FIRAT ÜNİVERSİTESİ
TEKNOLOJİ FAKÜLTESİ
Yazılım Mühendisliği

YMT463 –SAYISAL GÖRÜNTÜ İŞLEME VE YÖNTEMLERİ

Proje Uygulamaları ve Dokümantasyonu

PLAKA TESPİT SİSTEMİ

Proje Çalışma Grubu

15541516-Hüseyin SAĞLAM

14545534-Merve KARTAL

ÖNSÖZ

Hayatımızın hızla gelişimi artık teknolojinin gelişmesine öncülük etmektedir. Her geçen gün teknolojiye olan ihtiyaç artmaktadır. Bu nedenle teknolojinin hızla gelişmesi artık hayatımızın vazgeçilmez bir parçasıdır. Projemiz akıllı geçiş sistemidir. Projemizde fotoğrafı çekilen araba resimlerinin plakasını tespit etmektedir.

PROJENIN AMACI

Projemiz C# plaka tespit sistemini oluşturduk. Güvenliği sağlamak amaçlı akıllı geçiş sisteminde çekilen araba fotoğraflarına göre plakaların tespitini gerçekleştirip ekran da göstermeyi amaçladık.

PROJENIN ÖZETİ

Projemiz 2 kişilik bir gruptan oluşmaktadır. Projemizde görüntü işleme projesi yapılmaktadır. Bunun için C# kütüphaneleri kullanılmaktadır. Amaçlanan proje de akıllı kameralarla çekilen araba fotoğraflarında plakayı koordinatlarına göre bulup ekranda plakayı göstermektedir. Böylelikle istenilen arabanın plakasını kolaylıkla elde edebiliriz.

KULLANILAN YÖNTEMLER

Projemizde C# kütüphaneleri kullanılmaktadır. Projemiz görüntü işleme projesidir.

PROJEMİZİN KODLARI

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO;
using System.Drawing.Imaging;
using AForge.Imaging.Filters;
using System.Runtime.InteropServices;
using AForge.Imaging;
using AForge.Math.Geometry;
using AForge;
using System.Diagnostics;
using tesseract;
using System.Data.SqlClient;
namespace akıllıgecissistemleri
{
    public partial class OCR : Form
    {
        public OCR ocr1;
```

```
public Form1 frm1;
Form2 frm2 = new Form2();
#region Initialize
private List<string> lstimages = new List<string>();
int position = 0;
private TesseractProcessor m_tesseract = null;
private string m_path = Application.StartupPath + @"\data\";
private const string m_lang = "eng";
#endregion
#region InitializeComponent
/// <summary>
///
/// </summary>
public OCR()
{
    InitializeComponent();
}
#endregion
#region Ocr
private string Ocr(Bitmap image)
{
    m_tesseract.Clear();
    m_tesseract.ClearAdaptiveClassifier();
    return m_tesseract.Apply(image);              }
```

```
        #endregion
        #region OCR_Load
        private void OCR_Load(object sender, EventArgs e)
        {
            m_tesseract = new TesseractProcessor();
            bool succeed = m_tesseract.Init(m_path, m_lang, 3);
            if (!succeed)
            {
                MessageBox.Show("Tesseract initialization failed. The application will exit.");
                Application.Exit();
            }
            m_tesseract.SetVariable("tessedit_char_whitelist", "ABCDEFHKLMNPQRSTUVXYZ-.1234567890").ToString();
            System.Environment.CurrentDirectory = System.IO.Path.GetFullPath(m_path);
        }
        #endregion
        #region FilterLicensePlate
        private string FilterLicensePlate(string licensePlate)
        {
            licensePlate = licensePlate.Replace(".", "");
            licensePlate = licensePlate.Replace("-", "");
            licensePlate = licensePlate.Replace(" ", "");
            licensePlate = licensePlate.Replace("?", "");
            licensePlate = licensePlate.Replace("_", "");
```

```
            licensePlate = licensePlate.Replace(",", "");
            licensePlate = licensePlate.Replace("|", "");
            licensePlate = licensePlate.Replace(" ", "");
            licensePlate = licensePlate.Replace("l", "");
            licensePlate = licensePlate.Replace("~", "");
            licensePlate = licensePlate.Replace(",", "");
            licensePlate = licensePlate.Replace(":", "");
            licensePlate = licensePlate.Replace("'", "");
            licensePlate = licensePlate.Replace("!", "");
            licensePlate = licensePlate.ToUpper();
            return licensePlate;
        }
        #endregion
        private void Dosyasec(object sender, EventArgs e)
        {
            openFileDialog1.Filter = "Resim Dosyaları
(*.bmp)|*.jpg;*.gif;*.bmp;*.png;*.jpeg";
            openFileDialog1.Multiselect = false;
            openFileDialog1.FileName = "";
            if (openFileDialog1.ShowDialog() !=
System.Windows.Forms.DialogResult.OK) return;
            resimYukle(openFileDialog1.FileName);
        }
```

```
private void resimYukle(string resimYolu)
        {
            Bitmap resim = new Bitmap(resimYolu);
            if (resim.Width >= 480 || resim.Height >= 360)
            {
                float katsayi;
                int genislik;
                int yukseklik;
                // Resim boyutları pencereden taşacaksa, en/boy oranını koruyarak resmi yeniden boyutlandıralım
                if (resim.Width - 480 > resim.Height - 360)
                {
                    katsayi = (float)480 / resim.Width;
                    genislik = 480;
                    yukseklik = (int)(katsayi * resim.Height);
                }
                else
                {
                    katsayi = (float)360 / resim.Height;
                    yukseklik = 360;
                    genislik = (int)(katsayi * resim.Width);         }
                Bitmap boyutlandirilmis = new Bitmap(genislik, yukseklik);
                Graphics grafik = Graphics.FromImage(boyutlandirilmis);
                grafik.DrawImage(resim, 0, 0, genislik, yukseklik);
                pictureBox1.Image = boyutlandirilmis;            }
```

```
            else pictureBox1.Image = resim;
            pictureBox1.Enabled = false;
            Application.DoEvents();
            // plakaKonumunuTespitEt(); // Konum tespit işlemini başlat...
        }
        // Otsu Eşikleme
        [DllImport("kernel32.dll")]
        public static extern int GetTickCount();
        [DllImport(@"OtsuEsikleme.dll")]
        public static extern void OtsuEsikleme(ref byte pixelDizisi, ref byte
esikDeger, int genislik, int yukseklik);
        private System.Drawing.Point[] ToPointsArray(List<IntPoint> points)
        {
            return points.Select(p => new System.Drawing.Point(p.X,
p.Y)).ToArray();
        }
        public int[] sinir(List<IntPoint> list)
        {
            int[] sinirlar = { 0, 0, 0, 0 };
            int x1, x2, y1, y2;
            x1 = x2 = y1 = y2 = 0;
            bool first = true;
            foreach (IntPoint p in list)
            {
```

```
    if (first)
        {
            x1 = x2 = p.X;
            y1 = y2 = p.Y;
            first = false;
            continue;
        }
        if (p.X < x1)
            x1 = p.X;
        if (p.X > x2)
            x2 = p.X;
        if (p.Y < y1)
            y1 = p.Y;
        if (p.Y > y2)
            y2 = p.Y;
    }
    sinirlar[0] = x1;
    sinirlar[1] = x2;
    sinirlar[2] = y1;
    sinirlar[3] = y2;
    return sinirlar;
}
```

```
private void button2_Click(object sender, EventArgs e)
        {
         Bitmap bmp = new Bitmap(pictureBox1.Image);
         Bitmap bmpsobe;
         Bitmap bmpmedian;
         Bitmap otsu;
         if (pictureBox1.Image == null)
         {
            MessageBox.Show("Önce bir resim Seçin");
         }
         else
         {
            progressBar1.Visible = true;
            int i, j;
            Color ort;//Color sınıfından bir renk nesne tanımlıyoruz.
            //int r,g,b;
            progressBar1.Maximum = bmp.Width * bmp.Height;//İşlem
çubuğunun maksimim olduğu yer for döngüsünün sonundaki piksel değerine
erişmemiz durumundadır.
            for (i = 0; i <= bmp.Width - 1; i++)//dikey olarak görüntümüzü
tarıyoruz.
            {
               for (j = 0; j <= bmp.Height - 1; j++)//yatay olarak görüntümüzü
tarıyoruz.
               {
```

```
                ort = bmp.GetPixel(i, j);
                ort = Color.FromArgb((byte)((ort.R + ort.G + ort.B) / 3),
(byte)((ort.R + ort.G + ort.B) / 3), (byte)((ort.R + ort.G + ort.B) / 3));
                bmp.SetPixel(i, j, ort);
                if ((i % 10) == 0)//her on satırda bir göstergeyi güncelle
                {
                    progressBar1.Value = i * bmp.Height + j;
                    Application.DoEvents();
                }
            }
        }
    }
    ////////////////////// Median //////////////
    bmpmedian = ExtBitmap.MedianFilter(bmp, 3);
    ////////////////////// SOBEL ///////////////
    bmpsobe = ExtBitmap.Sobel3x3Filter(bmpmedian, true);
    Bitmap bmpsobe1 = (Bitmap)bmpsobe.Clone();
    ////////// otsu
    int x, y;
    int genislik = bmpsobe1.Width;
    int yukseklik = bmpsobe1.Height;
    byte[] pixeller = new byte[(int)genislik * yukseklik];
    Bitmap resim = (Bitmap)bmpsobe1.Clone();
    for (y = 0; y < yukseklik; y++)
        for (x = 0; x < genislik; x++)
```

```
            // Pixelleri kütüphanenin işleyebileceği tek boyutlu bir diziye atıyoruz.
            // Gri seviyede tüm ana renkler eşit olduğu için sadece kırmızıyı okumak gri seviye için yeterli.
            pixeller[y * genislik + x] = resim.GetPixel(x, y).R;
        byte esikDeger = 0;
        OtsuEsikleme(ref pixeller[0], ref esikDeger, genislik, yukseklik);
        int renkk;
        for (y = 0; y < yukseklik; y++)
           for (x = 0; x < genislik; x++)
           {
              renkk = pixeller[y * genislik + x]; // gri
              resim.SetPixel(x, y, Color.FromArgb(renkk, renkk, renkk)); // Gri seviyeyi argb moduna dönüştürüp resme aktarıyoruz.
           }
        otsu = (Bitmap)resim.Clone();
        Bitmap bmperosion;
        Bitmap bmpdilation;
        Bitmap bmpclosing;
        Bitmap kes;
        bmperosion = ExtBitmap.DilateAndErodeFilter(otsu, 3, akıllıgecissistemleri.ExtBitmap.MorphologyType.Erosion, true, true, true);
        Bitmap bmperosionn = (Bitmap)bmperosion.Clone();
        bmpdilation = ExtBitmap.DilateAndErodeFilter(bmperosionn, 7, akıllıgecissistemleri.ExtBitmap.MorphologyType.Dilation, true, true, true);
        Bitmap bmpdilationn = (Bitmap)bmpdilation.Clone();
```

```
            Bitmap one = (Bitmap)bmpdilationn.Clone();
            BlobsFiltering filter0 = new BlobsFiltering();
            // configure filter
            filter0.CoupledSizeFiltering = true;
            filter0.MinWidth = 70;
            filter0.MinHeight = 40;
            // apply the filter
            filter0.ApplyInPlace(one);
            Bitmap one2 = (Bitmap)one.Clone();
            bmpclosing = ExtBitmap.CloseMorphologyFilter(one2, 15, true, true,
true);
            Bitmap bmperosio = (Bitmap)bmpclosing.Clone();
            Bitmap bmperosionson = ExtBitmap.DilateAndErodeFilter(bmperosio,
9, akıllıgecissistemleri.ExtBitmap.MorphologyType.Erosion, true, true, true);
            Bitmap blobson = (Bitmap)bmperosionson.Clone();
            BlobsFiltering filterson = new BlobsFiltering();
            // configure filter
            filterson.CoupledSizeFiltering = true;
            filterson.MinWidth = 50;
            filterson.MinHeight = 200;
            // apply the filter
            filterson.ApplyInPlace(blobson);
            Bitmap rect = (Bitmap)blobson.Clone();
            Bitmap one1 = (Bitmap)rect.Clone();
             ConnectedComponentsLabeling filter = new
          ConnectedComponentsLabeling();
```

```
        // apply the filter
        Bitmap newImage = filter.Apply(one1);
        Bitmap newImage1 = (Bitmap)newImage.Clone();
        // check objects count
        int objectCount = filter.ObjectCount;
        BlobCounter blobCounter = new BlobCounter();
        blobCounter.ProcessImage(rect);
        Blob[] blobs = blobCounter.GetObjectsInformation();
        // create Graphics object to draw on the image and a pen
        Graphics g = Graphics.FromImage(rect);
        Pen bluePen = new Pen(Color.Blue, 2);
        // check each object and draw circle around objects, which
        // are recognized as circles
        textBox1.Text = "";
        for (int i = 0, n = blobs.Length; i < n; i++)
        {
            /*
             * x1=0
             * x2=1
             * y1=2
             * y2=3
             */
            List<IntPoint> edgePoints =
blobCounter.GetBlobsEdgePoints(blobs[i]);
```

```
            List<IntPoint> corners =
PointsCloud.FindQuadrilateralCorners(edgePoints);

            int[] sinirlar = sinir(corners);

            sinirlar[0] = sinirlar[0] - 2;

            sinirlar[1] = sinirlar[1] + 2;

            sinirlar[2] = sinirlar[2] - 2;

            sinirlar[3] = sinirlar[3] + 2;

            int en = sinirlar[1] - sinirlar[0];

            int boy = sinirlar[3] - sinirlar[2];

            float ort = (float)en / (float)boy;

            List<IntPoint> ucnoktalar = new List<IntPoint>();

            ucnoktalar.Add(new IntPoint(sinirlar[0], sinirlar[2]));

            ucnoktalar.Add(new IntPoint(sinirlar[1], sinirlar[2]));

            ucnoktalar.Add(new IntPoint(sinirlar[1], sinirlar[3]));

            ucnoktalar.Add(new IntPoint(sinirlar[0], sinirlar[3]));

            g.DrawPolygon(bluePen, ToPointsArray(ucnoktalar));

            g.DrawString("Plaka kordinatlari : (x,y): (" + sinirlar[0].ToString() +
"," + sinirlar[2].ToString() + ")\n en, boy,ort: " + (sinirlar[1] -
sinirlar[0]).ToString() + ", "

             + (sinirlar[3] - sinirlar[2]).ToString() + "," + ort.ToString() + " blob
sayisi:" + blobs.Length.ToString(), new Font("Arial", 8), Brushes.White, new
System.Drawing.Point(sinirlar[0], sinirlar[3] + 4));         }

        bluePen.Dispose();

        g.Dispose();

        Bitmap rect1 = (Bitmap)pictureBox1.Image.Clone();

        Graphics g1 = Graphics.FromImage(rect1);
```

```
        Pen bluePen2 = new Pen(Color.Red, 2);
        // check each object and draw circle around objects, which
        // are recognized as circles
        textBox1.Text = "";
        List<Blob> bloplar = new List<Blob>();
        for (int i = 0, n = blobs.Length; i < n; i++)
        {
            /*            x1,y1--------x2,y1
             * x1=0         |          |
             * x2=1         |          |
             * y1=2       x1,y2--------x2,y2
             * y2=3
             */
            List<IntPoint> edgePoints =
blobCounter.GetBlobsEdgePoints(blobs[i]);
            List<IntPoint> corners =
PointsCloud.FindQuadrilateralCorners(edgePoints);
            int[] sinirlar = sinir(corners);
            sinirlar[0] = sinirlar[0]-5;
            sinirlar[1] = sinirlar[1];
            sinirlar[2] = sinirlar[2] - 5;
            sinirlar[3] = sinirlar[3] + 5;
            int en = sinirlar[1] - sinirlar[0];
            int boy = sinirlar[3] - sinirlar[2];
            float ort = (float)en / (float)boy;
```

```
            if (ort >= 3 && ort <= 5.7)
            {
                g1.DrawLines(bluePen2, new System.Drawing.Point[] { new
System.Drawing.Point(sinirlar[0], sinirlar[2]),
                    new System.Drawing.Point(sinirlar[1] , sinirlar[2]), new
System.Drawing.Point(sinirlar[1], sinirlar[3]),
                    new System.Drawing.Point(sinirlar[0], sinirlar[3]), new
System.Drawing.Point( sinirlar[0], sinirlar[2]) });
                g1.DrawString("Plaka kordinatlari : (x,y): (" + sinirlar[0].ToString()
+ "," + sinirlar[2].ToString() + ")\n en, boy,ort: " + (sinirlar[1] -
sinirlar[0]).ToString() + ", " + (sinirlar[3] - sinirlar[2]).ToString() + "," +
ort.ToString() + " blob sayisi:" + blobs.Length.ToString(), new Font("Arial", 8),
Brushes.White, new System.Drawing.Point(sinirlar[0], sinirlar[3] + 4));
            }
        }
        bluePen2.Dispose();
        g1.Dispose();
        Bitmap bn = null;
        Bitmap kes1 = (Bitmap)rect1.Clone();
        Graphics g2 = Graphics.FromImage(kes1);
        Pen bluePen3 = new Pen(Color.Red, 2);
        // check each object and draw circle around objects, which
        // are recognized as circles
        textBox1.Text = "";
        for (int i = 0, n = blobs.Length; i < n; i++)
        {
```

```
            /*            x1,y1--------x2,y1
             * x1=0         |          |
             * x2=1         |          |
             * y1=2        x1,y2--------x2,y2
             * y2=3
             */
            List<IntPoint> edgePoints =
blobCounter.GetBlobsEdgePoints(blobs[i]);
            List<IntPoint> corners =
PointsCloud.FindQuadrilateralCorners(edgePoints);
            int[] sinirlar = sinir(corners);
            sinirlar[0] = sinirlar[0]-5;
            sinirlar[1] = sinirlar[1];
            sinirlar[2] = sinirlar[2] - 5;
            sinirlar[3] = sinirlar[3] + 5;
            int en = sinirlar[1] - sinirlar[0];
            int boy = sinirlar[3] - sinirlar[2];
            float ort = en / boy;
            if (ort >= 3 && ort <= 5.7)
            {
                g2.DrawLines(bluePen3, new System.Drawing.Point[] { new
System.Drawing.Point(sinirlar[0], sinirlar[2]),
                    new System.Drawing.Point(sinirlar[1] , sinirlar[2]), new
System.Drawing.Point(sinirlar[1], sinirlar[3]),
                    new System.Drawing.Point(sinirlar[0], sinirlar[3]), new
System.Drawing.Point( sinirlar[0], sinirlar[2]) });
```

```
                for (int ii = 0; ii < kes1.Width; ii++)
                {
                    for (int ji = 0; ji < kes1.Height; ji++)
                    {
                        if ((ii >= sinirlar[0] && ii <= sinirlar[1]) && (ji >= sinirlar[2])
&& ji <= sinirlar[3])
                            continue;
                        else
                            kes1.SetPixel(ii, ji, Color.Black);
                    }
                }
                bn = new Bitmap(en, boy);
                Graphics g3 = Graphics.FromImage(bn);
                g3.DrawImage(kes1, -sinirlar[0], -sinirlar[2]);
                pictureBox2.Image = bn;
            }
        }
        bluePen3.Dispose();
        g2.Dispose();
       form(bmp, "Grayscala");
        form(bmpmedian, "Median");
        form(bmpsobe, "sobel");
        form(otsu, "otsu");
        form(bmperosion, "erosion");
        form(bmpdilationn, "dilation");
```

```
        form(one, "one");
        form(bmpclosing, "closing");
        form(bmperosionson, "son");
        form(blobson, "blobson");
        form(newImage, "one1");
        form(rect, "rect");
        form(rect1, "rect1");
        form(kes1, "kes1");
        form(bn, "bn");
    }
    public void form(Bitmap bmp, String isim)
    {
       Form1 frmfrm = new Form1();
       frmfrm.Name = isim;
       frmfrm.Text = isim;
       frmfrm.setImage(bmp);
       if (bmp != null)
       {
          frmfrm.Height = bmp.Height + 50;
          frmfrm.Width = bmp.Width + 50;
       }
      frmfrm.Show();
    }
    #region btnLoad_Click
```

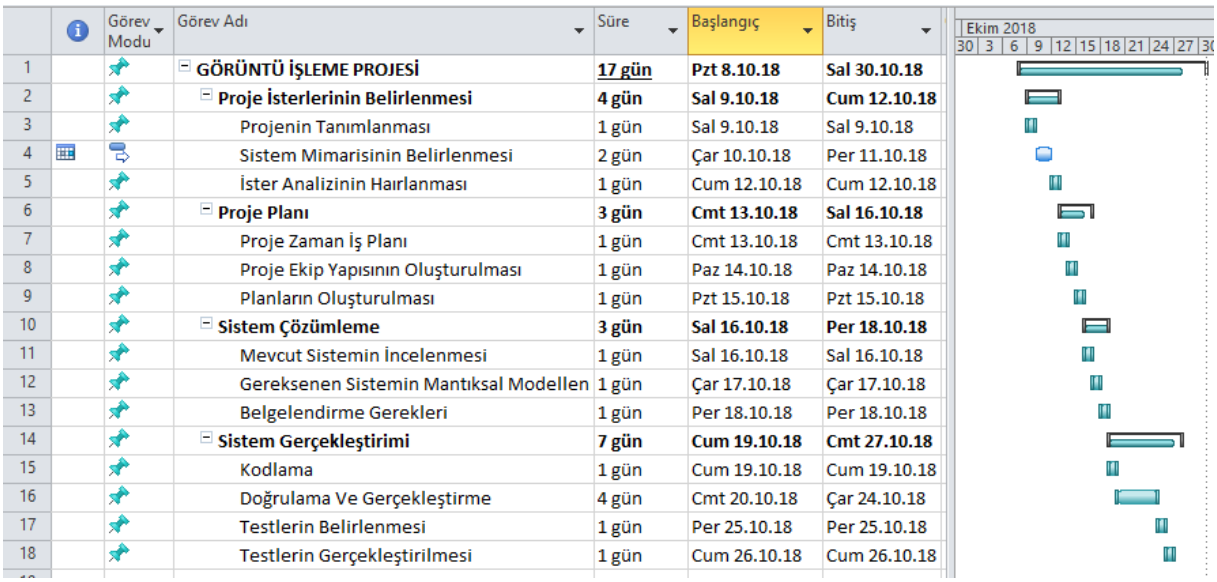
```
private void btnLoad_Click(object sender, EventArgs e)
{
    ProcessImage();
}
#endregion
#region ProcessImage
public void ProcessImage()
{
    Stopwatch watch = Stopwatch.StartNew(); // time the detection process
    Bitmap root = new Bitmap(pictureBox2.Image);
    Bitmap bmp = new Bitmap(root);
    Bitmap bmp1 = new Bitmap(root);
    Bitmap bmp2 = new Bitmap(root);
    string outputstring = "";
    ///root
    outputstring = this.Ocr(root);
    // txtNumbePlate1.Text = outputstring;
    //ptBNumberPlate1.Image = root;
    //resize image to 1920x1080
    outputstring = "";
    ResizeBilinear filterr = new ResizeBilinear(1920, 1080);
    bmp = filterr.Apply(bmp);
    outputstring = this.Ocr(bmp);
    txtNumbePlate1.Text = outputstring;
```

```
            ptBNumberPlate1.Image = bmp;
        }
        #endregion
        private void txtNumbePlate1_TextChanged(object sender, EventArgs e)
        {
            SqlConnection bag = new SqlConnection(" Data Source=DESKTOP-
BTFMRHO;Initial Catalog=plakatanimasistemi;Integrated Security=True");
            DataTable tablo = new DataTable();
            SqlDataAdapter adtr = new SqlDataAdapter();
            SqlCommand kmt = new SqlCommand();
            bag.Open();
            kmt.CommandText = "select * from araclar";
            kmt.Connection = bag;
            SqlDataReader oku = kmt.ExecuteReader();
            while (oku.Read())
            {
                String test = txtNumbePlate1.Text.Replace(" ", "").Replace("\n", "");
                //label2.Text = "" + txtNumbePlate2.Text.Replace(" ", "") + "-" +
oku[1].ToString();
            if (test == oku[1].ToString())
                    {
                    // listBox1.Items.Add(oku[0].ToString());
                    listBox1.Items.Add(oku[1].ToString());
                    listBox1.Items.Add(oku[2].ToString());
                    listBox1.Items.Add(oku[3].ToString());
```

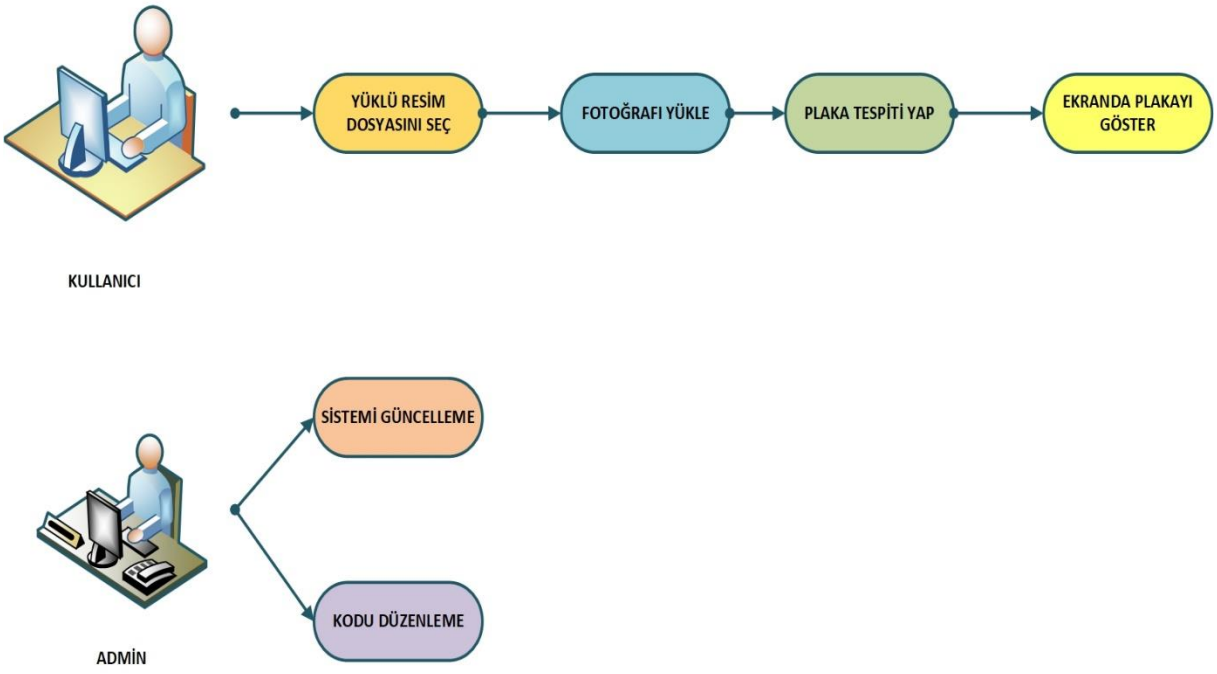
```
                listBox1.Items.Add(oku[4].ToString());
                listBox1.Items.Add(oku[5].ToString());
            }
            else
            {
                MessageBox.Show("yok:");
            }
        }
        oku.Close();
        bag.Close();
    }
    private void button3_Click(object sender, EventArgs e)
    {
      frm2.Show();
    }
  }
}
```

GANT DİYAGRAMI

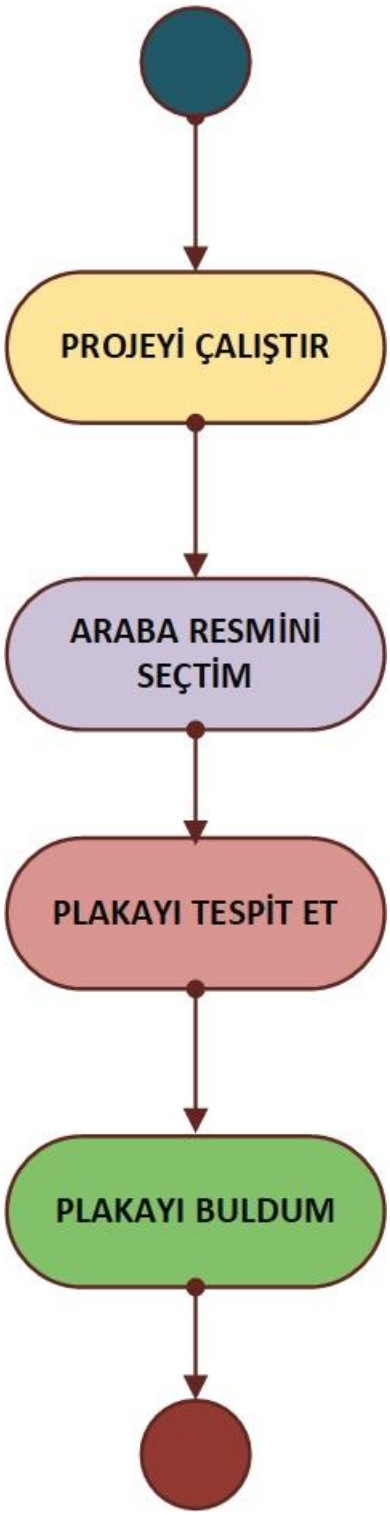
	Görev Modu	Görev Adı	Süre	Başlangıç	Bitiş
1		**GÖRÜNTÜ İŞLEME PROJESİ**	**17 gün**	**Pzt 8.10.18**	**Sal 30.10.18**
2		**Proje İsterlerinin Belirlenmesi**	**4 gün**	**Sal 9.10.18**	**Cum 12.10.18**
3		Projenin Tanımlanması	1 gün	Sal 9.10.18	Sal 9.10.18
4		Sistem Mimarisinin Belirlenmesi	2 gün	Çar 10.10.18	Per 11.10.18
5		İster Analizinin Hairlanması	1 gün	Cum 12.10.18	Cum 12.10.18
6		**Proje Planı**	**3 gün**	**Cmt 13.10.18**	**Sal 16.10.18**
7		Proje Zaman İş Planı	1 gün	Cmt 13.10.18	Cmt 13.10.18
8		Proje Ekip Yapısının Oluşturulması	1 gün	Paz 14.10.18	Paz 14.10.18
9		Planların Oluşturulması	1 gün	Pzt 15.10.18	Pzt 15.10.18
10		**Sistem Çözümleme**	**3 gün**	**Sal 16.10.18**	**Per 18.10.18**
11		Mevcut Sistemin İncelenmesi	1 gün	Sal 16.10.18	Sal 16.10.18
12		Gereksenen Sistemin Mantıksal Modellen	1 gün	Çar 17.10.18	Çar 17.10.18
13		Belgelendirme Gerekleri	1 gün	Per 18.10.18	Per 18.10.18
14		**Sistem Gerçekleştirimi**	**7 gün**	**Cum 19.10.18**	**Cmt 27.10.18**
15		Kodlama	1 gün	Cum 19.10.18	Cum 19.10.18
16		Doğrulama Ve Gerçekleştirme	4 gün	Cmt 20.10.18	Çar 24.10.18
17		Testlerin Belirlenmesi	1 gün	Per 25.10.18	Per 25.10.18
18		Testlerin Gerçekleştirilmesi	1 gün	Cum 26.10.18	Cum 26.10.18



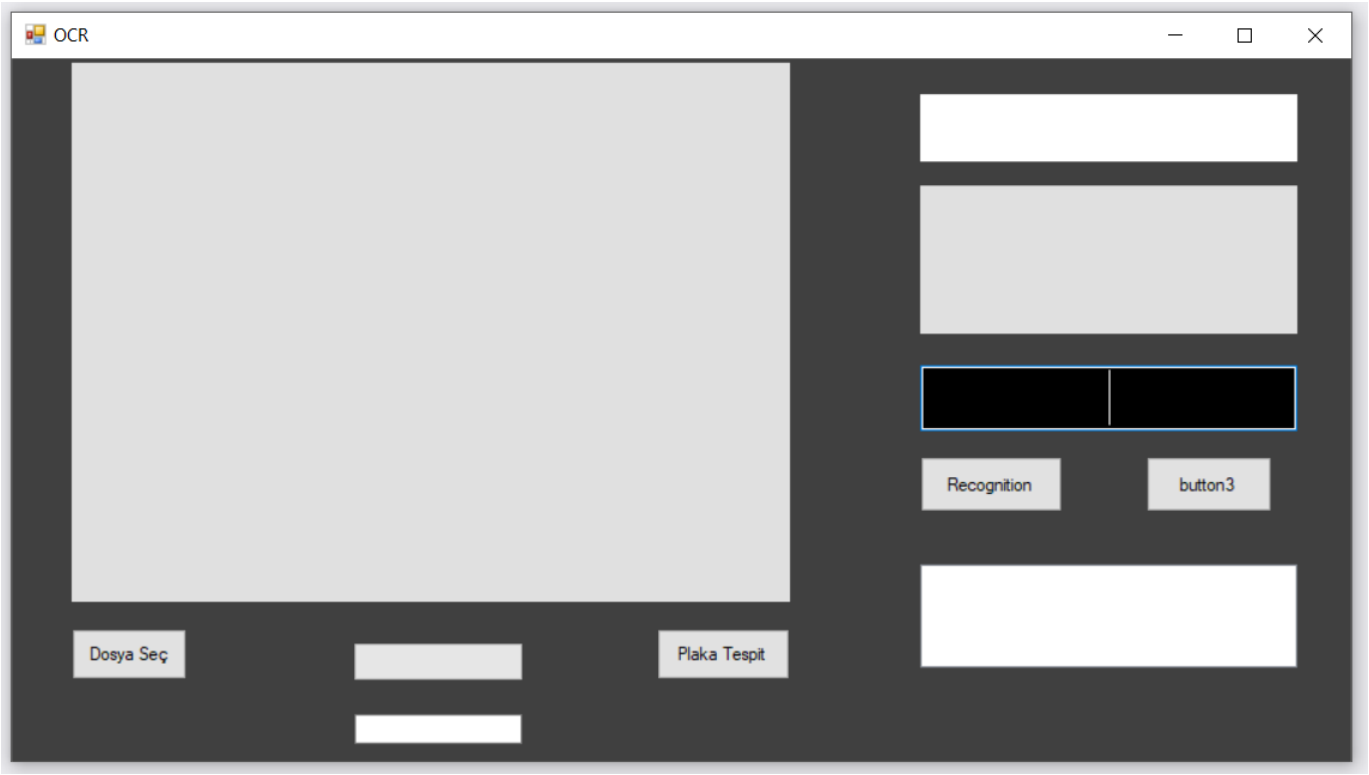
USE-CASE DİYAGRAMI



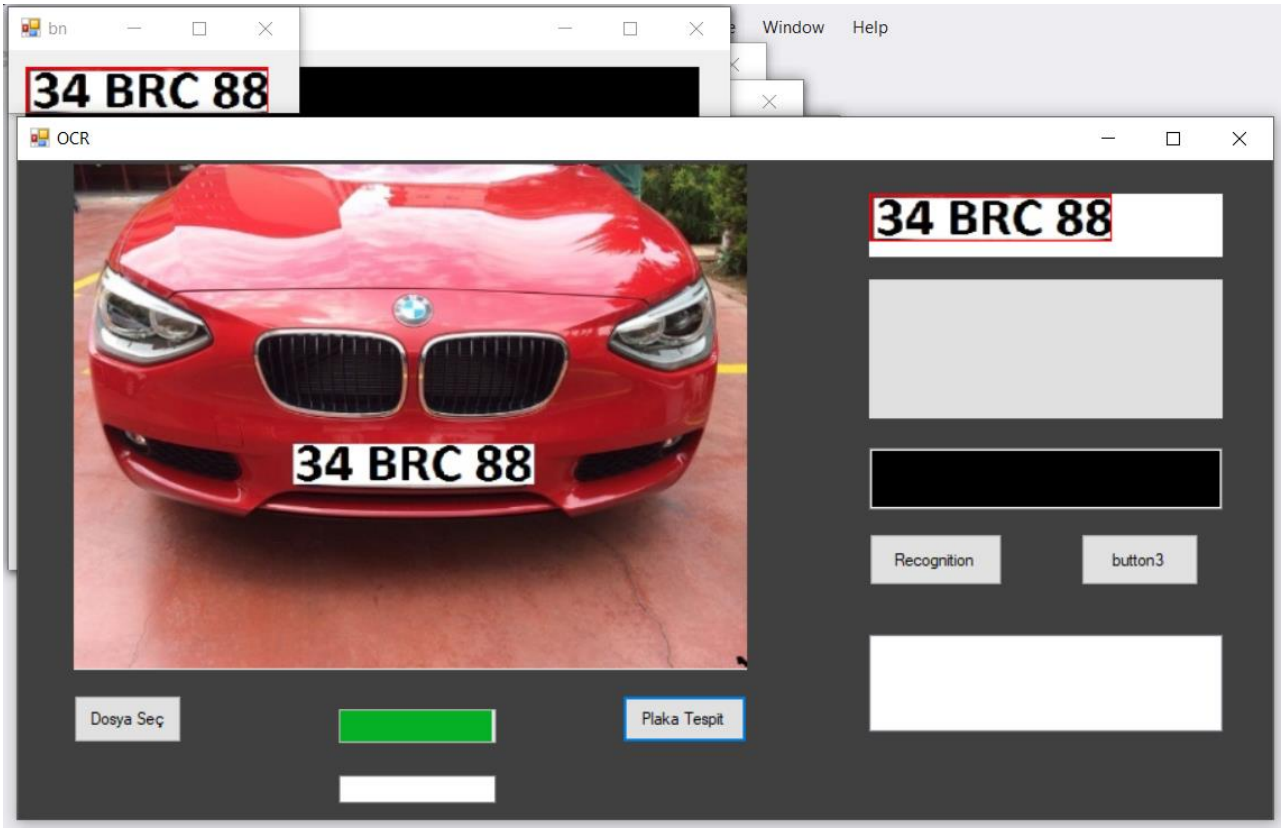
ACTİVİTY DİYAGRAMI



ARAYÜZ



Bu arayüz açılış ekranımızdır. Dosya Seç butonuna tıklayarak, image klasöründeki resimlerden birini seçiyoruz.



Bu arayüzümüz de ise araba resmini seçtikten sonra plaka tespit butonuna tıkladığımız da plaka tespiti gerçekleşecektir ve farklı bir form ekranında ve resmin yan tarafında plakayı gösterecektir.