Index.cshtml

```
@model List<string>
@{
    ViewBag.Title = "Index";
}

<h2>Index</h2>

@using (Html.BeginForm("Add", "Blob", FormMethod.Post, new
{enctype = "multipart/form-data" }))
{
    <div>
        <input type="file" name="pic" id="pic" />
        <input type="submit" value="Upload Now" id="s1" />
    </div>
}
<ul>
    @foreach (var item in Model)
    {
        <li>

            <input type="button" name="b1" id="b1"
value="Delete"
                    onclick="Remove('@item')" />
        <video src="@item" height="200" width="200" controls />
        </li>
    }
</ul>
@section scripts{
    <script>
        function Remove(x) {
            alert(x);
            var uri = "/Blob/remove";
            $.post(uri, { name: x }, function (y) {
                window.location.href = "/blob/index";
                alert(y);
            });
        }
</script>}
```

BLBLOBS.cs

```
using Microsoft.WindowsAzure.ServiceRuntime;
using Microsoft.WindowsAzure.Storage;
using Microsoft.WindowsAzure.Storage.Blob;//
using System.IO;

namespace MVCBLOBS.BL
{
    public class BLBLOBS
    {
        public CloudBlobContainer GetBLOBRef()
        {
            CloudStorageAccount storageac = CloudStorageAccount.Parse(
                RoleEnvironment.GetConfigurationSettingValue("scn"));
            CloudBlobClient blobclient = storageac.CreateCloudBlobClient();
            CloudBlobContainer blobcontainer = blobclient.GetContainerReference("myvideos");
            if (blobcontainer.CreateIfNotExists())
                blobcontainer.SetPermissions(new BlobContainerPermissions
                { PublicAccess = BlobContainerPublicAccessType.Blob });

            return blobcontainer;
        }

        public List<string> GetBlobList()
        {

            CloudBlobContainer blobcontainer = GetBLOBRef();
            List<string> blobs = new List<string>();
            foreach (var item in blobcontainer.ListBlobs())
            {
                blobs.Add(item.Uri.ToString());
            }
            return blobs;
        }

        public void AddBlob(HttpPostedFileBase pic)
        {
            if (pic.ContentLength > 0)
            {
                CloudBlobContainer blobcontainer = GetBLOBRef();
                CloudBlockBlob blob = blobcontainer.GetBlockBlobReference(pic.FileName);
                blob.UploadFromStream(pic.InputStream);
            }
        }

        internal void DeleteBlob(string name)
        {
            Uri ur = new Uri(name);
            string fname = Path.GetFileName(ur.LocalPath);
            CloudBlobContainer blobcontainer = GetBLOBRef();
            CloudBlockBlob blob = blobcontainer.GetBlockBlobReference(fname);
            blob.Delete();
        }
    }
}
```

BLOBController in MVC

```
namespace MVCBLOBS.Controllers
{
    public class BlobController : Controller
    {
        BL.BLBLOBS objbl = new BL.BLBLOBS();
        // GET: Blob
        public ActionResult Index()
        {
            return View(objbl.GetBlobList());
        }

        [HttpPost]
        public ActionResult Add(HttpPostedFileBase pic)
        {
            objbl.AddBlob(pic);

            return RedirectToAction("Index");
        }

        [HttpPost]
        public string Remove(string name)
        {
            objbl.DeleteBlob(name);
            return "Blob Removed Successfully";
        }
    }
}
```