MVC Copypasta

Bits and Bobs

Annotation for putting a Datetime in place

```
[Required]
[DataType(DataType.Date)]
[DisplayFormat(DataFormatString = "{0:yyyy-MM-dd}", ApplyFormatInEditMode = true)]
```

In the One

```
public virtual ICollection<[CHILDREN]> [CHILDREN] { get; set; }
public virtual ICollection<Joke> Jokes { get; set; }
```

In the Many

```
public int [PARENT]Id { get; set; }

public virtual [PARENT] [PARENT] { get; set; }
public int CategoryId { get; set; }

public virtual Category Category { get; set; }
```

MISC

```
[Required]
        public int ID { get; set; }
```

Paging and/or Searching

Remember to NuGet:
PagedList.MVC

Just Paging:

Controller

```
        [Authorize(Roles = "Admin,User")]
        public ActionResult Index(int? page)
        {
            var [TABLE] = db.[TABLE].Include(j => j.[1-MCOLUMN]);
            int pageSize = 2;
            int pageNumber = (page ?? 1);

            jokes = [TABLE].OrderBy(j => j.[COLUMN]);

            return View([TABLE].ToPagedList(pageNumber,pageSize));
        }
```

```
        ///Example implementation
        [Authorize(Roles = "Admin,User")]
        public ActionResult Index(int? page)
```

MVC Copypasta

```
        {
            var jokes = db.Jokes.Include(j => j.Category);
            int pageSize = 2;
            int pageNumber = (page ?? 1);

            jokes = jokes.OrderBy(j => j.Title);

            return View(jokes.ToPagedList(pageNumber,pageSize));
        }
```

MVC Copypasta

Index

```
<!--At the top before code-->
@model PagedList.IPagedList<[PROJECTNAME].Models.[TABLE]>
@using PagedList.Mvc

<!--At the bottom after code-->
<br />
        Page @(Model.PageCount < Model.PageNumber ? 0 : Model.PageNumber) of
@Model.PageCount
        @Html.PagedListPager(Model, Page => Url.Action("Index", new { Page}))
```

```
<!--At the top before code-->
@model PagedList.IPagedList<mvcJokeShop.Models.Joke>
@using PagedList.Mvc

<!--At the bottom after code-->
<br />
        Page @(Model.PageCount < Model.PageNumber ? 0 : Model.PageNumber) of
@Model.PageCount
        @Html.PagedListPager(Model, Page => Url.Action("Index", new { Page}))
```

Paging and Sorting:

Remember to NuGet:

PagedList.MVC

Sorting link example

```
@Html.ActionLink("Last Name","Index", new { sortOrder = ViewBag.NameSortParm})
```

Controller:

```
public ActionResult Index(string sortOrder,string currentFilter, string searchString,
int? page)
        {
            ViewBag.NameSortParm = String.IsNullOrEmpty(sortOrder) ? "name_desc" : "";
            ViewBag.TypeSortParm = sortOrder == "Type" ? "type_desc" : "Type";

            if (searchString != null)
            {
                page = 1;
            }
            else
            {
                searchString = currentFilter;
            }

            ViewBag.CurrentFilter = searchString;
            ViewBag.CurrentSort = sortOrder;

            var Pets = from p in db.Pets select p;

            if (!String.IsNullOrEmpty(searchString))
            {
```

MVC Copypasta

```
                [TABLE] = [TABLE].Where(p =>
p.[COL1].ToUpper().Contains(searchString.ToUpper()) ||
p.[COL2].ToUpper().Contains(searchString.ToUpper()));
            }

            switch (sortOrder)
            {
                case "[COL1]_desc":
                    [TABLE] = [TABLE].OrderByDescending(p => p.Name);
                    break;
                case "[COL1]":
                    Pets = Pets.OrderBy(p => p.Type);
                    break;
                case "[COL2]_desc":
                    Pets = Pets.OrderByDescending(p => p.[COL2]);
                     break;
                default:
                    Pets = Pets.OrderBy(p => p.[COL2]);
                    break;
            }

            int pageSize = 2; int pageNumber = (page ?? 1);

            return View([TABLE].Include(p => p.[PARENT
TABLE]).ToPagedList(pageNumber,pageSize));        }
```

```
public ActionResult Index(string sortOrder,string currentFilter, string searchString,
int? page)
        {
            ViewBag.NameSortParm = String.IsNullOrEmpty(sortOrder) ? "name_desc" : "";
            ViewBag.TypeSortParm = sortOrder == "Type" ? "type_desc" : "Type";

            if (searchString != null)
            {
                page = 1;
            }
            else
            {
                searchString = currentFilter;
            }

            ViewBag.CurrentFilter = searchString;
            ViewBag.CurrentSort = sortOrder;

            var Pets = from p in db.Pets select p;

            if (!String.IsNullOrEmpty(searchString))
            {
                Pets = Pets.Where(p =>
p.Name.ToUpper().Contains(searchString.ToUpper()) ||
p.Type.ToUpper().Contains(searchString.ToUpper()));
            }

            switch (sortOrder)
            {
                case "type_desc":
                    Pets = Pets.OrderByDescending(p => p.Name);
                    break;
                case "Type":
```

MVC Copypasta

```
                    Pets = Pets.OrderBy(p => p.Type);
                    break;
                case "name_desc":
                    Pets = Pets.OrderByDescending(p => p.Name);
                     break;
                default:
                    Pets = Pets.OrderBy(p => p.Name);
                    break;
            }

            //var pets = db.Pets.Include(p => p.Owner);
            int pageSize = 2; int pageNumber = (page ?? 1);

            return View(Pets.Include(p => p.Owner).ToPagedList(pageNumber,pageSize));
            // return View(pets.ToList());
        }
```

Index

```
<!--At the top before table-->


@model PagedList.IPagedList<[PROJECT NAME].Models.Pet>
@using PagedList.Mvc

@{
    ViewBag.Title = "Index";
}

<h2>Index</h2>

<p>
    @Html.ActionLink("Create New", "Create")
</p>
@using (Html.BeginForm())
{
    <p>
        Find by Name: @Html.TextBox("SearchString")
        <input type="submit" value="Search" />
    </p>
}

<!--At the bottom after code-->
Page @(Model.PageCount < Model.PageNumber ? 0 : Model.PageNumber) of @Model.PageCount
@Html.PagedListPager(Model, page => Url.Action("Index", new { page, sortOrder =
ViewBag.CurrentSort, currentFilter = ViewBag.CurrentFilter }))
```

MVC Copypasta

Image Uploading

File Controller

```
using mvcContactsUpload.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;

namespace mvcContactsUpload.Controllers
{
    public class FileController : Controller
    {
        // GET: File
        private ContactContext db = new ContactContext();

        public ActionResult Index(int id)
        {
            var fileToRetrieve = db.Files.Find(id);
            return File(fileToRetrieve.Content, fileToRetrieve.ContentType);
        }
    }
}
```

FileModel (One – to -many relationship)

```
    public class File
    {
        public int FileID { get; set; }
        [StringLength(255)]
        public string FileName { get; set; }
        [StringLength(100)]
        public string ContentType { get; set; }
        public byte[] Content { get; set; }
         public FileType FileType { get; set; }
        public int [PARENT]Id { get; set; }
        public virtual [PARENT] [PARENT] { get; set; }
    }
```

```
    public class File
    {
        public int FileID { get; set; }
        [StringLength(255)]
        public string FileName { get; set; }
        [StringLength(100)]
        public string ContentType { get; set; }
        public byte[] Content { get; set; }
         public FileType FileType { get; set; }
        public int ContactId { get; set; }
        public virtual Contact Contact { get; set; }
    }
```

MVC Copypasta

FileEnum

```
public enum FileType { Avatar = 1, Photo}
```

In the model that will have a picture to it

```
public virtual ICollection<File> Files { get; set; }
```

Controller/Details

```
[Object] [NAMEHERE] = db.[Objects].Include(c => c.Files).SingleOrDefault(c => c.ID == id);
```

```
        public ActionResult Details(int? id)
        {
            if (id == null)
            {
                return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
            }
            //Contact contact = db.Contacts.Find(id);
            Contact contact = db.Contacts.Include(c => c.Files).SingleOrDefault(c => c.ID == id);

            if (contact == null)
            {
                return HttpNotFound();
            }
            return View(contact);
        }
```

Controller/Create

```
                if (upload != null && upload.ContentLength > 0)
                {
                    var avatar = new File
                    {
                        FileName = System.IO.Path.GetFileName(upload.FileName),
                        FileType = FileType.Avatar,
                        ContentType = upload.ContentType

                    };
                    using (var reader = new System.IO.BinaryReader(upload.InputStream))
                    {
                        avatar.Content = reader.ReadBytes(upload.ContentLength);
                    }
                    [OBJECT].Files = new List<File> { avatar };
                }
```

```
        [HttpPost]
        [ValidateAntiForgeryToken]
```

MVC Copypasta

```
        public ActionResult Create([Bind(Include =
"ID,FirstName,MiddleName,LastName,StreetAddress,City,Province,EmailAddress,PhoneNumber"
)] Contact contact, HttpPostedFileBase upload)
        {
            if (ModelState.IsValid)
            {
                //db.Contacts.Add(contact);
               // db.SaveChanges();
               // return RedirectToAction("Index");
               if (upload != null && upload.ContentLength > 0)
                {
                    var avatar = new File
                    {
                        FileName = System.IO.Path.GetFileName(upload.FileName),
                        FileType = FileType.Avatar,
                        ContentType = upload.ContentType

                    };
                    using (var reader = new System.IO.BinaryReader(upload.InputStream))
                    {
                        avatar.Content = reader.ReadBytes(upload.ContentLength);
                    }
                    contact.Files = new List<File> { avatar };
                }
                db.Contacts.Add(contact);
                db.SaveChanges();
                return RedirectToAction("Index");
            }

            return View(contact);
        }
```

MVC Copypasta

Controller/Edit

```
                if (upload != null && upload.ContentLength > 0)
                {
                    var avatar = new File
                    {
                        FileName = System.IO.Path.GetFileName(upload.FileName),
                        FileType = FileType.Avatar,
                        ContentType = upload.ContentType

                    };
                    using (var reader = new System.IO.BinaryReader(upload.InputStream))
                    {
                        avatar.Content = reader.ReadBytes(upload.ContentLength);
                    }
                    contact.Files = new List<File> { avatar };
                }
```

```
        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult Edit([Bind(Include =
"ID,FirstName,MiddleName,LastName,StreetAddress,City,Province,EmailAddress,PhoneNumber"
)] Contact contact, HttpPostedFileBase upload)
        {
            if (ModelState.IsValid)
            {
                db.Entry(contact).State = EntityState.Modified;

                if (upload != null && upload.ContentLength > 0)
                {
                    var avatar = new File
                    {
                        FileName = System.IO.Path.GetFileName(upload.FileName),
                        FileType = FileType.Avatar,
                        ContentType = upload.ContentType

                    };
                    using (var reader = new System.IO.BinaryReader(upload.InputStream))
                    {
                        avatar.Content = reader.ReadBytes(upload.ContentLength);
                    }
                    contact.Files = new List<File> { avatar };
                }

                db.SaveChanges();
                return RedirectToAction("Index");
            }
            return View(contact);
        }
```

MVC Copypasta

Controller/Details

```
            Contact contact = db.Contacts.Include(c => c.Files).SingleOrDefault(c =>
c.ID == id);
```

```
        public ActionResult Details(int? id)
        {
            if (id == null)
            {
                return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
            }
            //Contact contact = db.Contacts.Find(id);
            Contact contact = db.Contacts.Include(c => c.Files).SingleOrDefault(c =>
c.ID == id);

            if (contact == null)
            {
                return HttpNotFound();
            }
            return View(contact);
        }
```

Create.cshtml, replace @using(Html.BeginForm())

```
@using (Html.BeginForm("Create", "[CONTROLLER NAME]", null, FormMethod.Post, new {
enctype = "multipart/form-data" }))
```

```
@using (Html.BeginForm("Create", "Contacts", null, FormMethod.Post, new { enctype =
"multipart/form-data" }))
```

Edit.cshtml, replace @using(Html.BeginForm())

```
@using (Html.BeginForm("Edit","[CONTROLLER NAME]",null,FormMethod.Post, new { enctype =
"multipart/form-data"}))
```

```
@using (Html.BeginForm("Edit","Contacts",null,FormMethod.Post, new { enctype =
"multipart/form-data"}))
```

Create.cshtml, Somewhere before the submit container

```
        <div class="form-group">
            @Html.Label("Avatar",new { @class = "control-label col-md-2"})
            <div class="col-md-10">
                <input type="file" id="Avatar" name="upload" />
            </div>
        </div>
```

MVC Copypasta

Display.cshtml (After the H4)

```
    @if (Model.Files.Any(f => f.FileType == FileType.Avatar))
    {
        <dt>Avatar</dt>
        <dd>
            <img src="~/File?id=@Model.Files.Last(f=>f.FileType ==
FileType.Avatar).FileID " alt="avatar" style="width:30%;" />
        </dd>
    }
```

Edit.cshtml, Somewhere before the submit container

```
        @if (Model.Files.Count > 0)
        {
            if (Model.Files.Any (f => f.FileType == FileType.Avatar))
            {
                <div class="form-group">
                    <span class="control-label col-md-2"><strong>Current
Avatar</strong></span>
                    <div class="col-md-10">
                        <img src="~/File?id=@Model.Files.Last(f => f.FileType  ==
FileType.Avatar).FileID" alt="avatar" style="width:30%;"/>
                    </div>
                </div>

            }
        }

        <div class="form-group">
            @Html.Label("Avatar",new { @class = "control-label col-md-2"})
            <div class="col-md-10">
                <input type="file" id="Avatar" name="upload" />
            </div>
        </div>
```

MVC Copypasta

Display.cshtml (After the H4)

```
    @if (Model.Files.Any(f => f.FileType == FileType.Avatar))
    {
        <dt>Avatar</dt>
        <dd>
            <img src="~/File?id=@Model.Files.Last(f=>f.FileType ==
FileType.Avatar).FileID " alt="avatar" style="width:30%;" />
        </dd>
    }
```

MVC Copypasta

Authentication:

Refer to “Auth ASPidentitySteps

```
  <connectionStrings>
    <add name="DefaultConnection" connectionString="Data Source=localhost;Initial
Catalog=[NEW DB HERE];Integrated Security=SSPI;" providerName="System.Data.SqlClient"
/>
  </connectionStrings>
```

```
  <connectionStrings>
    <add name="DefaultConnection" connectionString="Data Source=localhost;Initial
Catalog=mvcJokeShopTest;Integrated Security=SSPI;" providerName="System.Data.SqlClient"
/>
  </connectionStrings>
```

14) AccountController/Register (POST) AFTER CONFIGURING THE DATABASE

```
var currentUser = UserManager.FindByName(user.UserName);
                    var roleResult = UserManager.AddToRole(currentUser.Id, "User");
```

```
        // POST: /Account/Register
        [HttpPost]
        [AllowAnonymous]
        [ValidateAntiForgeryToken]
        public async Task<ActionResult> Register(RegisterViewModel model)
        {
            if (ModelState.IsValid)
            {
                var user = new ApplicationUser { UserName = model.Email, Email =
model.Email };
                var result = await UserManager.CreateAsync(user, model.Password);
                if (result.Succeeded)
                {
                    var currentUser = UserManager.FindByName(user.UserName);
                    var roleResult = UserManager.AddToRole(currentUser.Id, "User");

                    await SignInManager.SignInAsync(user, isPersistent:false,
rememberBrowser:false);

                    // For more information on how to enable account confirmation and
password reset please visit https://go.microsoft.com/fwlink/?LinkID=320771
                    // Send an email with this link
                    // string code = await
UserManager.GenerateEmailConfirmationTokenAsync(user.Id);
                    // var callbackUrl = Url.Action("ConfirmEmail", "Account", new {
userId = user.Id, code = code }, protocol: Request.Url.Scheme);
                    // await UserManager.SendEmailAsync(user.Id, "Confirm your
account", "Please confirm your account by clicking <a href=\"" + callbackUrl +
"\">here</a>");

                    return RedirectToAction("Index", "Home");
                }
```

MVC Copypasta

```
                AddErrors(result);
            }

            // If we got this far, something failed, redisplay form
            return View(model);
        }
```

Restrictive annotations (For Controllers)

```
[Authorize(Roles = "Admin")]
[Authorize(Roles = "User")]
```