", textBox3.Text);

command.Parameters.AddWithValue("@cost\_name", textBox1.Text);

command.Parameters.AddWithValue("@cost\_surname", textBox2.Text);

command.Parameters.AddWithValue("@cost\_phone", textBox4.Text);

command.Parameters.AddWithValue("@cost\_mail", textBox5.Text);

command.ExecuteNonQuery();

conClose();

com = "INSERT INTO cars(car\_plaka,car\_marka,car\_model,car\_renk) VALUES(@car\_plaka,@car\_marka,@car\_model,@car\_renk)";

conOpen();

command.Connection = connect;

command.CommandText = com;

command.Parameters.AddWithValue("@car\_plaka", textBox6.Text);

command.Parameters.AddWithValue("@car\_marka", comboBox1.Text);

command.Parameters.AddWithValue("@car\_model", comboBox2.Text);

command.Parameters.AddWithValue("@car\_renk", comboBox3.Text);

command.ExecuteNonQuery();

conClose();

com = "INSERT INTO proccessing(cust\_tc,car\_plaka,park\_name,pro\_start) VALUES(@cust\_tc,@car\_plaka,@park\_name,@pro\_start)";

conOpen();

command.Connection = connect;

command.CommandText = com;

command.Parameters.AddWithValue("@cust\_tc", textBox3.Text);

command.Parameters.AddWithValue("@car\_plaka", textBox6.Text);

command.Parameters.AddWithValue("@park\_name", comboBox4.Text);

command.Parameters.AddWithValue("@pro\_start", DateTime.Now.ToString());

command.ExecuteNonQuery();

conClose();

MessageBox.Show("Kayıt Tamalanmıştır");

clearData();

doldurPark();

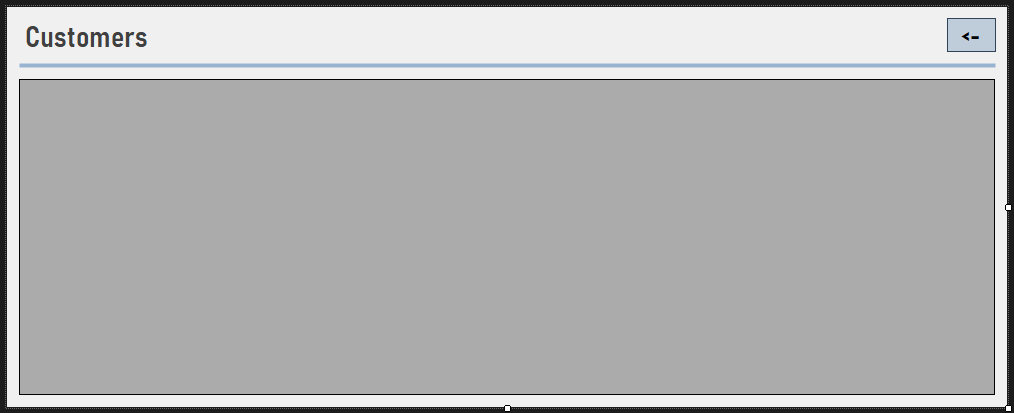
}

}

}

}

**Form\_List.cs**



using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

namespace otoparkOtomasyonProje

{

public partial class form\_List : Form

{

public form\_List()

{

InitializeComponent();

}

//veri tabanına bağlanmamız için Ole nesnemizdeki değişkenlerimizi hazırlıyoruz

public OleDbConnection connect = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=otoPark.mdb;");

public OleDbDataAdapter adap;

public DataSet dataSet = new DataSet();

public string com = "";

//bağlnatı açılır ve bir sorun olmadığı kontrol edilir

public void conOpen()

{

try

{

connect.Close();

connect.Open();

}

catch (Exception a)

{

MessageBox.Show("Bağlantı açılamadı");

MessageBox.Show(a.ToString());

}

}

//min buttouna tıklandığında ana forma geri dönülür

private void button\_Min\_Click(object sender, EventArgs e)

{

form\_Ana formAna = new form\_Ana();

formAna.Show();

this.Hide();

}

//Form load işleminde SQL SELECT komutu ile exitCust tablosundaki tüm veriler adap a kayıt edilip dataSet’e fill işlemi gerçekleşir böylece dataGridView a bunu yansıtıp kullanıclar kolaylıkla göre bilmesi sağlanır

private void form\_List\_Load(object sender, EventArgs e)

{

com = "SELECT \* FROM exitCust";

conOpen();

adap = new OleDbDataAdapter(com, connect);

adap.Fill(dataSet,"hello");

adap.Dispose();

connect.Close();

dataGridView1.DataSource = dataSet;

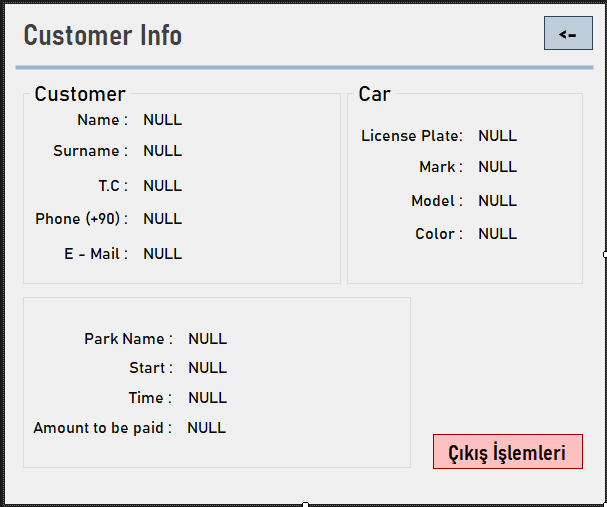
dataGridView1.DataMember = "hello";

}

}

}

**form\_Cust.cs**



using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

namespace otoparkOtomasyonProje

{

public partial class form\_Cust : Form

{

public form\_Cust()

{

InitializeComponent();

}

//veri tabanından bazı bilgilere ihtiyacımız var bunlar çıkış işlemleri için gerekli olan değişkenlerimizdir

public int id\_car=0;

public int id\_cust = 0;

public int id\_pro = 0;

public double custPay = 0;

public string tc = "";

public string plaka = "";

public class dataBase

{

//veri tabanı değişkenleri

public OleDbConnection connect = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data Source=otoPark.mdb;");

public OleDbCommand command = new OleDbCommand();

public OleDbDataAdapter adap = new OleDbDataAdapter();

public OleDbDataReader dr;

public DataSet dataSet = new DataSet();

public string com = "";

//veri tabanına bağlanır

public void conOpen()

{

try

{

connect.Close();

connect.Open();

}

catch (Exception a)

{

MessageBox.Show("Bağlantı açılamadı");

MessageBox.Show(a.ToString());

}

}

// istenilen tablodan veri getirilir

public void veriGetir(string tabloName)

{

com = "SELECT \* FROM " + tabloName + " WHERE car\_plaka=@park";

conOpen();

command.CommandText = com;

command.Connection = connect;

command.Parameters.AddWithValue("@park", form\_Ana.custInfo);

this.dr = command.ExecuteReader();

}

//girilen tc parametresine göre istenilen Müşterinin verileri getirilir

public void veriGetirCust(string tc1)

{

com = "SELECT \* FROM customers WHERE cost\_tc=@tc";

conOpen();

command.CommandText = com;

command.Connection = connect;

command.Parameters.AddWithValue("@tc", tc1);

this.dr = command.ExecuteReader();

}

//çıkış işlemi yapılırken otomatik olarak programın kayıt ettiği veriler ile beraber SQL in INSERT komutu ile bu parametredeki veriler alınıp exitCust tablosuna kayıt edilir

public void exitRegist(string ad,string tc, string tel , string mail , string plaka, string pay, DateTime start, DateTime exit)

{

com = "INSERT INTO exitCust(cust\_name, cust\_tc, car\_plaka , cust\_phone, cust\_mail, pro\_pay, pro\_start, pro\_end) VALUES(@cust\_name, @cust\_tc, @car\_plaka , @cust\_phone, @cust\_mail, @pro\_pay, @pro\_start, @pro\_end)";

conOpen();

command.CommandText = com;

command.Connection = connect;

command.Parameters.AddWithValue("@cust\_name", ad);

command.Parameters.AddWithValue("@cust\_tc", tc);

command.Parameters.AddWithValue("@car\_plaka", plaka);

command.Parameters.AddWithValue("@cust\_phone", tel);

command.Parameters.AddWithValue("@cust\_mail", mail);

command.Parameters.AddWithValue("@pro\_pay", pay);

command.Parameters.Add("@pro\_start", OleDbType.Date).Value = start;

command.Parameters.Add("@pro\_end", OleDbType.Date).Value = exit;

command.ExecuteNonQuery();

conClose();

}

//bağlantı kapatılır

public void conClose()

{

if (this.dr != null)

{

this.dr.Dispose();

}

this.command.Dispose();

this.com = "";

this.connect.Close();

}

//Çıkış işlemi sırasında veriler proccessing tablolarındaki silinmesi için basit bir metod

public void dataDelete(string tabloName, string sutunName, string id\_data)

{

com = "DELETE FROM " + tabloName + " WHERE " + sutunName + "=" + id\_data + ";";

conOpen();

command.CommandText = com;

command.Connection = connect;

command.ExecuteNonQuery();

conClose();

}

}

//Ana forma yönlendirilmesi için button komutları

private void button\_Min\_Click(object sender, EventArgs e)

{

timer1.Enabled = false;

form\_Ana formAna = new form\_Ana();

formAna.Show();

this.Hide();

}

//exit buttonuna yani çıkış işlemleri butonuna tıklandığında yakarıdaki metotlar sayesinde verileri silinip exitCust tablosuna kayıt edilmesini sağlıyor ve ücreti ekrana yansıtıyor

private void button\_Exit\_Click(object sender, EventArgs e)

{

DialogResult Secim = new DialogResult();

Secim = MessageBox.Show("Araç Çıkış işlemi gerçekleşsin mi? ", "EXIT!", MessageBoxButtons.YesNo, MessageBoxIcon.Question, MessageBoxDefaultButton.Button2);

if (Secim == DialogResult.Yes)

{

MessageBox.Show(label26.Text, "Tutar");

timer1.Enabled = false;

string adCust = "", tcCust = "", carPlaka = "", phoneCust = "", mailCust = "", custPays = "";

custPays = custPay.ToString("##,000.00")+" TL";

DateTime custExit = DateTime.Now;

DateTime custStart = DateTime.Parse(label24.Text);

adCust = label14.Text;

tcCust = label16.Text;

carPlaka = label19.Text;

phoneCust = label17.Text;

mailCust = label18.Text;

dataBase delCust = new dataBase();

delCust.exitRegist(adCust,tcCust,phoneCust,mailCust,carPlaka, custPays, custStart,custExit);

delCust.dataDelete("customers", "id\_cost", id\_cust.ToString());

delCust.dataDelete("cars", "id\_car", id\_car.ToString());

delCust.dataDelete("proccessing", "id\_pro", id\_pro.ToString());

form\_Ana formAna = new form\_Ana();

formAna.Show();

this.Close();

}

}

DateTime start, exit;

TimeSpan fark;

private void form\_Cust\_Load(object sender, EventArgs e)

{

dataBase proc = new dataBase();

proc.conOpen();

proc.veriGetir("proccessing");

while (proc.dr.Read())

{

tc = proc.dr["cust\_tc"].ToString();

plaka = proc.dr["car\_plaka"].ToString();

id\_pro = int.Parse(proc.dr["id\_pro"].ToString());

label23.Text = proc.dr["park\_name"].ToString();

label24.Text = proc.dr["pro\_start"].ToString();

}

proc.conClose();

dataBase car = new dataBase();

car.conOpen();

car.veriGetir("cars");

while (car.dr.Read())

{

id\_car = int.Parse(car.dr["id\_car"].ToString());

label19.Text = car.dr["car\_plaka"].ToString();

label20.Text = car.dr["car\_marka"].ToString();

label21.Text = car.dr["car\_model"].ToString();

label22.Text = car.dr["car\_renk"].ToString();

}

car.conClose();

dataBase cust = new dataBase();

cust.conOpen();

cust.veriGetirCust(tc);

while (cust.dr.Read())

{

id\_cust = int.Parse(cust.dr["id\_cost"].ToString());

label14.Text = cust.dr["cost\_name"].ToString();

label15.Text = cust.dr["cost\_surname"].ToString();

label16.Text = cust.dr["cost\_tc"].ToString();

label17.Text = cust.dr["cost\_phone"].ToString();

label18.Text = cust.dr["cost\_mail"].ToString();

}

car.conClose();

start = DateTime.Parse(label24.Text);

timer1.Enabled = true;

}

//timer sayesinde ekrana anlık olarak her saniye müşterinin vereceği ücret yansıtılır

private void timer1\_Tick(object sender, EventArgs e)

{

exit = DateTime.Now;

fark = exit - start;

custPay = double.Parse(fark.TotalMinutes.ToString()) \* 0.2;

label25.Text = fark.TotalMinutes.ToString("##") + " Dakika - Yaklaşık " + fark.TotalHours.ToString("##") + " Saat";

label26.Text = custPay.ToString("#,###.00") + " TL";

}

}

}