ArkSigner

VIA

Entegrasyon Dökümanı

ArkSigner

VIA C#

Entegrasyon Dökümanı

Yayın Tarihi: 20.01.2020

Amaç

Bu dokümanda, Asp.Net ve C# teknolojileri ile geliştirilmiş uygulamaların, VIA ile entegrasyonu anlatılmaktadır. Entegrasyon örneklerinde bu teknoloji kullanılacaktır.

Bir uygulamanın sisteme entegre olabilmesi için, uygulamanın VIA portaline erişebilir özellikte olması gerekmektedir. Bu doküman, belirtilen bu konuların çözüldüğünü farz etmektedir.

Not:

VIA portal sistemi üzerinden imzalama yapılabilmesi için son kullanıcı bilgisayarında ArkSigner Client uygulamasının kurulu olması gerekmektedir.

Uygulamaya https://www.arksigner.com/indir URL adresinden ulaşabilirsiniz.

ArkSigner

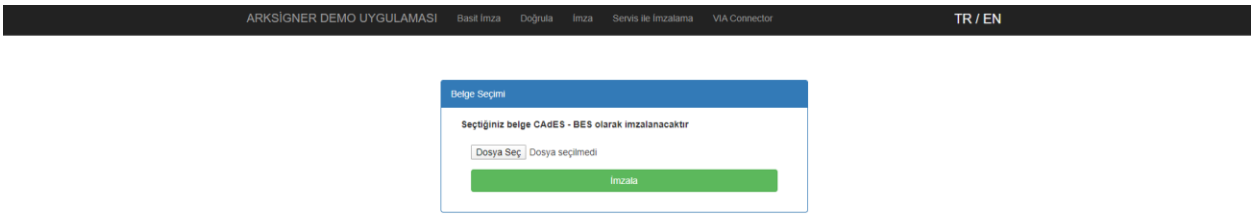
VIA C#

Entegrasyon Dökümanı

Yayın Tarihi: 20.01.2020

Entegrasyon Adımları

Örnek Client Uygulama Ekranı



Şekil 1

Şekil 1 de gösterilen Demo VIA Client Uygulaması, imzalanacak olan dosyanın seçildiği ekrandır. Örnek kod bloklarına bu doküman içerisinden ya da proje içerisinden ulaşabilirsiniz.

Menülerde bulunan seçenekler:

- **Basit İmza**: En sade ve basit imzalama ekranıdır. Belge seçilir ve imzala butonu ile Basic Controller içerisinde bulunan Basic methodu çağrılır. Method içerisinde imza parametreleri tanımlanmıştır. İmza tipi CAdES – BES olarak tanımlanmıştır.

- **Doğrula**: İmzalı bir belgenin doğrulanması için kullanılır. İmzalı belge seçilerek 'Doğrula' butonu ile sistem belgenin doğrulamasını yapar. İmzanın geçerlilik süresi ve kim tarafından imzalandığı bilgileri doğrulama servisleri tarafından client uygulamasına iletilir.
- **İmza:** Seçilen belgelerin imza tipleri, tümleşik- ayrık imza bilgileri ve imzalayacak olan kullanıcının kimlik numaraları tanımlanarak VIA portal sistemi ile imzalama işlemi yapılır. Bu ekranda seçilen belgeler VIA portal uygulamasına HttpWebRequest yöntemi ile iletilir.
- **Servis ile İmzalama:** Seçilen belgelerin imza tipleri, tümleşik- ayrık imza bilgileri ve imzalayacak olan kullanıcının kimlik numaraları tanımlanarak VIA portal sistemi ile imzalama işlemi yapılır. Bu ekranda seçilen belgeler VIA portal uygulamasına VIA Servis aracılığı ile iletilir.
- **VIA Connector:** Seçilen belgelerin imza tipleri, tümleşik- ayrık imza bilgileri ve imzalayacak olan kullanıcının kimlik numaraları tanımlanarak VIA portal sistemi ile imzalama işlemi yapılır. Bu ekranda seçilen belgeler VIA portal uygulamasına VIA Connector adında metod ile iletilir. Entegrasyon kısmında basit ve kullanışlı olan metod olarak öngörülür.

ArkSigner

VIA C#

Entegrasyon Dökümanı

Yayın Tarihi: 20.01.2020

VIA Portali ile Entegrasyonun Sağlanması

Sisteminizin VIA Portali ile entegre bir şekilde çalışabilmesi için belirlemiş olduğunuz endpoint'in Request taleplerine cevap vermiş olması gerekmektedir. Web Config'e eklenmesi gerekenler

<appSettings>

```
1.    <add key="ServiceAddress" value="http://VIAPortalURLAdresi" />
2.    <add key="AppAddress" value="http://SİzinsisteminizinURLAdresi " />
3.    <add key="WebApiAddress" value="http://VIAPortalDogrulamaServisiURLadresi" />
4.    <add key="AppId" value="VIA Portalinde tanımlanan uygulama ID si" />
5.    <add key="AppPass" value="VIA Portalinde tanımlanan uygulamaya ait şifre bilgisi" />
```

</appSettings>

VIA sisteminin kullanılması için gerekli olan konfigürasyon ayarları yapılır. Seçilen dosyanın portale gönderilmesi 3 farklı seçenek ile yapılmaktadır.

- VIA Connector ile Entegrasyon
- VIA Servis ile Entegrasyon
- HttpWebRequest yöntemi ile Entegrasyon

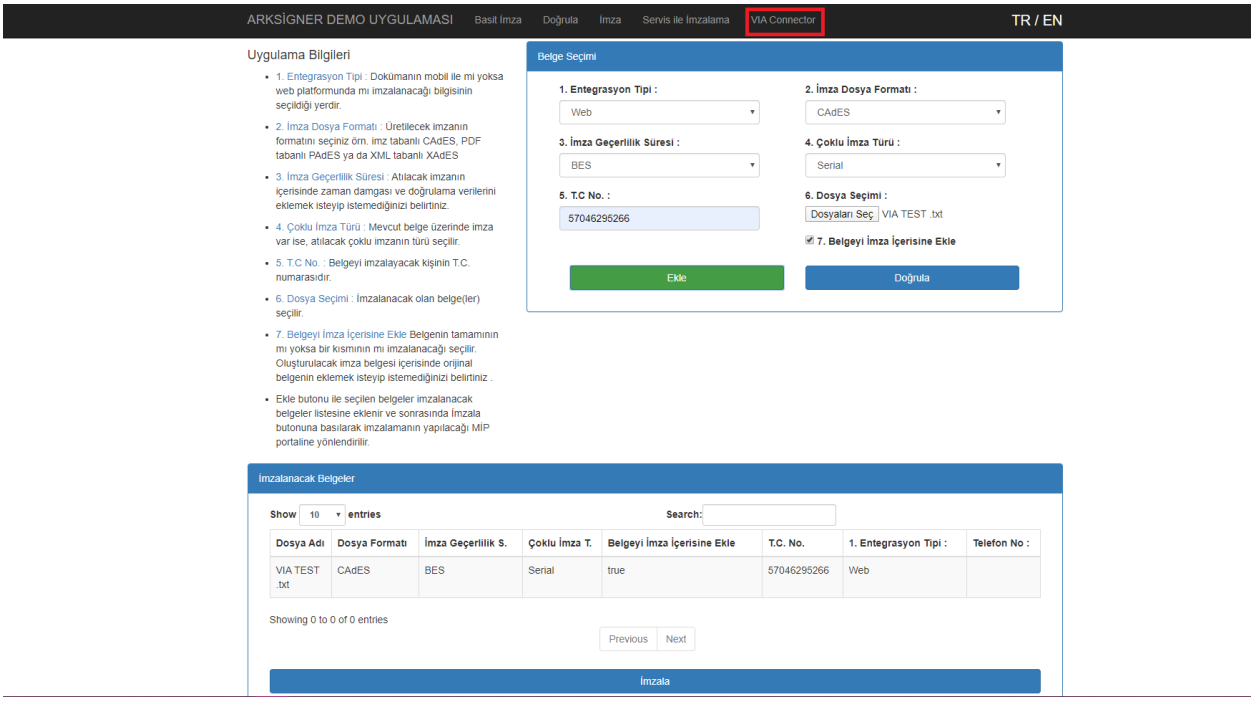
Yukarıda verilen web config ayarları için VIA Portaline erişecek olan uygulamanın tanımlanmış olması gerekmektedir. Portal, tanımlanan uygulama için bir uygulama Id (AppId) ve uygulama şifresi(AppPass) üretecektir.

VIA Connector ile Entegrasyon

Adım 1 :

Bu aşamada imzalanacak doküman VIA Service kullanılarak VIA portaline gönderilir. İmzalama işlemi **Adım 2** sayfasında anlatılacaktır. İmzalama işlemi tamamlandıktan sonra VIA tarafından ilgili belgenin indirilmesi için gerekli Url adresleri tarafınıza iletilir. İmzalanan belgenin VIA portal sisteminden alınması için gerekli işlemler **Adım 3** sayfasında tanımlanmıştır.

Not: VIAConnector metodunun kullanılabilmesi için VIA Servis uygulamasının ulaşılabilir olması ve projenize VIA_CONNECTOR.dll kütüphanesinin referans olarak eklenmesi gerekmektedir.



Şekil 2

İmzala butonu ile VIAConnector Controller içerisindeki SignWithVIAConnector metodu tetiklenir.

```
1.  //VIA Connector Kullanarak imzalama işlemi
2.  [HttpPost]
3.  public async Task<ActionResult> SignWithVIAConnector(List<SignListClass> Signs)
4.  {
5.   // iletilecek olan dosyaların toplanması
6.   // dosyaların REST bir şekilde sunucuya iletilmesi
7.   // Browser'a 302'nin gönderilmesi
8.   Dictionary<string, object> responseController = new Dictionary<string, object>();
9.   if (!ModelState.IsValid)
10.  {
11.      TempData["ErrorMsg"] = "Not all fields are filled";
12.  }
13.  try
14.  {
15.    string customKey = "5704629526666";
```

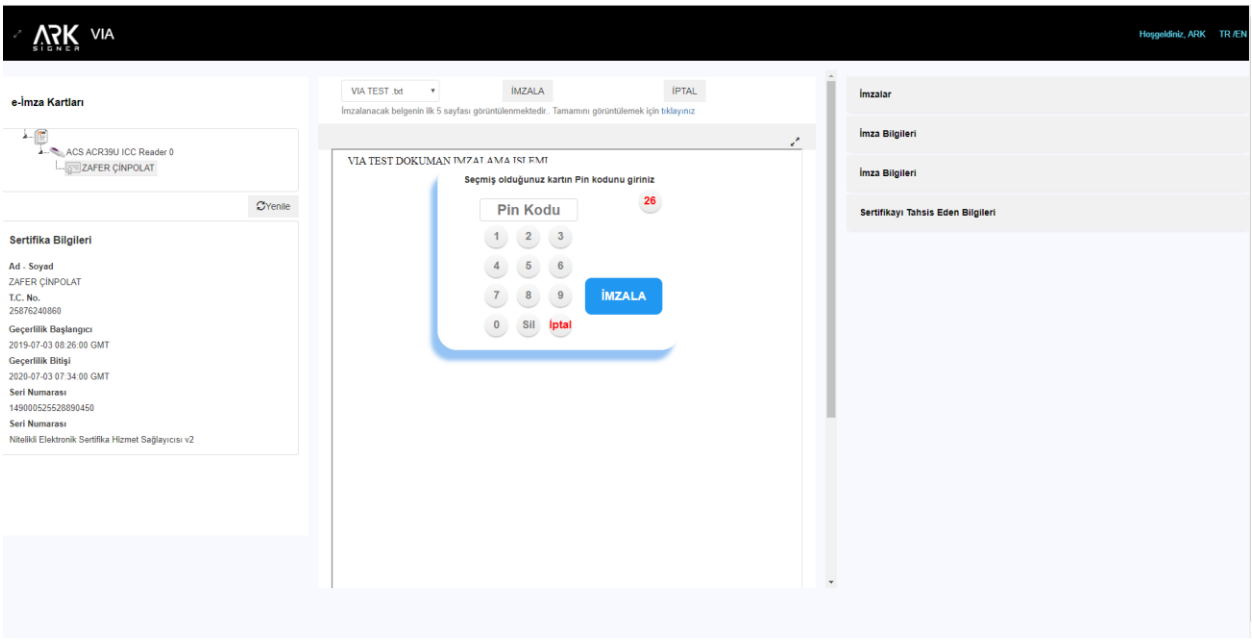
```
16.    string lang = "tr-Tr"; //* Set service language for request message
17.    HttpCookie cookie = HttpContext.Request.Cookies["lang"];
18.
19.    if (cookie != null && !string.IsNullOrEmpty(cookie["lang"]))
20.        lang = cookie["lang"];
21.
22.
23.    for (int i = 0; i < Signs.Count; i++)
24.        Signs[i].customKey = customKey;
25.
26.    SignModel signModel = new SignModel();
27.    signModel.pass = Config.AppPass;
28.    signModel.signDocuments = Signs;
29.    signModel.callbackUrl = Config.AppAddress + Url.Action("GetStatus","Home");
30.    signModel.cancelUrl = Config.AppAddress + "/Home/WebApiService";
31.    signModel.appId = Config.AppId;
32.    signModel.serviceAddress = Config.ServiceAddress;
33.    signModel.lang = lang;
34.    signModel.connectorServiceAddress = Config.WebApiAddress;
35.
36.    VIA_CONNECTOR.ArkSignerVIAConnector connector = new VIA_CONNECTOR.ArkSignerVIAConn
   ector();
37.    var jsonTask = connector.CreateSigningTask(signModel);
38.    jsonTask.Wait();
39.    string res = jsonTask.Result.ToString();
40.    Dictionary<string, object> apires = JsonConvert.DeserializeObject<Dictionary<string,
    object>>(res);
41.
42.    if ((bool)apires["success"] == true)
43.    {
44.        ResponseDocument responseDocument = new ResponseDocument
45.        {
46.            url = Config.ServiceAddress + "/" + apires["url"] + ".aspx",
47.            transactionUUID = apires["transactionUUID"].ToString(),
48.        };
49.        Session["responseDocument"] = responseDocument;
50.        responseController["success"] = "true";
51.        responseController["url"] = "/Response/Index";
52.    }
53.    else
54.    {
55.        responseController["success"] = "false";
56.        responseController["message"] = apires["message"].ToString();
57.    }
58.  }
59.  catch (Exception ex)
60.  {
61.      responseController["message"] = MerkeziImzaClient.Language.Resource.error;
62.  }
63.  return Json(responseController);
64. }
```

Response Controller içerisinde bulunan Index metoduna ait kod bilgileri aşağıdaki gibidir:

```
1.    // GET: Response
2.  public ActionResult Index()
3.  {
4.
5.   try
6.   {
7.    ResponseDocument responseDocument = Session["responseDocument"] as ResponseDocument;

8.    if (responseDocument != null)
9.    {
10.       Response.Clear();
11.       var sb = new System.Text.StringBuilder();
12.       sb.Append("<html>");
13.       sb.AppendFormat("<body onload='document.forms[0].submit()'>");
14.       sb.AppendFormat("<form action='{0}' method='post'>", responseDocument.url);
15.       sb.AppendFormat("<input type='hidden' name='documentUUID' value='{0}'>", response
    Document.transactionUUID);
16.       sb.Append("</form>");
17.       sb.Append("</body>");
18.       sb.Append("</html>");
19.
20.       Response.Write(sb.ToString());
21.       Response.End();
22.       return new RedirectResult("responseDocument.url", false);
23.   }
24.   else
25.   {
26.       return View();
27.   }
28.  }
29.  catch (Exception)
30.  {
31.      throw;
32.  }
33. }
34.
```

Adım 2:

VIA portaline gönderilen belgeler VIA İmzalama ekranı üzerinden İMZALA butonu ile imzalanır. IPTAL butonu ise işlemin iptal edilmesini ve CancelURL kısmında tanımlanan URL adresine sistemin yönlendirilmesini sağlar.



Şekil 3

İMZALA butonuna basıldıktan sonra Şekil 3 de görülen pin giriş ekranı çıkar ve pin girişi yapıldıktan sonra imzala butonuna basılır ve imzalama işlemi gerçekleştirilir. VIA portalinde imzalama işlemini tamamladıktan sonra otomatik olarak ana uygulamaya tekrar dönülür.

Adım 3:

Adım 2 aşamasında belge imzalanır ve imzalı belge VIA veri tabanına kaydedilir. Bu aşamada imzalı verinin VIA portalinde bulunan servis tarafından nasıl alınacağı anlatılmıştır.

Adım 1 de tanımlanmış olan aşağıdaki kod bloğu imzalama sonrasında VIA portalinin tekrar hangi metoda döneceğini belirtmektedir.

```
signModel.callbackUrl = Config.AppAddress + Url.Action("GetStatus", "Home"); // Imzalama işlem
i sonrasında portal sisteminin tekrar hangi Url adresine dönüş yapacağı tanımlanır.
```

Yukarıdaki örnekte imzalama sonrasında Home Controller altında bulunan GetStatus metodu çağrılacak şekilde tanımlamalar yapılmıştır. GetStatus metoduna ait kod bloğu:

```
1.  [HttpGet]
2.  public ActionResult GetStatus(bool success, string downloadUrl, string downloadAction,
    string documentUUID, string downloadParameter)
3.  {
4.      string url = "?action=" + downloadAction + "&downloadParameter=" + downloadParamete
    r + "&documentUUID=" + documentUUID + "&downloadUrl=" + downloadUrl;
5.      ViewBag.Url = url;
6.
7.      return View();
8.  }
```

Yukarıdaki kod bloğundan indirme linkine ait veriler bulunmaktadır. GetStatus ekranında bulunan “İNDİR” butonu ile imzalanmış olan belgeler VIA portalinden alınır. (Home Controller içinde bulunan DownloadSignedFile adlı metod)

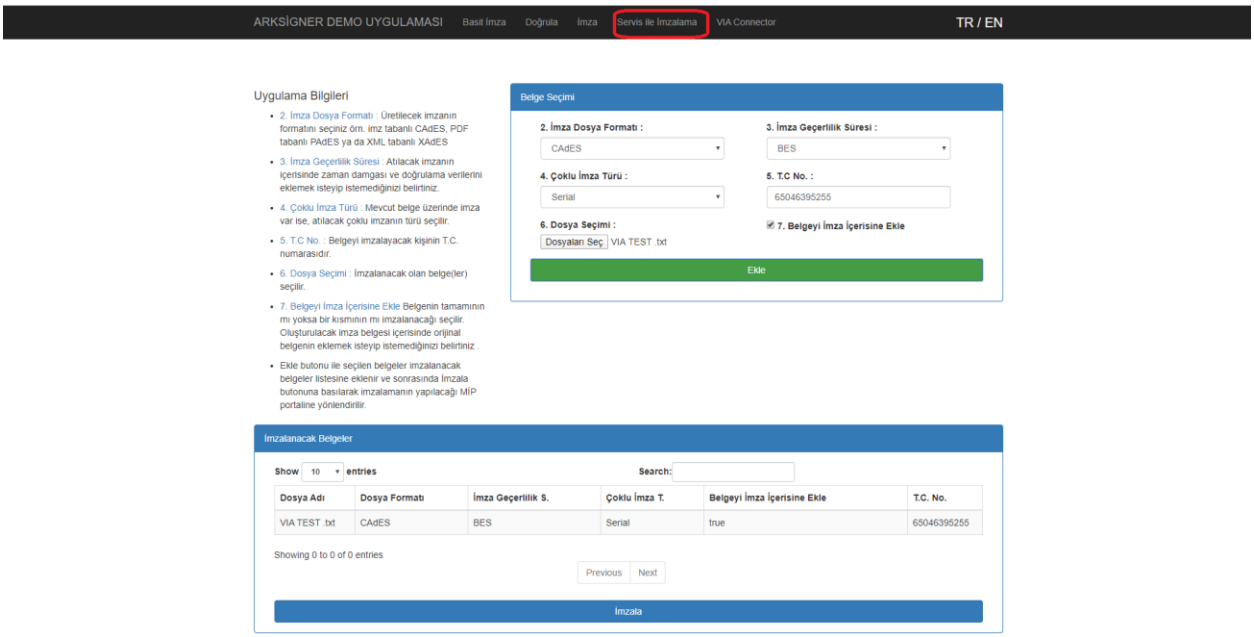
```
1.  [HttpGet]
2.  public ActionResult DownloadSignedFile()
3.  {
4.   try
5.   {
6.     byte[] complatedData = null;
7.     string downloadUrl = Request.Form["downloadUrl"];
```

```
8.      if (String.IsNullOrEmpty(downloadUrl))
9.      {
10.         downloadUrl = Request.Params["downloadUrl"];
11.     }
12.     string documentUUID = Request.Form["documentUUID"];
13.     if (String.IsNullOrEmpty(documentUUID))
14.     {
15.         documentUUID = Request.Params["documentUUID"];
16.     }
17.     string downloadAction = Request.Form["action"];
18.     if (String.IsNullOrEmpty(downloadAction))
19.     {
20.         downloadAction = Request.Params["action"];
21.     }
22.
23.     string url = Config.ServiceAddress + "/" + downloadUrl + "?action=" + downloadAction
    ;
24.     string myParameters = "documentUUID=" + documentUUID + "";
25.
26.     using (WebClient wc = new WebClient())
27.     {
28.        wc.Headers[HttpRequestHeader.ContentType] = "application/x-www-form-
    urlencoded";
29.        wc.Encoding = System.Text.Encoding.UTF8;
30.        string HtmlResult = wc.UploadString(url, myParameters);
31.
32.        List<GetSign> responseJson = JsonConvert.DeserializeObject<List<GetSign>>(HtmlRes
    ult);
33.        string configPath = MvcApplication.SignPath + "\\" + DateTime.Now.ToString("yyyy-
    MM-dd") + "\\";
34.
35.        if (!System.IO.Directory.Exists(configPath))
36.        {
37.            System.IO.Directory.CreateDirectory(configPath);
38.        }
39.
40.        using (MemoryStream memoryStream = new MemoryStream())
41.        {
42.         using (var archive = new ZipArchive(memoryStream, ZipArchiveMode.Create, true))

43.         {
44.          foreach (GetSign item in responseJson)
45.          {
46.            //to do :  Dosyanın cades mi pades mi xades mi bilgisinin de geçilmesi lazım

47.            complatedData = Convert.FromBase64String(item.base64FileData);
48.            string extension = string.Empty;
49.            extension = item.FileName.Split('.')[1];
50.            if (item.SignDesc == "CAdES")
51.            {
52.                extension = "imz";
```

```
            }

            if (responseJson.Count > 1)
            {
                string statusMessage = "";
                if (item.success == false) //Eksik İmzalı Dokümanalar.
                    statusMessage = "_Eksik_İmza_";

                System.IO.File.WriteAllBytes(configPath + "\\" + Guid.NewGuid().ToString(
    ) + item.customKey + statusMessage + "." + extension + "", Convert.FromBase64String(ite
    m.base64FileData));
                var file = archive.CreateEntry(Guid.NewGuid().ToString() + item.customKey
     + "." + extension + "", CompressionLevel.NoCompression);
                using (var fileStream = file.Open())
                {
                    fileStream.Write(complatedData, 0, complatedData.Length);
                }
            }
            else
            {
                string statusMessage = "";
                if (item.success == false) //Eksik İmzalı Dokümanalar.
                    statusMessage = "_Eksik_İmza_";
                Response.Clear();
                Response.AddHeader("Cache-Control", "no-cache, must-revalidate, post-
    check=0, pre-check=0");
                Response.AddHeader("Pragma", "no-cache");
                Response.AddHeader("Content-Description", "Indirme");
                Response.AddHeader("Content-
    Type", "application/" + Path.GetExtension(item.FileName).Substring(1));
                Response.AddHeader("Content-Transfer-Encoding", "binary\n");
                Response.AddHeader("content-
    disposition", "attachment;filename="+ Guid.NewGuid().ToString() + item.customKey + stat
    usMessage + "." + extension + "");
                Response.BinaryWrite(complatedData);
                Response.Flush();
                Response.Close();
                Response.End();
                return View();
            }

         }
        }
        memoryStream.Position = 0;
        ViewBag.Path = MvcApplication.SignPath;
        return File(memoryStream.ToArray(), "application/octet", "signed.zip");
       }
    }
  }
  catch (Exception)
  {
```

```
97.       return Redirect("/Home/Index");
98.   }
99. }
```

Vıa Servis ile Entegrasyon



Şekil 4

Adım 1 :

Bu aşamada imzalanacak doküman VIA Service kullanılarak VIA portaline gönderilir. İmzalama işlemi **Adım 2** sayfasında anlatılacaktır. İmzalama işlemi tamamlandıktan sonra VIA tarafından ilgili belgenin indirilmesi için gerekli Url adresleri tarafınıza iletilir. İmzalanan belgenin VIA portal sisteminden alınması için gerekli işlemler **Adım 3** sayfasında tanımlanmıştır.

İmzala butonu ile VIAService Controller içerisindeki WebAPIService metodu tetiklenir.

```
1.  [HttpPost]
2.  public async Task<ActionResult> WebApiService(List<SignListClass> Signs)
3.  {
4.      // iletilecek olan dosyaların toplanması
5.      // dosyaların REST bir şekilde sunucuya iletilmesi
6.      // Browser'a 302'nin gönderilmesi
7.   Dictionary<string, object> responseController = new Dictionary<string, object>();
8.   if (!ModelState.IsValid)
9.   {
10.      TempData["ErrorMsg"] = "Not all fields are filled";
11.  }
12.  try
13.  {
14.   string customKey = "57029526666";
15.   string lang = "tr-Tr"; //* Set service language for request message
16.   HttpCookie cookie = HttpContext.Request.Cookies["lang"];
17.
18.   if (cookie != null && !string.IsNullOrEmpty(cookie["lang"]))
19.       lang = cookie["lang"];
20.
21.
22.   for (int i = 0; i < Signs.Count; i++)
23.       Signs[i].customKey = customKey;
24.
25.   SignModel signModel = new SignModel();
26.   signModel.pass = Config.AppPass;
27.   signModel.signDocuments = Signs;
28.   signModel.callbackUrl = Config.AppAddress + Url.Action("GetStatus","Home");
29.   signModel.cancelUrl = Config.AppAddress + "/Home/WebApiService";
30.   signModel.appId = Config.AppId;
31.   signModel.serviceAddress = Config.ServiceAddress;
32.   signModel.lang = lang;
33.
34.
35.   using (var client = new HttpClient())
36.   {
37.    client.BaseAddress = new Uri(Config.WebApiAddress);
38.    client.DefaultRequestHeaders.Accept.Clear();
39.    client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("applica
    tion/json"));
40.    var uri = Config.WebApiAddress + "/CreateSigningTask";
41.    var response = await client.PostAsJsonAsync(uri, signModel).ConfigureAwait(false); /
    / PostAsync(uri, null)
42.    var result = await response.Content.ReadAsStringAsync();
43.    var jsonTask = response.Content.ReadAsAsync<Dictionary<string, object>>();
44.    jsonTask.Wait();
45.    Dictionary<string, object> apires = jsonTask.Result;
```

```
46.
47.    if ((bool)apires["success"] == true)
48.    {
49.      ResponseDocument responseDocument = new ResponseDocument
50.      {
51.          url = Config.ServiceAddress + "/" + apires["url"] + ".aspx",
52.          transactionUUID = apires["transactionUUID"].ToString(),
53.      };
54.      Session["responseDocument"] = responseDocument;
55.      responseController["success"] = "true";
56.      responseController["url"] = "/Response/Index";
57.    }
58.    else
59.    {
60.        responseController["success"] = "false";
61.        responseController["message"] = apires["message"].ToString();
62.    }
63.   }
64.  }
65.  catch (Exception ex)
66.  {
67.      responseController["message"] = MerkeziImzaClient.Language.Resource.error;
68.  }
69.  return Json(responseController);
70. }
```

Response Controller içerisinde bulunan Index metoduna ait kod bilgileri şu şekildedir.

```
35.   // GET: Response
36. public ActionResult Index()
37. {
38.
39.  try
40.  {
41.   ResponseDocument responseDocument = Session["responseDocument"] as ResponseDocument;

42.   if (responseDocument != null)
43.   {
44.       Response.Clear();
45.       var sb = new System.Text.StringBuilder();
46.       sb.Append("<html>");
47.       sb.AppendFormat("<body onload='document.forms[0].submit()'>");
48.       sb.AppendFormat("<form action='{0}' method='post'>", responseDocument.url);
49.       sb.AppendFormat("<input type='hidden' name='documentUUID' value='{0}'>", response
    Document.transactionUUID);
50.       sb.Append("</form>");
51.       sb.Append("</body>");
52.       sb.Append("</html>");
53.
```

```
54.         Response.Write(sb.ToString());
55.         Response.End();
56.         return new RedirectResult("responseDocument.url", false);
57.     }
58.     else
59.     {
60.         return View();
61.     }
62.   }
63.   catch (Exception)
64.   {
65.       throw;
66.   }
67. }
```

Bu işlemlerden sonra VIA Portaline yönlendirme yapılır. Sonrasında ise **"VIA Connector ile Entegrasyon"** kısmında anlatılan **Adım 2** ve **Adım 3** kısımları ile işlemlere devam edilir.

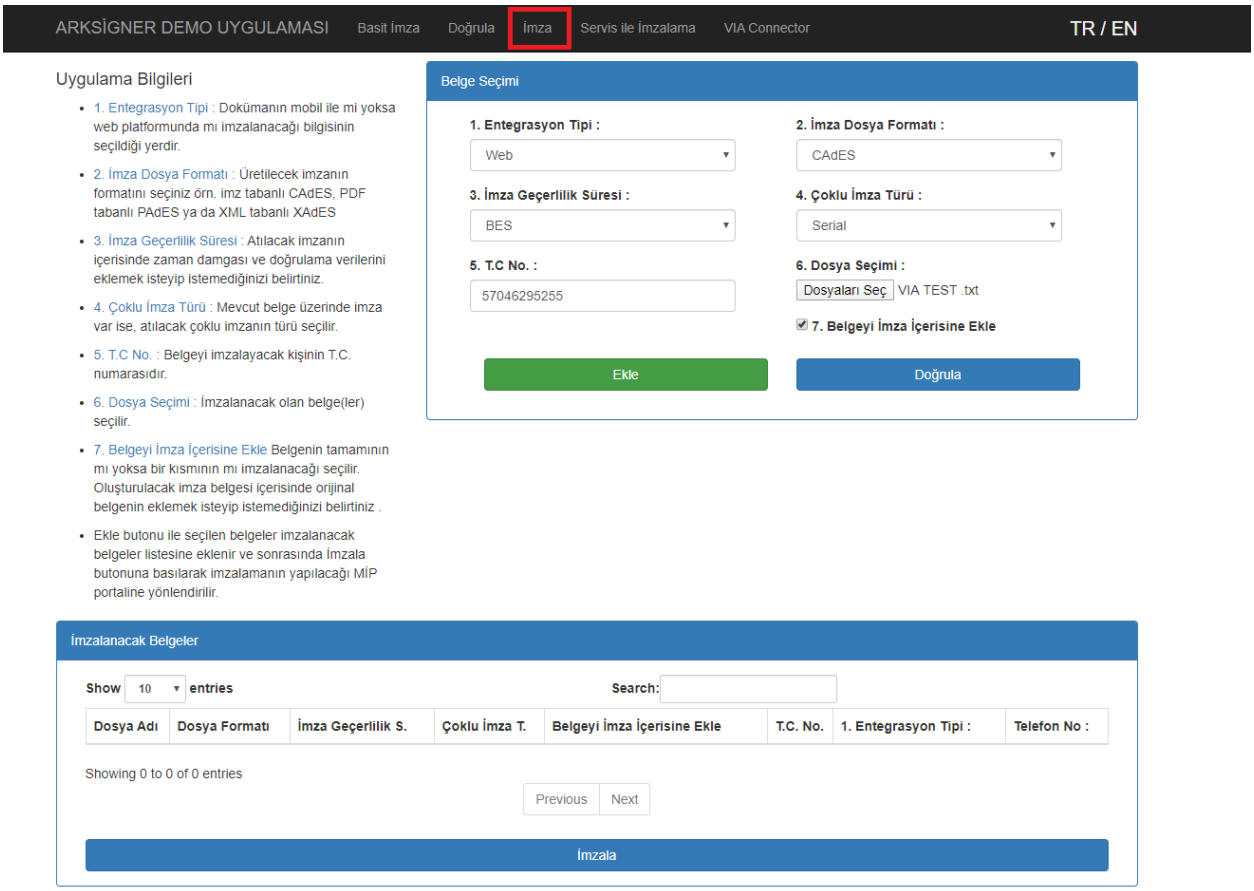
ArkSigner

VIA C#

Entegrasyon Dökümanı

Yayın Tarihi: 20.01.2020

Httpweb Request Yöntemi ile Entegrasyon



Şekil 5

Adım 1 :

Bu aşamada imzalanacak doküman HttpWebRequest yöntemi kullanılarak VIA portaline gönderilir. İmzalama işlemi **Adım 2** sayfasında anlatılacaktır. İmzalama işlemi tamamlandıktan sonra VIA tarafından ilgili belgenin indirilmesi için gerekli Url adresleri tarafınıza iletilir.

İmzalanan belgenin VIA portal sisteminden alınması için gerekli işlemler **Adım 3** de tanımlanmıştır.

İmzala butonu ile Home Controller içerisindeki Index metodu tetiklenir.

```
1.  [HttpPost]
2.    public ActionResult Index(List<SignListClass> Signs)
3.    {
4.
5.        Dictionary<string, object> responseController = new Dictionary<string, object>();

6.        if (String.IsNullOrEmpty(Session["email"] as string) || Session["email"] as strin
    g == "pleksus@via.com")
7.        {
8.            responseController["success"] = "false";
9.            responseController["message"] = "Bu İşlem İçin Yetkiniz Yok.";
10.           return Json(responseController);
11.       }
12.
13.       // iletilecek olan dosyaların toplanması
14.       // dosyaların REST bir şekilde sunucuya iletilmesi
15.       // Browser'a 302'nin gönderilmesi
16.       if (!ModelState.IsValid)
17.       {
18.           TempData["ErrorMsg"] = "Not all fields are filled";
19.       }
20.       try
21.       {
22.       ////////////////////////////////////////////////////////
23.       // Sunucuya gönderilecek olan POST request'in oluşturulması
24.       string callbackUrl = Config.AppAddress + Url.Action("GetStatus");
25.       string cancelUrl = Config.AppAddress + "/Home/CancelProcess";
26.
27.       ////////////////////////////////////////////////////////
28.       // Opsiyonel olarak T.C. Kimlik No tabanlı kimlik doğrulama
29.       // yapılacaksa onun bildirilmesi
30.       string identityNo = Signs[0].identityNo;
31.       string customKey = Guid.NewGuid().ToString() + Guid.NewGuid().ToString();
32.       NameValueCollection parameters = new NameValueCollection() { { "appId", Config.Ap
    pId }, { "pass", Config.AppPass }, { "callbackUrl", callbackUrl }, { "cancelUrl", cance
    lUrl }, { "identityNo", identityNo } };
33.       string url = Config.ServiceAddress + "/uploader?action=file-upload";
34.       string boundary = "----------------------------
    " + DateTime.Now.Ticks.ToString("x");
35.
36.       HttpWebRequest request = (HttpWebRequest)WebRequest.Create(url);
```

```
37.       request.ContentType = "multipart/form-data; boundary=" +
38.                                 boundary;
39.
40.       request.Method = "POST";
41.       request.KeepAlive = true;
42.       request.CookieContainer = cookieContainer;
43.       Stream memStream = new System.IO.MemoryStream();
44.
45.       var boundarybytes = System.Text.Encoding.ASCII.GetBytes("\r\n--" +
46.                                                               boundary + "\r\n");
47.       var endBoundaryBytes = System.Text.Encoding.ASCII.GetBytes("\r\n--" +
48.                                                                  boundary + "--");
49.
50.       string formdataTemplate = "\r\n--" + boundary +
51.                                   "\r\nContent-Disposition: form-
    data; name=\"{0}\";\r\n\r\n{1}";
52.
53.       if (parameters != null)
54.       {
55.           foreach (string key in parameters.Keys)
56.           {
57.               string formitem = string.Format(formdataTemplate, key, parameters[key]);

58.               byte[] formitembytes = System.Text.Encoding.UTF8.GetBytes(formitem);
59.               memStream.Write(formitembytes, 0, formitembytes.Length);
60.           }
61.       }
62.
63.       string headerTemplate =
64.           "Content-Disposition: form-data; name=\"{0}\"; filename=\"{1}\"\r\n" +
65.           "Content-Type: application/octet-stream\r\n\r\n";
66.
67.       foreach (var sign in Signs)
68.       {
69.           sign.customKey = "2";
70.           sign.customKey = Guid.NewGuid().ToString() + Guid.NewGuid();
71.           memStream.Write(boundarybytes, 0, boundarybytes.Length);
72.           var header = string.Format(headerTemplate, sign.FileName, sign.FileName);
73.           var headerbytes = System.Text.Encoding.UTF8.GetBytes(header);
74.
75.           memStream.Write(headerbytes, 0, headerbytes.Length);
76.           string data = JsonConvert.SerializeObject(sign);
77.
78.           byte[] byteData = Encoding.UTF8.GetBytes(data);
79.           memStream.Write(byteData, 0, byteData.Length);
80.       }
81.
82.       memStream.Write(endBoundaryBytes, 0, endBoundaryBytes.Length);
83.       request.ContentLength = memStream.Length;
84.       using (Stream requestStream = request.GetRequestStream())
85.       {
```

```
86.           memStream.Position = 0;
87.           byte[] tempBuffer = new byte[memStream.Length];
88.           memStream.Read(tempBuffer, 0, tempBuffer.Length);
89.           memStream.Close();
90.           requestStream.Write(tempBuffer, 0, tempBuffer.Length);
91.       }
92.
93.       using (WebResponse response = request.GetResponse())
94.       {
95.        using (Stream responseStream = response.GetResponseStream())
96.        {
97.            using (MemoryStream stream = new MemoryStream())
98.            {
99.                responseStream.CopyTo(stream);
100.                       byte[] responseBytes = stream.ToArray();
101.                       string responseStr = Encoding.UTF8.GetString(responseBytes);
102.                       dynamic responseObj = JsonConvert.DeserializeObject(responseStr);

103.                       if (responseObj.success == "true")
104.                       {
105.                           ResponseDocument responseDocument = new ResponseDocument
106.                           {
107.                               url = Config.ServiceAddress + "/" + responseObj.url + ".a
    spx?transactionUUID=" + responseObj.transactionUUID,
108.                               //url = Config.ServiceAddress + "/" + responseObj.url + "
    .aspx",
109.                               transactionUUID = responseObj.transactionUUID
110.                           };
111.                           Session["responseDocument"] = responseDocument;
112.                           responseController["success"] = "true";
113.                           responseController["url"] = "/Response/Index";
114.                       }
115.                       else
116.                       {
117.                           responseController["success"] = "false";
118.                           responseController["message"] = responseObj.message.Value;
119.                       }
120.                   }
121.               }
122.               }
123.               }
124.               catch (Exception ex)
125.               {
126.                   log.Fatal("Index", ex);
127.                   responseController["message"] = MerkeziImzaClient.Language.Resource.er
    ror;
128.               }
129.               return Json(responseController);
130.           }
```

Response Controller içerisinde bulunan Index metoduna ait kod bilgileri şu şekildedir.

```
1.    // GET: Response
2.  public ActionResult Index()
3.  {
4.
5.   try
6.   {
7.    ResponseDocument responseDocument = Session["responseDocument"] as ResponseDocument;

8.    if (responseDocument != null)
9.    {
10.       Response.Clear();
11.       var sb = new System.Text.StringBuilder();
12.       sb.Append("<html>");
13.       sb.AppendFormat("<body onload='document.forms[0].submit()'>");
14.       sb.AppendFormat("<form action='{0}' method='post'>", responseDocument.url);
15.       sb.AppendFormat("<input type='hidden' name='documentUUID' value='{0}'>", response
    Document.transactionUUID);
16.       sb.Append("</form>");
17.       sb.Append("</body>");
18.       sb.Append("</html>");
19.
20.       Response.Write(sb.ToString());
21.       Response.End();
22.       return new RedirectResult("responseDocument.url", false);
23.   }
24.   else
25.   {
26.       return View();
27.   }
28.  }
29.  catch (Exception)
30.  {
31.      throw;
32.  }
33. }
```

Bu işlemlerden sonra VIA Portaline yönlendirme yapılır. Sonrasında ise **“VIA Connector ile Entegrasyon”** kısmında anlatılan **Adım 2** ve **Adım 3** kısımları ile işlemlere devam edilir.