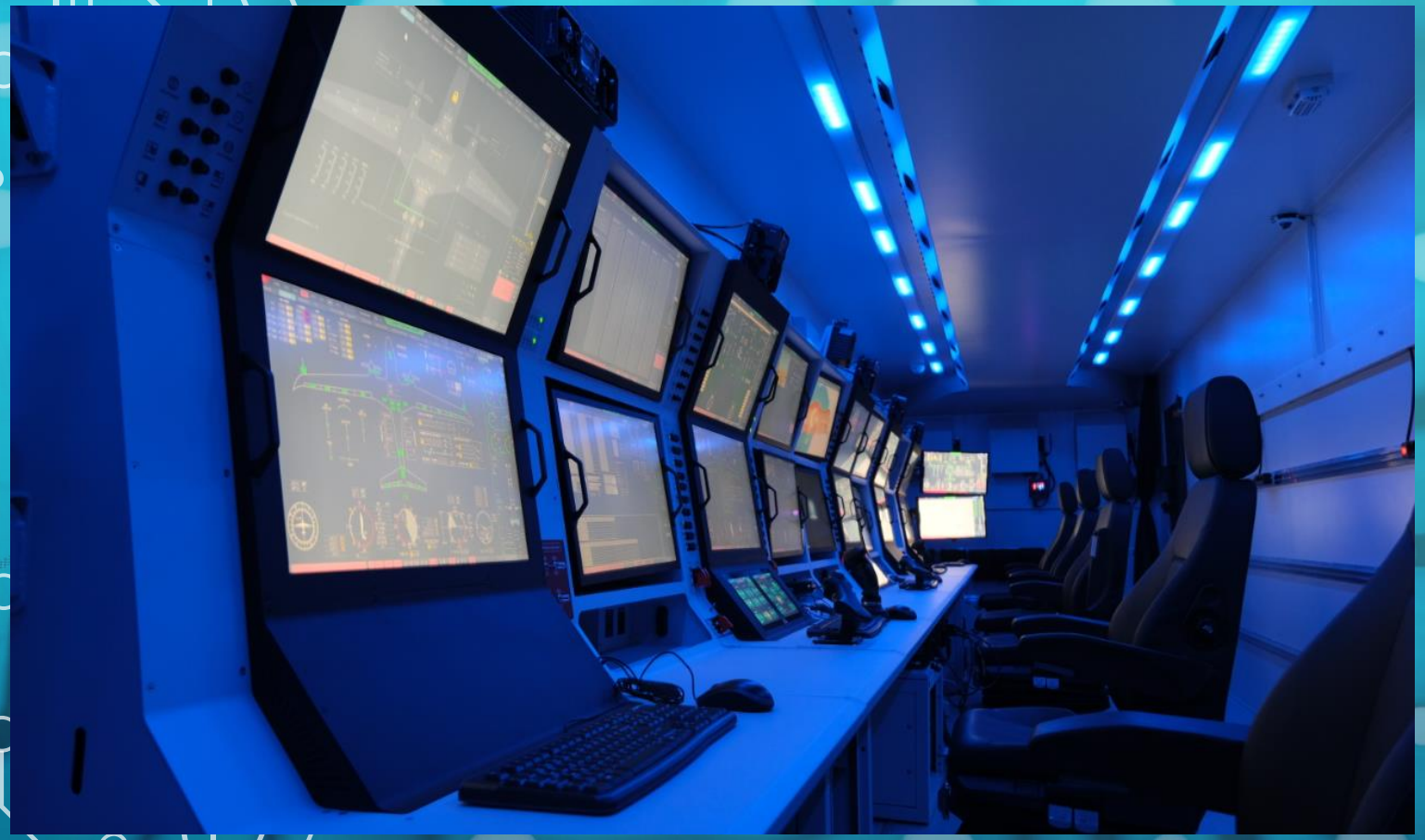
DOKUNMATİK PANEL KALİBRASYON VE BAKIM ARAYÜZÜ

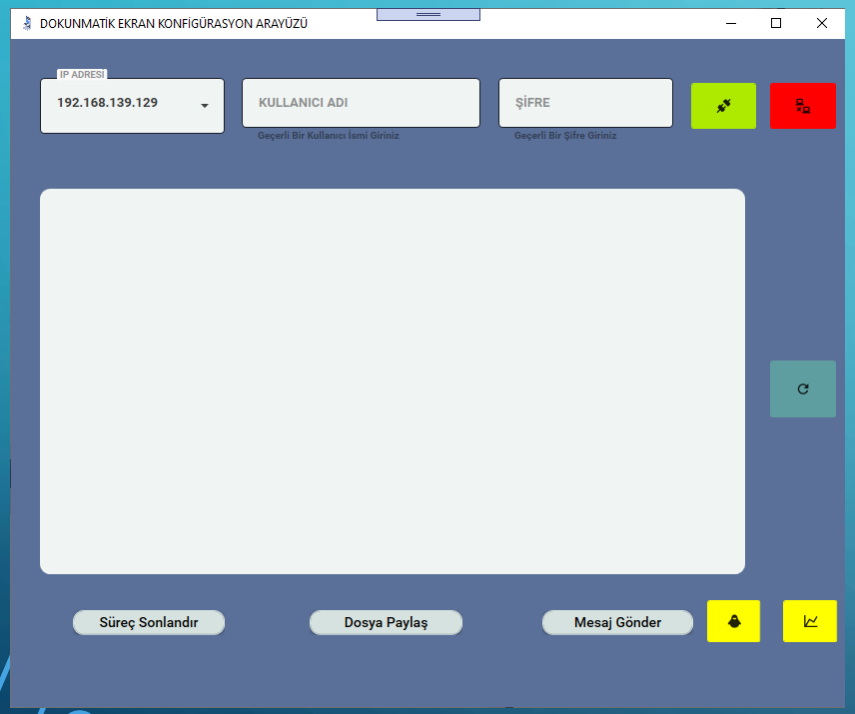
HAKAN AKGÜN

BURAK KARAMETE

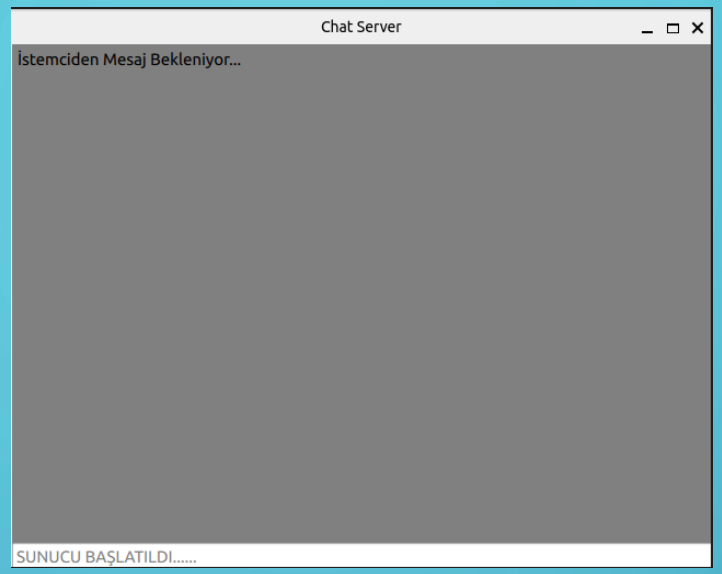


PROJE AMACI

- Bu proje , yer istasyonunda bulunan bilgisayar ile İHA kullanımına yardımcı arayüze sahip tabletler arasındaki kalibrasyonu sağlayan ve bazı teknik işlemleri gerçekleştirilmesi sağlayan bir arayüz projesidir.
- Böylelikle tablete erişim sağlamadan yer istasyonu bilgisayarından tablet ile ilgili işler gerçekleştirilebilecektir.



Şekil 1. Windows C# uygulaması



Şekil 2. Linux C++ uygulaması

- Windows bilgisayarda çalışacak olan proje C# WPF MVVM pattern ve Material Design ile oluşturulmuştur. Tableti simüle edecek sanal makineye(LİNUX) bağlanarak işlemleri gerçekleştirir.

- Tablette yani sanal makinede de olan arayüz projesi ise mesajlaşma uygulaması olarak, QT C++ QML ile gerçekleştirilmiştir.

MVVM PATTERN

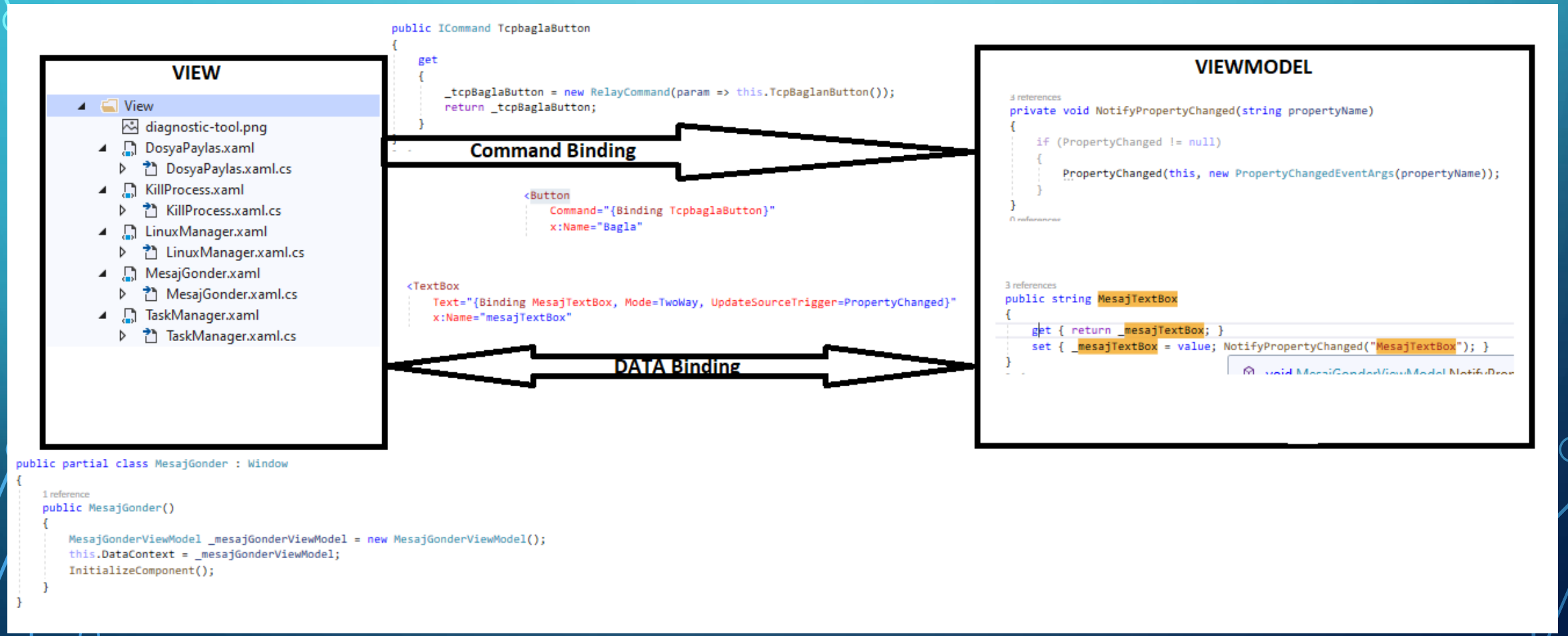
- MVVM: Presentation Model mimarisinin WPF ve Silverlight teknolojileri için özelleştirilmiş bir halidir.(Presentation model presenter ile karşılaştırıldığında View ile çok daha fazla konuşkan iletişime sahiptir.) Arayüz(View) ile logic işlemlerin(model) birbirlerinden bağımsız çalışmasını sağlayan yapıdır. Amacı geliştirmeyi kolaylaştırmak ve frontend developer ile backend developerların bağımsız çalışabilmesidir. Temel parçaları;
- Model: Model, uygulama içinde kullanacağımız verilerdir.
- View: View, datanın sunulduğu katmandır. Tüm görsellikler viewda yer alır. Kısaca verinin sunulduğu yerdir(Ekran). Bu sınıfların amacı ViewModelden istediğimiz verileri Observe etmektir. Burada lojik işlemler gerçekleştirilmez
- ViewModel: ViewModel ise model ve viewı bağlayan yapıdır. View ile model arasında bir yapıştırıcı görevi görür. View doğrudan ViewModel yardımıyla modele erişir ve bazı işlemleri gerçekleştirir. Teknolojik olarak WPFden konuşursak ViewModel aslında viewın DataContext idir. Bu sınıflarda lojik işlemler gerçekleştirilir.

DataContext: *Veri bağlama işlemlerinde kullanılmaktadır.* Bu bileşenler sayesinde XAML içerisinden, ilgili kaynaklara bağlanma, tek yönlü ve çift yönlü olarak veri transfer etme(Data Binding) işlemlerini gerçekleştirebiliriz.

Neden MVVM:

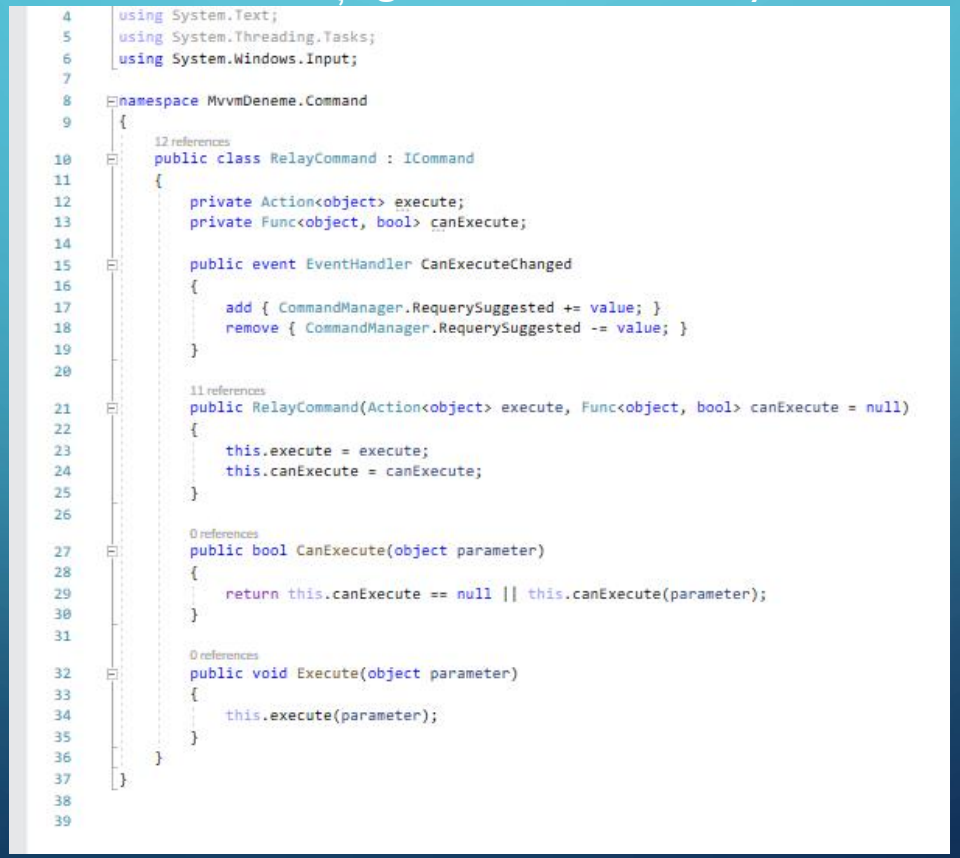
* Commanding, Binding gibi özellikleri kullanarak hızlı ve etkin uygulama geliştirme.
* View arkasında bulunan kod dosyasına sıfır kod yazarak bağımlılıkları minimuma indirmek ve uygulama içerisinde katmanların bağımlılıklarını minimuma indirmek.
* Daha genişletilebilir, bakımı kolay ve test edilebilir uygulama geliştirme.
* Arayüzü kolay bir şekilde taşınabilen uygulamalar geliştirmek.

DİAGRAM


VIEW
View
diagnostic-tool.png
DosyaPaylas.xaml
DosyaPaylas.xaml.cs
KillProcess.xaml
KillProcess.xaml.cs
LinuxManager.xaml
LinuxManager.xaml.cs
MesajGonder.xaml
MesajGonder.xaml.cs
TaskManager.xaml
TaskManager.xaml.cs
public ICommand TcpbaglaButton
{
get
{
_tcpBaglaButton = new RelayCommand(param => this.TcpBaglanButton());
return _tcpBaglaButton;
}
}
Command Binding
<Button
Command="{Binding TcpbaglaButton}"
x:Name="Bagla"
<TextBox
Text="{Binding MesajTextBox, Mode=TwoWay, UpdateSourceTrigger=PropertyChanged}"
x:Name="mesajTextBox"
DATA Binding
VIEWMODEL
3 references
private void NotifyPropertyChanged(string propertyName)
{
if (PropertyChanged != null)
{
PropertyChanged(this, new PropertyChangedEventArgs(propertyName));
}
}
3 references
public string MesajTextBox
{
get { return _mesajTextBox; }
set { _mesajTextBox = value; NotifyPropertyChanged("MesajTextBox"); }
}
public partial class MesajGonder : Window
{
1 reference
public MesajGonder()
{
MesajGonderViewModel _mesajGonderViewModel = new MesajGonderViewModel();
this.DataContext = _mesajGonderViewModel;
InitializeComponent();
}
}

- **NotifyPropertyChanged:** C# tarafından uygulama ekranında bulunan herhangi bir UI Control'ünün değeri değiştirme işlemlerini sık sık yaparız. İşte bu gibi işlemleri örneğin TextBlock'un Text'ini değiştirme işlemini C# tarafında tblName.Text="Hakan"; yazmak yerine INotifyPropertyChanged interface'ini kullanarak bu gibi işlemleri kolaylıkla ve daha yönetilebilir bir şekilde yapabiliriz. Sadece text için değil buton click eventleri içinde kullanılabilir.

- **RelayCommand.cs Class'ı:** RelayCommand button'a tıklandığında çalışacak olan event gibi düşünebiliriz, Butonun click statelerini aşağıda ki metodlar sayesinde handle edip yönetimini sağlıyoruz



```
using System.Text;
using System.Threading.Tasks;
using System.Windows.Input;

namespace MvvmDeneme.Command
{
    12 references
    public class RelayCommand : ICommand
    {
        private Action<object> execute;
        private Func<object, bool> canExecute;

        public event EventHandler CanExecuteChanged
        {
            add { CommandManager.RequerySuggested += value; }
            remove { CommandManager.RequerySuggested -= value; }
        }

        11 references
        public RelayCommand(Action<object> execute, Func<object, bool> canExecute = null)
        {
            this.execute = execute;
            this.canExecute = canExecute;
        }

        0 references
        public bool CanExecute(object parameter)
        {
            return this.canExecute == null || this.canExecute(parameter);
        }

        0 references
        public void Execute(object parameter)
        {
            this.execute(parameter);
        }
    }
}
```

İlk Ekran Kodları

```
291             }
292         }
            1 reference
293         public void Baglanti()
294         {
295             if (TextSelect == null || SifreSelect == null || SelectedIps == null)
296             {
297                 MessageBox.Show("Bilgileri Eksiksiz Giriniz!");
298             }
299             else
300             {
301                 host = SelectedIps.ToString();
302                 username = TextSelect.ToString();
303                 password = SifreSelect.ToString();
304                 string result = SshBaglanti("ps -aux");//ps -aux
305                 string[] subStrings = result.Split('\n');
306                 MyListItems = subStrings;
307             }
308         }
            1 reference
309         private string SshBaglanti(string command)
310         {
311             string result = string.Empty;
312             try
313             {
314                 AuthenticationMethod method = new PasswordAuthenticationMethod(username, password);
315                 ConnectionInfo connection = new ConnectionInfo(host, username, method); // ip: 192.168.139.129  kullaciadi: hkn  sifre: 123
316                 sshClient = new SshClient(connection);
317
318                 sshClient.Connect();
319                 SshCommand cmd = sshClient.RunCommand(command);
320                 result = cmd.Result;
321             }
322             catch (Exception ex)
323             {
324                 MessageBox.Show(ex.Message);
325                 if (!sshClient.IsConnected)
326                 {
327                     TextSelect = "";
328                     SifreSelect = "";
329                     MessageBox.Show("Bağlanamadı, Kontrol Ediniz!");
330                 }
331             }
332             return result;
333         }
334     }
335 }
336
```

Şekil 3. SSH Bağlantısı

```
            1 reference
197         public void BaglantiKopar()
198         {
199             string[] arr = new string[] { };
200             if (!(this.sshClient == null) && this.sshClient.IsConnected)
201             {
202                 this.sshClient.Disconnect();
203                 SelectedIps = null;
204                 TextSelect = string.Empty;
205                 SifreSelect = string.Empty;
206                 MyListItems = arr;
207                 CpuLabel = "";
208                 MemLabel = "";
209                 MessageBox.Show("Bağlantı Koptu");
210             }
211             else
212                 MessageBox.Show("Bağlantınız yok");
213         }
            1 reference
214         public void Yenile()
215         {
216             string result = string.Empty;
217             if ((this.sshClient == null) || !(this.sshClient.IsConnected))
218             {
219                 MessageBox.Show("Bağlantınız Bulunmamaktadır.");
220             }
221             else
222             {
223                 SshCommand cmd = this.sshClient.RunCommand("ps -aux");
224                 result = cmd.Result;
225             }
226             string[] subStrings = result.Split('\n');
227             MyListItems = subStrings;
228
229             CpuLabel = "";
230             MemLabel = "";
231         }
```

Şekil 4. Bağlantı Kopar ve Yenile

SSH: Bir bilgisayar ile diğer bir bilgisayar arasında güvenli bir bağlantı kurulumunu sağlayan ağ protokolüdür. Bilgisayar/Sunucu, Sunucu/Sunucu arasında bağlantıları sağlar. Ssh iki bilgisayar arasındaki bağlantıyı gizli ve açık olarak iki anahtarla şifreler.

```
1 reference
232     public void ProcessSonlandır()
233     {
234         try
235         {
236             string secilen = MyListItems[SelectedIndex].ToString();
237
238             string[] parse = secilen.Split(' ');
239
240             int index = 0;
241             string pid = string.Empty;
242             string cpu = string.Empty;
243             string memory = string.Empty;
244             foreach (string c in parse)
245             {
246                 if (c.Length != 0)
247                 {
248                     if (index == 1)
249                     {
250                         pid = c;
251                     }
252                     else if (index == 2)
253                     {
254                         cpu = c;
255                     }
256                     else if (index == 3)
257                     {
258                         memory = c;
259                         break;
260                     }
261                     index++;
262                 }
263             }
264             if (!(secilen == MyListItems[0]))
265             {
266                 if (secilen != null || secilen == "")
267                 {
268                     CpuLabel = (pid + " Numaralı Sürecin CPU Kullanımı(%): " + cpu);
269                     MemLabel = (pid + " Numaralı Sürecin Memory Kullanımı(%)" + memory);
270                 }
271                 if (this.host == "" || this.username == "" || this.password == "")
272                 {
273                     MessageBox.Show("Lütfen IP Kullanıcı Ve Şifreyi Giriniz");
274                 }
275                 else
276                 {
277                     KillProcess _killProcessView = new KillProcess(sshClient, pid);
278                     Application.Current.MainWindow = _killProcessView;
279                     _killProcessView.surecLbl.Content = (pid + " Numaralı Süreci Sonlandırmaya Emin misiniz?");
280                     Application.Current.MainWindow.Show();
281                 }
282             }
283             else
284             {
285                 MessageBox.Show("Yanlış Seçim, Lütfen Bir Süreç Seçiniz.");
286             }
287         }
288         catch (Exception)
289         {
290             MessageBox.Show("Lütfen Bir Süreç Seçiniz");
```

Şekil 5. Süreç Sonlandırma

```
1 reference
174     public void LinuxManage()
175     {
176         LinuxManager _linuxManagerView = new LinuxManager(this.sshClient);
177         Application.Current.MainWindow = _linuxManagerView;
178     }
1 reference
179     public void TaskManage()
180     {
181         TaskManager _taskManagerView = new TaskManager();
182         Application.Current.MainWindow = _taskManagerView;
183         //Application.Current.MainWindow.Show();
184     }
1 reference
185     public void DosyaGonder()
186     {
187         DosyaPaylas _dosyaGonderView = new DosyaPaylas(host, username, password);
188         Application.Current.MainWindow = _dosyaGonderView;
189         Application.Current.MainWindow.Show();
190     }
1 reference
191     public void MesajGonder()
192     {
193         MesajGonder _mesajGonderView = new MesajGonder();
194         Application.Current.MainWindow = _mesajGonderView;
195         Application.Current.MainWindow.Show();
196     }
```

Şekil 6. İlk Ekranda Olan Diğer Butonlar

Süreç Sonlandır Ekranı

```
namespace MvvmDeneme.ViewModel
{
    3 references
    public partial class KillProcessViewModel : INotifyPropertyChanged
    {
        private ICommand _pidSonlandirButton;
        private SshClient sshclient;
        private string pidSayisi = string.Empty;
        private string _tp;
        public event PropertyChangedEventHandler PropertyChanged;
        1 reference
        private void NotifyPropertyChanged(string propertyName)
        {
            if (PropertyChanged != null)
            {
                PropertyChanged(this, new PropertyChangedEventArgs(propertyName));
            }
        }
        1 reference
        public string TextSelect
        {
            get { return _tp; }
            set { _tp = value; NotifyPropertyChanged("TextSelect"); }
        }
        1 reference
        public KillProcessViewModel(SshClient sshClient, string pidSayisi)
        {
            this.sshclient = sshClient;
            this.pidSayisi = pidSayisi;
        }

        0 references
        public ICommand pidSonlandirButton
        {
            get
            {
                _pidSonlandirButton = new RelayCommand(param => this.ProcessSonlandirPid());
                return _pidSonlandirButton;
            }
        }
        1 reference
        public void ProcessSonlandirPid()
        {
            string command = ("kill -9 " + this.pidSayisi);
            SshCommand cmd = sshclient.RunCommand(command);
            if (cmd.Error == null)
            {
                TextSelect = (this.pidSayisi + " numaralı sürec sonlandirilmiştir.");
            }
            else
            {
                MessageBox.Show(cmd.Error);
                Application.Current.MainWindow.Close();
            }
        }
    }
}
```

Şekil 7. Süreç Sonlandır Butonu

Şekil 8. KillProcess.xaml

Dosya Paylaşımı Ekranı

```
1 reference
private void comboBox_SelectionChanged()
{
    if (host == "" || username == "" || password == "")
    {
        MessageBox.Show("Bağlantı Yok!");
    }
    else
    {
        using (SftpClient client = new SftpClient(this.host, 22, this.username, this.password))
        {
            client.KeepAliveInterval = TimeSpan.FromSeconds(60);
            client.ConnectionInfo.Timeout = TimeSpan.FromMinutes(180);
            client.OperationTimeout = TimeSpan.FromMinutes(180);
            client.Connect();
            if (SelectServerClient == "SUNUCU/İSTEMCİ")
            {
                ButtonGizleB = Visibility.Hidden;
                ButtonGizleA = Visibility.Visible;
                LblFilePath = "";
                treeView.Clear();
                ListDirectory(treeView, "C:\\Users\\PC_1770\\Desktop");//C:\\Users\\PC_1770\\Desktop
            }
            if (SelectServerClient == "İSTEMCİ/SUNUCU")
            {
                ButtonGizleA = Visibility.Hidden;
                ButtonGizleB = Visibility.Visible;
                LblFilePath = "";
                treeView.Clear();
                DownloadDirectoryForLinux(client, "/home/hkn/Desktop");
                //RemoteFileButton_Click(this.sftpclient);
            }
        }
    }
}
```

Şekil 9. SFTP Bağlantısı

SFTP: FTP bilgisayarlar arasında güvenli dosya alışverişi standart yöntemidir. SFTP ise daha güvenli halidir. SSH protokolünü kullanır. Paketler şifrelenir ve paketler halinde taşınır. SFTP SSH ile aynı port üzerinde çalışır(22). SFTPde log kayıtları tutulur.

```
private void ListDirectory(ObservableCollection<TreeViewItem> treeView, string path)
{
    //treeView.Clear();
    var rootDirectoryInfo = new DirectoryInfo(path);

    treeView.Add(CreateDirectoryNode(rootDirectoryInfo));
}
2 references
private static TreeViewItem CreateDirectoryNode(DirectoryInfo directoryInfo)
{
    var directoryNode = new TreeViewItem { Header = directoryInfo.Name };
    foreach (var directory in directoryInfo.GetDirectories())
        directoryNode.Items.Add(CreateDirectoryNode(directory));

    foreach (var file in directoryInfo.GetFiles())
        directoryNode.Items.Add(new TreeViewItem { Header = file.Name });

    return directoryNode;
}
2 references
private void DownloadDirectoryForLinux(SftpClient client, string remotePath)
{
    try
    {
        IEnumerable<SftpFile> files = client.ListDirectory(remotePath);//lblFilePath2.Content.ToString()
        foreach (SftpFile file in files)
        {
            if ((file.Name != ".") && (file.Name != ".."))
            {
                string sourceFilePath = remotePath + "/" + file.Name;

                if (file.IsDirectory)
                {
                    var directoryNode = new TreeViewItem { Header = "**" + file.Name };
                    treeView.Add(directoryNode);//?
                    DownloadDirectoryForLinux(client, sourceFilePath);
                }
                else
                {
                    var directoryNode = new TreeViewItem { Header = file.Name };
                    treeView.Add(directoryNode);
                }
            }
        }
    }
    catch (Exception ex)
    {
        throw;
    }
}
```

(local variable) SftpFile file

Şekil 10. Treeview Linux Bilgisayar Ve Windows Bilgisayar için Veri Yükleme

```
35      public DosyaPaylas(string host, string username, string password)
36      {
37          this.host = host;
38          this.username = username;
39          this.password = password;
40
41          DosyaPaylasViewModel _dosyaPaylasViewModel = new DosyaPaylasViewModel(host, username, password);
42          this.DataContext = _dosyaPaylasViewModel;
43          InitializeComponent();
44      }
        1 reference
45      private void DowloadFileButton_Click(object sender, RoutedEventArgs e)
46      {
47          string localDir = @"C:\Users\PC_1770\Desktop";
48          var remoteDir = "/home/hkn/Desktop";
49
50          using (SftpClient client = new SftpClient(this.host, this.username, this.password))
51          {
52              client.KeepAliveInterval = TimeSpan.FromSeconds(60);
53              client.ConnectionInfo.Timeout = TimeSpan.FromMinutes(180);
54              client.OperationTimeout = TimeSpan.FromMinutes(180);
55              client.Connect();
56              bool connected = client.IsConnected;
57              if (connected == true)
58                  MessageBox.Show("SFTP Bağlantısı Sağlandı");
59
60              client.ChangeDirectory(remoteDir);
61
62              DownloadDirectory(client, remoteDir, localDir);
63
64              MessageBox.Show("Dosyanız Server İle Paylaşılmıştır.");
65
66              client.Disconnect();
67          }
68      }
        1 reference
69      private void ShareFileButton_Click(object sender, RoutedEventArgs e)
70      {
71          string localDir = @"C:\Users\PC_1770\Desktop";
72          var remoteDir = "/home/hkn/Desktop";
73
74          using (SftpClient client = new SftpClient(this.host, 22, this.username, this.password))
75          {
76              client.KeepAliveInterval = TimeSpan.FromSeconds(60);
77              client.ConnectionInfo.Timeout = TimeSpan.FromMinutes(180);
78              client.OperationTimeout = TimeSpan.FromMinutes(180);
79              client.Connect();
80              bool connected = client.IsConnected;
81              if (connected == true)
82                  MessageBox.Show("SFTP Bağlantısı Sağlandı.");
83
84              client.ChangeDirectory(remoteDir);
85              //MessageBox.Show(client.WorkingDirectory);
86
87              UploadDirectory(client, localDir, remoteDir);
88              MessageBox.Show("Dosyanız Client İle Paylaşılmıştır.");
89
90              client.Disconnect();
91          }
92      }
```

Şekil 11. Xaml Üzerindeki Buton Click Eventleri

```
private void DownloadDirectory(SftpClient client, string remotePath, string localPath)
{
    if (lblFilePath.Content.ToString() == "")
    {
        MessageBox.Show("Lütfen Paylaşım Yapacağınız Dosyayı Seçiniz!!!");
    }
    try
    {
        Directory.CreateDirectory(localPath);
        IEnumerable<SftpFile> files = client.ListDirectory(remotePath);//lblFilePath2.Content.ToString()
        string destFilePath = System.IO.Path.Combine(localPath, lblFilePath.Content.ToString());

        string a = "/home/hkn/Desktop/" + lblFilePath.Content.ToString();
        using (Stream fileStream = File.Create(destFilePath))
        {
            if (lblFilePath.HasContent)
                client.DownloadFile(a, fileStream);
        }
        this.Close();
    }
    catch (Exception ex)
    {
        throw;
    }
}
```

Şekil 12. Treeview Seçilene Göre Dosya İndirme Yükleme

```
2 references
private void UploadDirectory(SftpClient client, string localPath, string remotePath)
{
    if (lblFilePath.Content.ToString() == "")
    {
        MessageBox.Show("Lütfen Paylaşım Yapacağınız Dosyayı Seçiniz!!!");
    }
    try
    {
        IEnumerable<FileSystemInfo> infos = new DirectoryInfo(localPath).EnumerateFileSystemInfos(lblFilePath.Content.ToString());
        foreach (FileSystemInfo info in infos)
        {
            if (info.Attributes.HasFlag(FileAttributes.Directory))
            {
                string subPath = remotePath + "/" + info.Name;
                if (!client.Exists(subPath))
                {
                    client.CreateDirectory(subPath);
                }
                UploadDirectory(client, info.FullName, remotePath + "/" + info.Name);
            }
            else
            {
                using (Stream fileStream = new FileStream(info.FullName, FileMode.Open))
                {
                    Console.WriteLine("Uploading {0} ({1:N0} bytes)", info.FullName, ((FileInfo)info).Length);

                    client.UploadFile(fileStream, remotePath + "/" + info.Name);
                }
            }
        }
        this.Close();
    }
    catch (Exception ex)
    {
        throw;
    }
}
```

```
1 reference
private void treeView_SelectedItemChanged(object sender, RoutedPropertyChangedEventArgs<object> e)
{
    selected_item = treeView.SelectedItem as TreeViewItem;
    lblFilePath.Content = selected_item.Header.ToString();
}
```

Şekil 13. Treeviewda Seçilenin Header İsmine Göre İşlemler Gerçekleştiriliyor

MESAJ GÖNDER EKRANI

```
0 references
public ObservableCollection<string> Ips { get; set; } = new ObservableCollection<string>()
{
        "192.168.139.129",
        "IP2",
        "IP3",
        "IP4"
};
3 references
public string SelectedIps
{
    get { return _sp; }
    set { _sp = value; NotifyPropertyChanged("SelectedIps"); }
}
3 references
public string PortSelect
{
    get { return _tp; }
    set { _tp = value; NotifyPropertyChanged("PortSelect"); }
}
3 references
public string MesajTextBox
{
    get { return _mesajTextBox; }
    set { _mesajTextBox = value; NotifyPropertyChanged("MesajTextBox"); }
}
0 references
public ICommand TcpbaglaButton
{
    get
    {
        _tcpBaglaButton = new RelayCommand(param => this.TcpBaglanButton());
        return _tcpBaglaButton;
    }
}
0 references
public ICommand MesajgonderButton
{
    get
    {
        _mesajGonderButton = new RelayCommand(param => this.MesajGonderButton());
        return _mesajGonderButton;
    }
}
```

Şekil 14. Button Command Ve Data Binding İşlemleri

```
1 reference
private void TcpBaglanButton()
{
    if (PortSelect == null || SelectedIps == null)
    {
        MessageBox.Show("Bilgileri Eksiksiz Giriniz!");
    }
    else
    {
        int PORT = Convert.ToInt32(PortSelect);
        string uzakBilgisayarIp = SelectedIps.ToString();

        try
        {
            this.soket.Connect(new IPEndPoint(IPAddress.Parse(uzakBilgisayarIp), PORT));
            MessageBox.Show("Başarıyla Bağlanıldı. Mesajınızı Girebilirsiniz.");
        }
        catch (Exception ex)
        {
            MessageBox.Show("\n (X) -> Bağlanmaya çalışırken hata oluştu: " + ex.Message);
            Application.Current.MainWindow.Close();
        }
    }
}
1 reference
private void MesajGonderButton()
{
    if (this.soket.Connected)
    {
        string gonder = MesajTextBox.ToString();
        this.soket.Send(Encoding.UTF8.GetBytes(gonder));
        MesajTextBox = "";
        if (gonder == "exit")
        {
            this.soket.Shutdown(SocketShutdown.Both);
            this.soket.Close();
            MessageBox.Show("Bağlantı Koparıldı.");
            MesajTextBox = "";
            PortSelect = "";
            SelectedIps = "";
            Application.Current.MainWindow.Close();
        }
    }
    else
        MessageBox.Show("Bağlantınız Yok");
}
```

Şekil 15. TCP Bağlantısı ve Mesaj Gönderme

TCP (**Transmission Control Protocol**) : Bilgisayarlar arasındaki iletişimin, küçük paketler hâlinde ve kayıpsız olarak gerçekleştirilmesini sağlayan bir protokoldür. Aslında TCP protokolünün en önemli özelliği veriyi karşı tarafa gönderirken veya alırken verinin bütünlüğünü sağlamasıdır. Projede kullanılan SSH ve SFTP protokoller TCP kullanır. Bu projede sunucu Linux tarafı olarak yapıldı çünkü mesajı gönderecek olan Windows bilgisayardır. C++ uygulaması sunucuyu başlatır ve listen() ile mesajı bekler.

```
public MesajGonderViewModel()
{
    Socket soket2 = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
    this.soket = soket2;
}
```

```
#include "TcpServer.hpp"

TcpServer::TcpServer(QObject *parent) : QObject(parent)
{
    connect(&_server, &QTcpServer::newConnection, this, &TcpServer::onNewConnection);
    connect(this, &TcpServer::newMessage, this, &TcpServer::onNewMessage);
    if(_server.listen(QHostAddress::Any, 45000)) {
        qInfo() << "İstemci Bekleniyor ...";
    }
}
```

Şekil 16. C++ Uygulaması Port Üzerinden Sunucuyu Başlatıp Dinlemeye Başlar


```
10  }
11
12  void TcpServer::onNewConnection()
13  {
14      const auto client = _server.nextPendingConnection();
15      if(client == nullptr) {
16          return;
17      }
18
19      qInfo() << "Yeni İstemci Bağlı.";
20
21      //_clients.insert(this->getClientKey(client), client);
22
23      connect(client, &QTcpSocket::readyRead, this, &TcpServer::onReadyRead);
24      connect(client, &QTcpSocket::disconnected, this, &TcpServer::onClientDisconnected);
25  }
26
27  void TcpServer::onReadyRead()
28  {
29      const auto client = qobject_cast<QTcpSocket*>(sender());
30
31      if(client == nullptr) {
32          return;
33      }
34
35      const auto message = "İstemci: " + client->readAll();
36
37      emit newMessage(message);
38  }
39
40  void TcpServer::onClientDisconnected()
41  {
42      const auto client = qobject_cast<QTcpSocket*>(sender());
43
44      if(client == nullptr) {
45          return;
46      }
47      const auto message = "İstemci Ayrıldı";
48      emit newMessage(message);
49      qInfo() << "İstemci Ayrıldı";
50      //_clients.remove(this->getClientKey(client));
51  }
52
53  void TcpServer::onNewMessage(const QByteArray &ba)
54  {
55      for(const auto &client : qAsConst(_clients)) {
56          client->write(ba);
57          client->flush();
58      }
59  }
60
```

WİNDOWS PERFORMANS EKRANI

```
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 MetroFramework.Forms;
using MetroFramework;

namespace MvvmDeneme
{
    5 references
    public partial class Manage : MetroFramework.Forms.MetroForm
    {
        1 reference
        public Manage()
        {
            InitializeComponent();
        }
        1 reference
        private void timer_Tick(object sender, EventArgs e)
        {
            float fcpu = pCPU.NextValue();
            float fram = pRAM.NextValue();
            metroProgressBar1CPU.Value = (int)fcpu;
            metroProgressBar2RAM.Value = (int)fram;
            lblCPU.Text = string.Format("{0:0.00}%", fcpu);
            lblRAM.Text = string.Format("{0:0.00}%", fram);
            chart1.Series["CPU"].Points.AddY(fcpu);
            chart1.Series["RAM"].Points.AddY(fram);
        }
        1 reference
        private void Manage_Load(object sender, EventArgs e)
        {
            timer.Start();
        }
    }
}
```

Şekil 17. WinForm Uygulaması

Wpf proje içerisinde winform çağırılarak oluşturulan ekrandır. Windows bilgisayarın Performansını göstermek için performanceCounter() ve timer() kullanılmıştır.

PerformanceCounter: sistem üzerinde çalışan tüm işlemlerin (process), işlemcinin (cpu), belleğin (ram), servislerin vs. durumunu öğrenebileceğimiz ve anlık olarak verilerini alabileceğimiz bileşendir.

Properties

pCPU System.Diagnostics.PerformanceCounter

(ApplicationSettings)	
(Name)	**pCPU**
CategoryName	**Processor**
CounterName	**% Processor Time**
GenerateMember	True
InstanceLifetime	Global
InstanceName	**_Total**
MachineName	.
Modifiers	Private
ReadOnly	True

Timer: Zamanlayıcı olarak kullanılan .NET komponentidir. Programlamada zamanlama ölçümü gerektiren her durumda kullanılabilmektedir. İstenen komutların belirlenen zaman aralığında tekrarlanmasını sağlamak için kullanılmaktadır. Tekrarlanmasını istediğimiz kodları **Timer_Tick** olayına yazarız. İnverval özelliği kodlarımızın ne kadar sürede tekrarlanacağını belirlediğimiz özelliktir. Milisaniye cinsinden değer verilir.

Properties

timer System.Windows.Forms.Timer

(ApplicationSettings)	
(Name)	**timer**
Enabled	False
GenerateMember	True
Interval	**200**
Modifiers	Private
Tag	

SANAL MAKİNE(LİNUX) PERFORMANS EKRANI

```
public partial class LinuxCpuMem : MetroFramework.Forms.MetroForm
{
    Thread t1;
    string result = string.Empty;
    List<DateTime> TimeList = new List<DateTime>();
    1 reference
    public LinuxCpuMem(SshClient sshClient)
    {
        InitializeComponent();
        t1 = new Thread(() => KomutCalistir(sshClient));
        t1.Start();
        myChart.ChartAreas[0].AxisX.LabelStyle.Format = "hh:mm";
        panelChart.AutoScroll = true;
    }
    1 reference
    public void KomutCalistir(SshClient sshClient)
    {
        while ((sshClient.IsConnected))
        {
            //Thread.Sleep(500);
            SshCommand cmd = sshClient.RunCommand("top -n 1 -b");
            result = cmd.Result;

            string cpuDegeri = ParseCpu(result);
            string cpu = cpuDegeri.Remove(0, 9);
            double cpuInt = Double.Parse(cpu, CultureInfo.InvariantCulture);

            int width = myChart.Width;
            int height = myChart.Height;

            DateTime now = DateTime.Now;
            TimeList.Add(now);
            myChart.Size = new Size(width++, height++);

            if (InvokeRequired)
            {
                BeginInvoke(new Action(() =>
                {
                    lblCpu.Text = cpuDegeri;
                    lblRam.Text = ParseMemory(result);
                    //myChart.ChartAreas[0].AxisX.Maximum = 100;
                    myChart.Series["CPU"].Points.AddXY(now, cpuInt/10);
                }));
            }
            else
            {
                t1.Abort();
            }
        }
    }
    1 reference
```

Şekil 18. Thread ile Komut Çalıştır Methodu

Ssh bağlantısı ile komut, thread ile çalıştırılarak CPU ve RAM bilgileri alınır.

WPFde ve Winformda temel olarak geri planda 2 thread bulunmakta; birisi rendering için diğeri UI'ı yönetmek için kullanılır. Rendering thread UI'a veri getirirken, ekrana çizim yaparken, eventler'i handle ederken devreye girer. UI Thread'i ise genellikle 1 tane olur, bazı durumlarda birden fazla da olabilir.

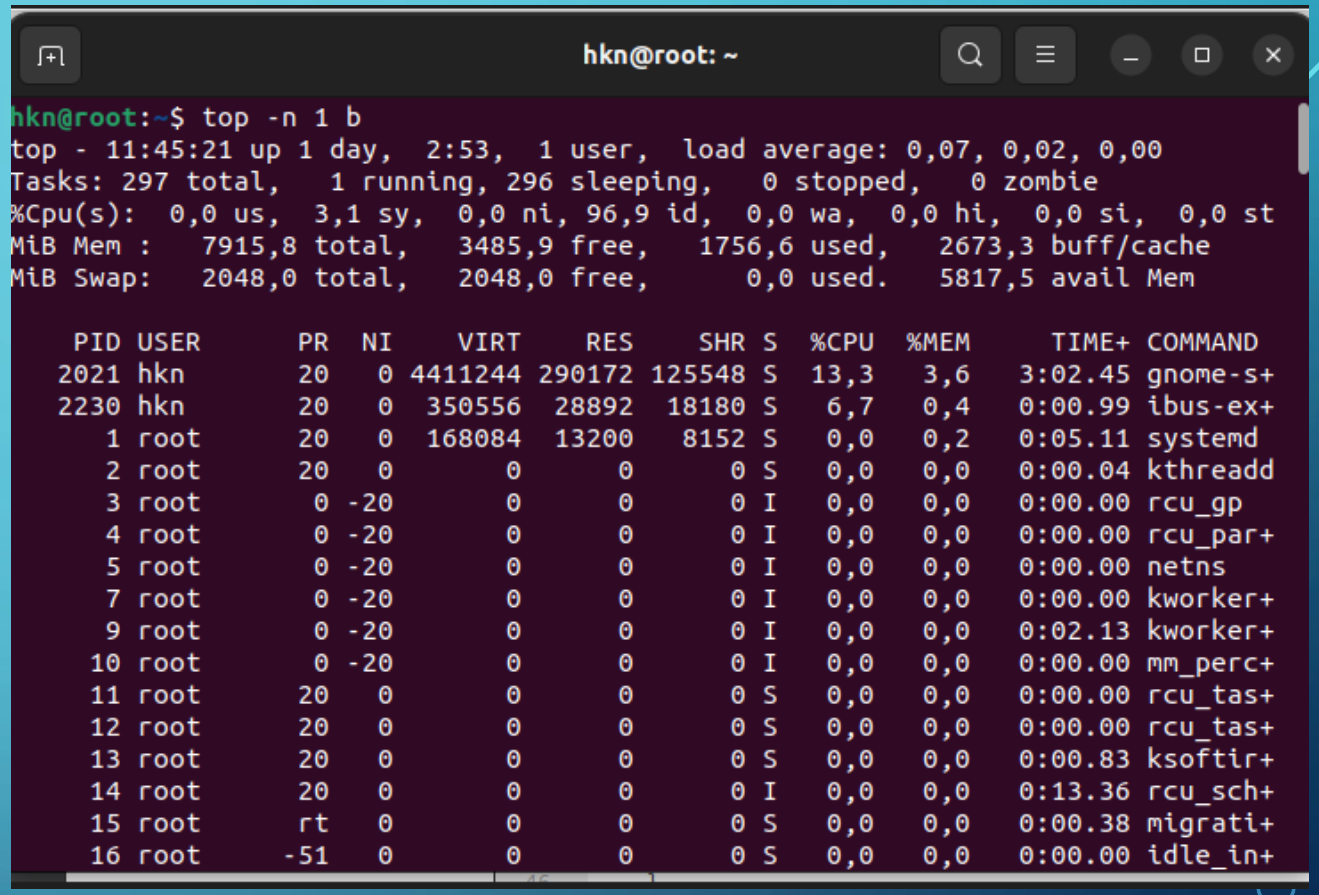
Yeni açılan ekranda grafik güncellemesi ve labellerin güncellenmesi için UI threadından farklı yeni bir thread açıldı. Dispatcher yani threadların senkron problemini çözmek için de winformda kullanılan InvokeRequired ve BeginInvoke kullanıldı label ve grafiğin sürekli güncellenmesi sağlandı.

InvokeRequired: Forma gelen talebin farklı bir iş parçacığından gelip gelmediğini kontrol eder.

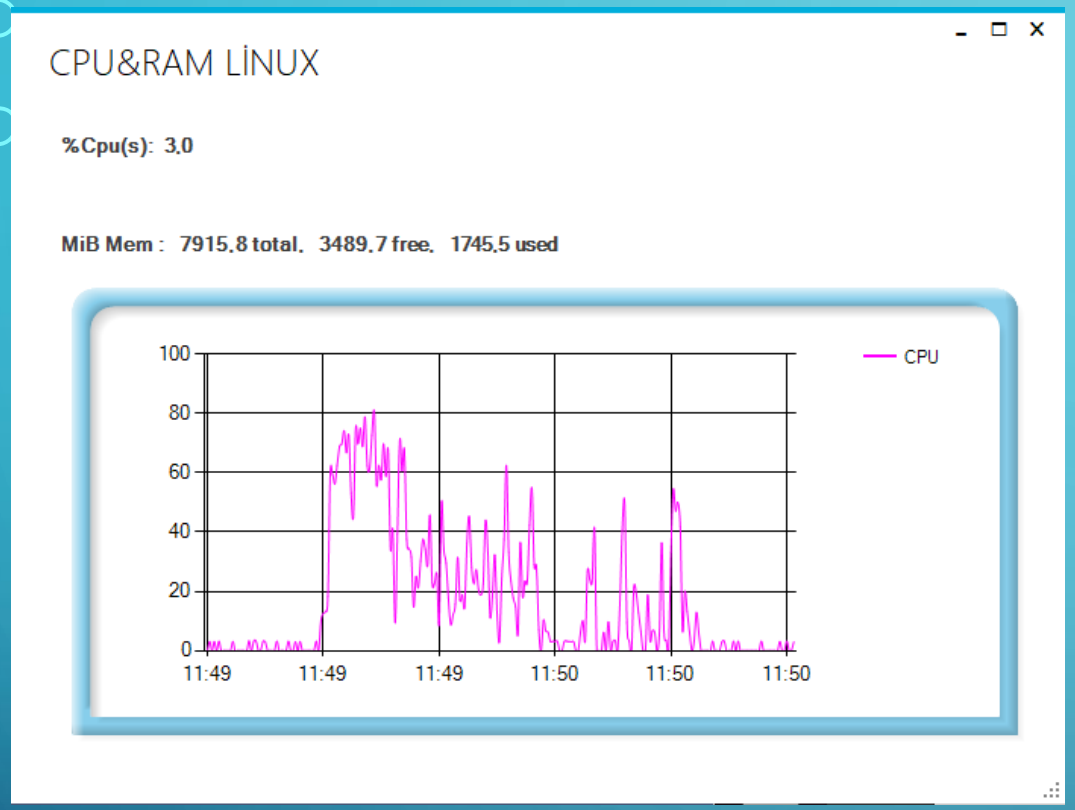
Eğer farklı bir iş parçacığından talep gelmişse aşağıdaki BeginInvoke metoduyla işlem gerçekleştiriliyor.

```
        public string ParseMemory(string contentOfOutput)
        {
            var line = contentOfOutput.Split('\n');
            string cpuLine;
            cpuLine = line[3];
            var wordCpu = cpuLine.Substring(0, 54);
            return wordCpu;
        }
        1 reference
        public string ParseCpu(string contentOfOutput)
        {
            var line = contentOfOutput.Split('\n');
            string cpuLine;
            cpuLine = line[2];
            var wordCpu = cpuLine.Substring(0, 13);
            return wordCpu;
        }
    }
```

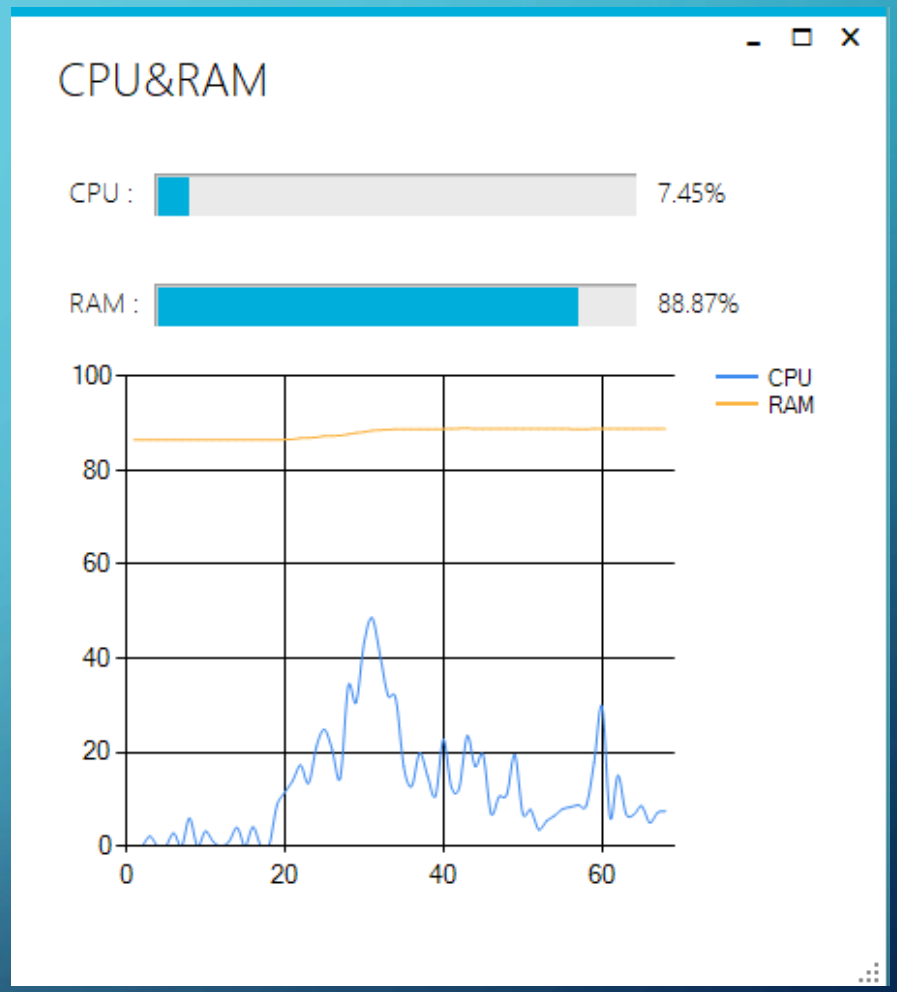
Şekil 19. Komutun Parse Edilmesi



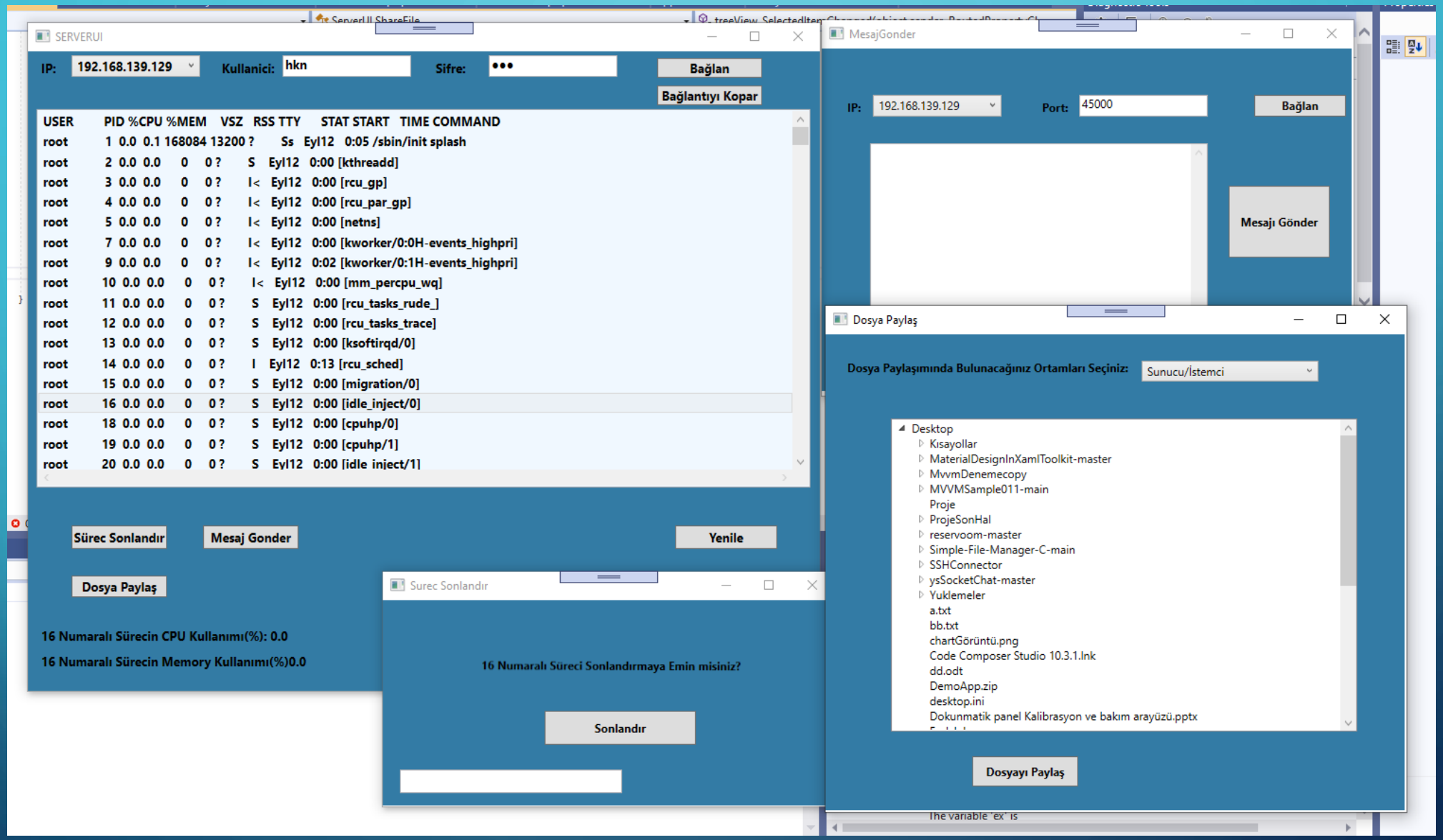
Şekil 20. top –n 1b Komutu



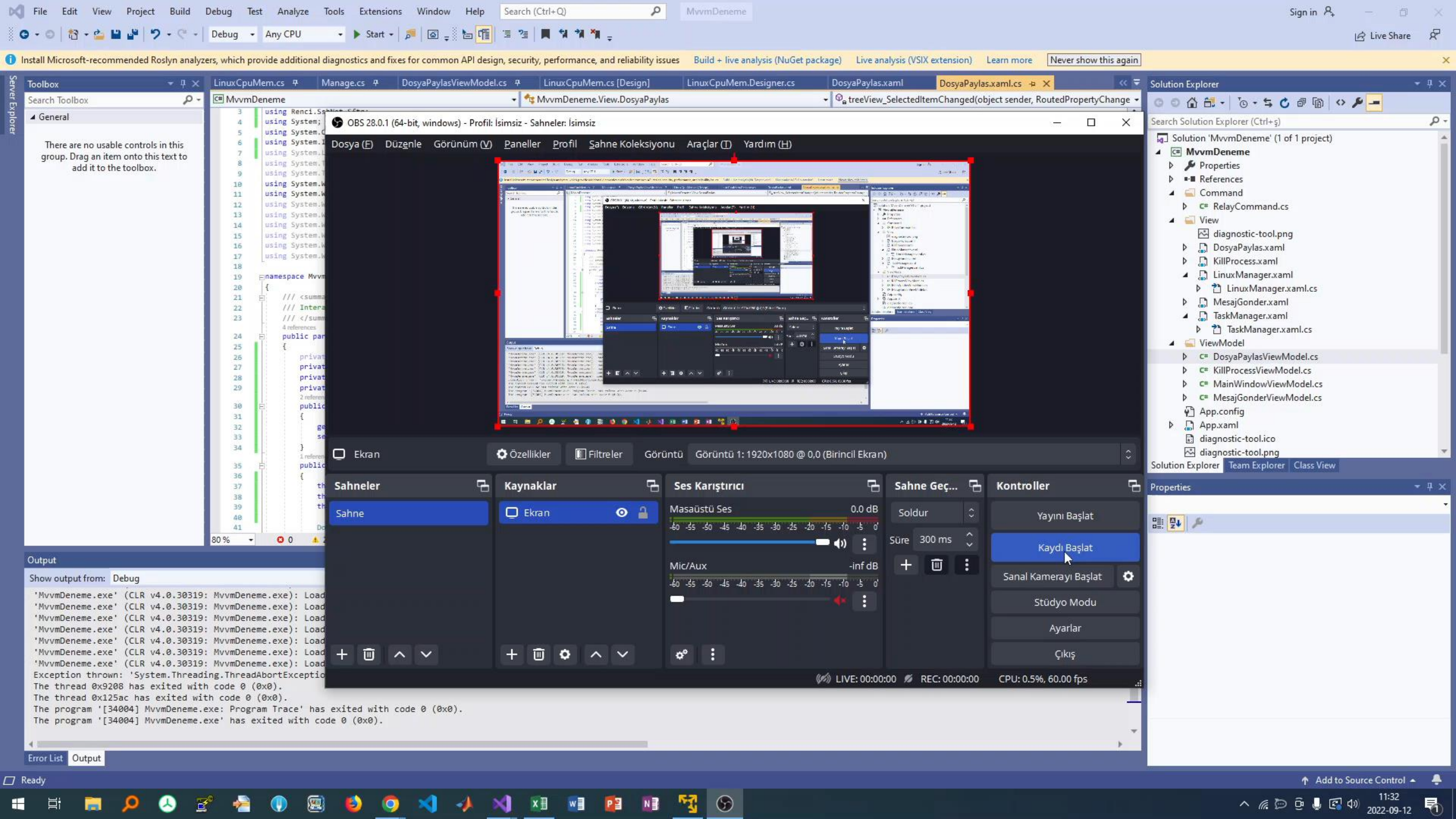
Şekil 21. Tablet Performans Grafiği

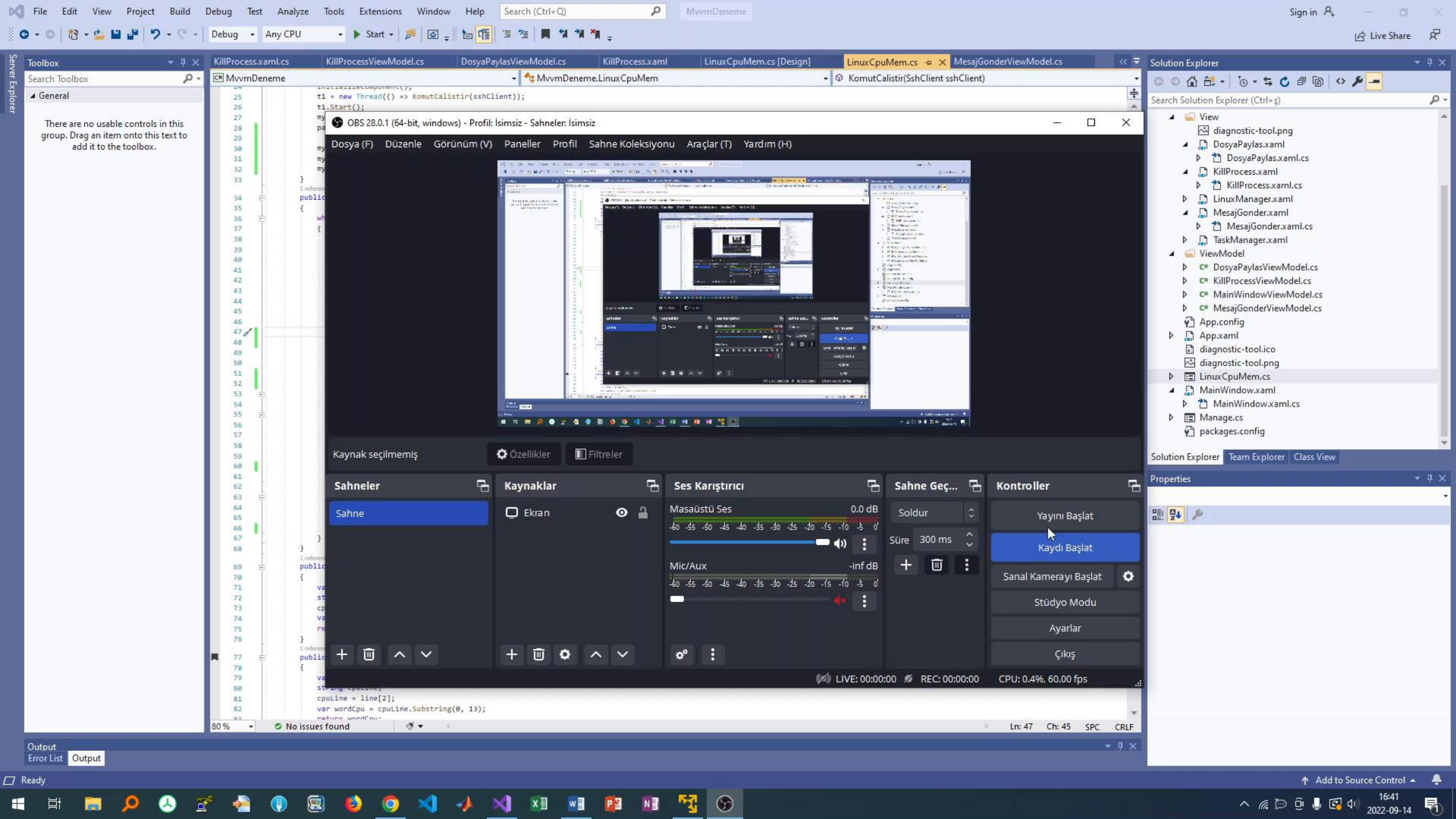


Şekil 22. Windows Performans Grafiği



Şekil 23. Proje İlk Hali


File Edit View Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
MvvmDeneme
Sign in
Live Share
Debug
Any CPU
Start
Install Microsoft-recommended Roslyn analyzers, which provide additional diagnostics and fixes for common API design, security, performance, and reliability issues
Build + live analysis (NuGet package)
Live analysis (VSIX extension)
Learn more
Never show this again
Toolbox
Search Toolbox
General
There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.
LinuxCpuMem.cs
Manage.cs
DosyaPaylasViewModel.cs
LinuxCpuMem.cs [Design]
LinuxCpuMem.Designer.cs
DosyaPaylas.xaml
DosyaPaylas.xaml.cs
MvvmDeneme
MvvmDeneme.View.DosyaPaylas
treeView_SelectedItemChanged(object sender, RoutedPropertyChange
OBS 28.0.1 (64-bit, windows) - Profil: İsimsiz - Sahneler: İsimsiz
Dosya (F) Düzenle Görünüm (V) Paneller Profil Sahne Koleksiyonu Araçlar (T) Yardım (H)
Ekran
Özellikler
Filtreler
Görüntü
Görüntü 1: 1920x1080 @ 0,0 (Birincil Ekran)
Sahneler
Sahne
Kaynaklar
Ekran
Ses Karıştırıcı
Masaüstü Ses
0.0 dB
Mic/Aux
-inf dB
Sahne Geç...
Soldur
Süre
300 ms
Kontroller
Yayını Başlat
Kaydı Başlat
Sanal Kamerayı Başlat
Stüdyo Modu
Ayarlar
Çıkış
LIVE: 00:00:00
REC: 00:00:00
CPU: 0.5%, 60.00 fps
Solution Explorer
Search Solution Explorer (Ctrl+ş)
Solution 'MvvmDeneme' (1 of 1 project)
MvvmDeneme
Properties
References
Command
RelayCommand.cs
View
diagnostic-tool.png
DosyaPaylas.xaml
KillProcess.xaml
LinuxManager.xaml
LinuxManager.xaml.cs
MesajGonder.xaml
TaskManager.xaml
TaskManager.xaml.cs
ViewModel
DosyaPaylasViewModel.cs
KillProcessViewModel.cs
MainWindowViewModel.cs
MesajGonderViewModel.cs
App.config
App.xaml
diagnostic-tool.ico
diagnostic-tool.png
Solution Explorer
Team Explorer
Class View
Properties
Output
Show output from: Debug
'MvvmDeneme.exe' (CLR v4.0.30319: MvvmDeneme.exe): Load
Exception thrown: 'System.Threading.ThreadAbortExceptio
The thread 0x9208 has exited with code 0 (0x0).
The thread 0x125ac has exited with code 0 (0x0).
The program '[34004] MvvmDeneme.exe: Program Trace' has exited with code 0 (0x0).
The program '[34004] MvvmDeneme.exe' has exited with code 0 (0x0).
Error List
Output
Ready
Add to Source Control
11:32
2022-09-12


File Edit View Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
MvvmDeneme
Sign in
Live Share
Debug
Any CPU
Start
Toolbox
Search Toolbox
General
There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.
KillProcess.xaml.cs
KillProcessViewModel.cs
DosyaPaylasViewModel.cs
KillProcess.xaml
LinuxCpuMem.cs [Design]
LinuxCpuMem.cs
MesajGonderViewModel.cs
MvvmDeneme
MvvmDeneme.LinuxCpuMem
KomutCalistir(SshClient sshClient)
t1 = new Thread(() => KomutCalistir(sshClient));
t1.Start();
OBS 28.0.1 (64-bit, windows) - Profil: İsimsiz - Sahneler: İsimsiz
Dosya (F) Düzenle Görünüm (V) Paneller Profil Sahne Koleksiyonu Araçlar (T) Yardım (H)
Kaynak seçilmemiş
Özellikler
Filtreler
Sahneler
Sahne
Kaynaklar
Ekran
Ses Karıştırıcı
Masaüstü Ses
0.0 dB
Mic/Aux
-inf dB
Sahne Geç...
Soldur
Süre
300 ms
Kontroller
Yayını Başlat
Kaydı Başlat
Sanal Kamerayı Başlat
Stüdyo Modu
Ayarlar
Çıkış
LIVE: 00:00:00
REC: 00:00:00
CPU: 0.4%, 60.00 fps
cpuLine = line[2];
var wordCpu = cpuLine.Substring(0, 13);
80 %
No issues found
Ln: 47
Ch: 45
SPC
CRLF
Solution Explorer
Search Solution Explorer (Ctrl+ş)
View
diagnostic-tool.png
DosyaPaylas.xaml
DosyaPaylas.xaml.cs
KillProcess.xaml
KillProcess.xaml.cs
LinuxManager.xaml
MesajGonder.xaml
MesajGonder.xaml.cs
TaskManager.xaml
ViewModel
DosyaPaylasViewModel.cs
KillProcessViewModel.cs
MainWindowViewModel.cs
MesajGonderViewModel.cs
App.config
App.xaml
diagnostic-tool.ico
diagnostic-tool.png
LinuxCpuMem.cs
MainWindow.xaml
MainWindow.xaml.cs
Manage.cs
packages.config
Solution Explorer
Team Explorer
Class View
Properties
Output
Error List
Output
Ready
Add to Source Control
16:41
2022-09-14

TEŞEKKÜRLER