Documents:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.ComponentModel;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace KHBPADevProject.Models

{

public class Document

{

[Key]

[DatabaseGenerated(DatabaseGeneratedOption.Identity)]

public int Id { get; set; }

public byte[] FileBytes { get; set; } //File Content

public int ContentLength { get; set; } //File Length

public string ContentType { get; set; } //application/pdf

public string FileName { get; set; } //Name with extention

public string UploadedBy { get; set; } //Upload User

[DisplayFormat(DataFormatString = "{0:MM/dd/yyyy}", ApplyFormatInEditMode = true)]

public DateTime UploadDate { get; set; } //Date Uploaded

[DisplayFormat(DataFormatString = "{0:MM/dd/yyyy}", ApplyFormatInEditMode = true)]

public DateTime? InformationDate { get; set; } //Additional Information for Entity

public string EntityType { get; set; } //Type for Entity (Newsletter, photo, document, etc)

public string EntityID { get; set; }

public string AdditionalText { get; set; } //Additional information (title/etc)

public List<DocumentAttributes> DocAttributes { get; set; } //Attributes (ID, Etc. Other Attributes for Documents

public bool? Verified { get; set; } //Indicator for verified status (Board Mins, ID, Etc)

}

public class DocumentAttributes

{

[Key]

[DatabaseGenerated(DatabaseGeneratedOption.Identity)]

public int Id { get; set; }

public Document Doc { get; set; }

public string Attribute { get; set; }

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using KHBPADevProject.Models;

namespace KHBPADevProject.Controllers

{

public class DocumentController : Controller

{

public KHBPADevProjectContext \_db = new KHBPADevProjectContext();

// GET: Document

public ActionResult Index()

{

return View();

}

public ActionResult DocumentManagement()

{

return View("Index");

}

[HttpPost]

public ActionResult UploadDocument(int studentNumber, HttpPostedFileBase file)

{

byte[] uploadedFile = new byte[file.InputStream.Length];

file.InputStream.Read(uploadedFile, 0, uploadedFile.Length);

/\* var student = \_db.Students.FirstOrDefault(s => s.StudentNumber == studentNumber);

if (student != null)

{

var documentModel = new Document

{

StudentId = student.Id,

StudentNumber = studentNumber,

UploadedBy = HttpContext.User.Identity.Name,

ContentLength = file.ContentLength,

ContentType = file.ContentType,

FileName = file.FileName,

FileBytes = uploadedFile

};

\_db.Documents.Add(documentModel);

\_db.SaveChanges();

}\*/

return View("DocumentManagement");

}

//POST: Document/UploadBoardMins

[HttpPost]

public ActionResult UploadBoardMins(string MeetingOrg, DateTime MeetingDate, bool isVerified, HttpPostedFileBase file)

{

byte[] uploadedFile = new byte[file.InputStream.Length];

file.InputStream.Read(uploadedFile, 0, uploadedFile.Length);

/\* var student = \_db.Students.FirstOrDefault(s => s.StudentNumber == studentNumber);

if (student != null)

{

var documentModel = new Document

{

StudentId = student.Id,

StudentNumber = studentNumber,

UploadedBy = HttpContext.User.Identity.Name,

ContentLength = file.ContentLength,

ContentType = file.ContentType,

FileName = file.FileName,

FileBytes = uploadedFile

};

\_db.Documents.Add(documentModel);

\_db.SaveChanges();

}\*/

return View("DocumentManagement");

}

public ActionResult GetDocument(int studentId)

{

/\* var allDocumentsForStudent = \_db.Documents.Where(d => d.StudentId == studentId);

var oneDocumentFromStudent = allDocumentsForStudent.FirstOrDefault();

if (oneDocumentFromStudent != null)

{

return File(oneDocumentFromStudent.FileBytes, "application/octet-stream", oneDocumentFromStudent.FileName);

}

\*/

return RedirectToAction("DocumentManagement");

}

#region Newsletter

// GET: Document/Newsletter

public ActionResult Newsletter()

{

List<Document> Newsletter = (from doc in \_db.Documents

where doc.EntityType == "Newsletter"

orderby doc.InformationDate descending

select doc).ToList();

return View(Newsletter);

}

//GET: Document/NewsletterViewDocument

public ActionResult ViewNewsletter(int Id)

{

var mins = \_db.Documents.Where(d => d.Id == Id);

var viewDoc = mins.FirstOrDefault();

if (viewDoc != null)

{

return File(viewDoc.FileBytes, "application/octet-stream", viewDoc.FileName);

}

return RedirectToAction("Newsletter");

}

//POST: Document/NewsletterViewDocument

public ActionResult DeleteNewsletter(int Id)

{

var mins = \_db.Documents.Where(d => d.Id == Id);

var viewDoc = mins.FirstOrDefault();

if (viewDoc != null)

{

\_db.Documents.Remove(viewDoc);

\_db.SaveChanges();

}

return RedirectToAction("Newsletter");

}

//GET: Document/UploadNewsletter

public ActionResult UploadNewsletter()

{

return View();

}

//POST: Document/UploadNewsletter

[HttpPost]

public ActionResult UploadNewsletter(DateTime NewsletterDate, HttpPostedFileBase file)

{

byte[] uploadedFile = new byte[file.InputStream.Length];

file.InputStream.Read(uploadedFile, 0, uploadedFile.Length);

//var docs = db.Documents.FirstOrDefault(s => s.StudentNumber == studentNumber);

var documentModel = new Document

{

InformationDate = NewsletterDate,

//Verified = DocuStatus,

UploadDate = DateTime.Now,

UploadedBy = HttpContext.User.Identity.Name,

ContentLength = file.ContentLength,

ContentType = file.ContentType,

FileName = "Newsletter.pdf",

FileBytes = uploadedFile,

EntityType = "Newsletter"

};

\_db.Documents.Add(documentModel);

\_db.SaveChanges();

return RedirectToAction("Newsletter");

}

#endregion

}

}

Index

@model IEnumerable<KHBPADevProject.Models.Document>

@{

ViewBag.Title = "Newsletters";

}

<style>

section {

margin-top: 150px;

margin-left: 60px;

margin-right: 30px;

margin-bottom: 100px;

}

</style>

<section id="intro">

<h2>Newsletter</h2>

<p>

@Html.ActionLink("Upload New", "UploadNewsletter")

</p>

<table class="table">

<tr>

<th>

Newsletter Date@\*Html.DisplayNameFor(model => model.InformationDate)\*@

</th>

<th></th>

</tr>

@foreach (var item in Model)

{

<tr>

<td>

@Html.DisplayFor(modelItem => item.InformationDate)

</td>

<td>

@Html.ActionLink("View", "ViewNewsletter", new { id = item.Id }) |

@Html.ActionLink("Delete", "DeleteNewsletter", new { id = item.Id })

</td>

</tr>

}

</table>

<p> This is where newsletters will be entered with the most recent date at the top and older dates at the bottom </p>

</section>

@using KHBPADevProject.Controllers

@{

ViewBag.Title = "UploadNewsletter";

}

<style>

section {

margin-top: 150px;

margin-left: 60px;

margin-right: 30px;

margin-bottom: 100px;

}

</style>

<section id="intro">

<h2>Newsletter</h2>

<div class="panel panel-primary">

<div class="panel-heading">

<h4>Upload Documents</h4>

</div>

<div class="panel-body">

@using (Html.BeginForm("UploadNewsletter", "Document", FormMethod.Post, new { enctype = "multipart/form-data" }))

{

<div>

<label for="NewsletterDate">Newsletter Date:</label>

<input type="text" name="NewsletterDate" id="NewsletterDate" />

</div>

<br />

<div>

<label for="file">Newsletter File</label>

<input type="file" name="file" id="file" />

</div>

<br />

<input value="Upload File" type="submit" />

}

</div>

</div>

</section>

Events:

using System;

using System.ComponentModel.DataAnnotations;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace KHBPADevProject.Models

{

public class Event

{

[Required]

public int id { get; set; }

[Display(Name = "Title")]

public string title { get; set; }

[Display(Name = "Text")]

public string text { get; set; }

[Display(Name ="Start Date")]

public DateTime start\_date { get; set; }

[Display(Name = "End Date")]

public DateTime end\_date { get; set; }

}

}

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Entity;

using System.Linq;

using System.Net;

using System.Web;

using System.Web.Mvc;

using KHBPADevProject.Models;

namespace KHBPADevProject.Controllers

{

public class EventsController : Controller

{

private KHBPADevProjectContext db = new KHBPADevProjectContext();

// GET: Events

public ActionResult Index()

{

return View(db.Events.ToList());

}

// GET: Events/Details/5

public ActionResult Details(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Event @event = db.Events.Find(id);

if (@event == null)

{

return HttpNotFound();

}

return View(@event);

}

// GET: Events/Create

public ActionResult Create()

{

return View();

}

// POST: Events/Create

// To protect from overposting attacks, please enable the specific properties you want to bind to, for

// more details see https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Create([Bind(Include = "id,title,text,start\_date,end\_date")] Event @event)

{

if (ModelState.IsValid)

{

db.Events.Add(@event);

db.SaveChanges();

return RedirectToAction("Index");

}

return View(@event);

}

// GET: Events/Edit/5

public ActionResult Edit(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Event @event = db.Events.Find(id);

if (@event == null)

{

return HttpNotFound();

}

return View(@event);

}

// POST: Events/Edit/5

// To protect from overposting attacks, please enable the specific properties you want to bind to, for

// more details see https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Edit([Bind(Include = "id,text,start\_date,end\_date")] Event @event)

{

if (ModelState.IsValid)

{

db.Entry(@event).State = EntityState.Modified;

db.SaveChanges();

return RedirectToAction("Index");

}

return View(@event);

}

// GET: Events/Delete/5

public ActionResult Delete(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Event @event = db.Events.Find(id);

if (@event == null)

{

return HttpNotFound();

}

return View(@event);

}

// POST: Events/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public ActionResult DeleteConfirmed(int id)

{

Event @event = db.Events.Find(id);

db.Events.Remove(@event);

db.SaveChanges();

return RedirectToAction("Index");

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

db.Dispose();

}

base.Dispose(disposing);

}

}

}

\_Partial View on homescreen

@using KHBPADevProject.Models

@model IEnumerable<Event>

<style>

body {

margin-top: 150px;

}

.bs-callout {

padding: 20px;

margin: 20px 0;

border: 1px solid #eee;

border-left-width: 5px;

border-radius: 3px;

}

.bs-callout h4 {

margin-top: 0;

margin-bottom: 5px;

}

.bs-callout p:last-child {

margin-bottom: 0;

}

.bs-callout code {

border-radius: 3px;

}

.bs-callout + .bs-callout {

margin-top: -5px;

}

.bs-callout-default {

border-left-color: whitesmoke;

color: aliceblue;

}

.bs-callout-default h4 {

color: whitesmoke;

}

</style>

@if (Model.Count() == 0)

{

<div>

<p>There are no events scheduled.</p>

</div>

} else {

foreach (Event e in Model)

{

<div class="bs-callout bs-callout-default">

<h4>@e.text</h4>

@e.start\_date.ToString("MM/dd/yyyy") - @e.end\_date.ToString("MM/dd/yyyy")

</div>

}

}

@model IEnumerable<KHBPADevProject.Models.Event>

@{

ViewBag.Title = "Index";

}

<style>

section {

margin-top: 150px;

margin-left: 60px;

margin-right: 30px;

margin-bottom: 100px;

}

</style>

<section id="intro">

<h2>Index</h2>

<p>

@Html.ActionLink("Create New", "Create")

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.title)

</th>

<th>

@Html.DisplayNameFor(model => model.text)

</th>

<th>

@Html.DisplayNameFor(model => model.start\_date)

</th>

<th>

@Html.DisplayNameFor(model => model.end\_date)

</th>

<th></th>

</tr>

@foreach (var item in Model)

{

<tr>

<td>

@Html.DisplayFor(modelItem => item.title)

</td>

<td>

@Html.DisplayFor(modelItem => item.text)

</td>

<td>

@Html.DisplayFor(modelItem => item.start\_date)

</td>

<td>

@Html.DisplayFor(modelItem => item.end\_date)

</td>

<td>

@Html.ActionLink("Edit", "Edit", new { id = item.id }) |

@Html.ActionLink("Details", "Details", new { id = item.id }) |

@Html.ActionLink("Delete", "Delete", new { id = item.id })

</td>

</tr>

}

</table>

</section>