Controller/HomeController.cs

**using** **HW8\_Pirates.Models**;

**using** **System**;

**using** **System.Collections.Generic**;

**using** **System.Diagnostics**;

**using** **System.Linq**;

**using** **System.Web**;

**using** **System.Web.Mvc**;

**namespace** **HW8\_Pirates.Controllers**

{

**public** **class** **HomeController** : Controller

{

**private** Model db = **new** Model();

*/// <summary>*

*/// Home page view*

*/// </summary>*

*/// <returns>Home page view</returns>*

**public** ActionResult Index()

{

**return** View();

}

*/// <summary>*

*/// Initial http get for crew listing view*

*/// </summary>*

*/// <returns>Crew listing view</returns>*

**public** ActionResult CrewView()

{

**return** View(db.Crews);

}

*/// <summary>*

*/// The initial http get for the ships view*

*/// </summary>*

*/// <returns>The ships listing view</returns>*

**public** ActionResult ShipsView()

{

**return** View(db.Ships);

}

*/// <summary>*

*/// Initial http get for the paged pirates view*

*/// </summary>*

*/// <param name="button">Which page button was clicked</param>*

*/// <returns>The view with only 3 pirates depending on which page</returns>*

**public** ActionResult PiratesView(string button)

{

int page = 0;

**if** (button != **null**)

{

page = int.Parse(button);

}

int pageSize = 3;

ViewBag.PagesCount = GetPiratePagesCount(pageSize);

var piratesList = (**from** p **in** db.Pirates

**orderby** p.ID

**select** p).Skip(pageSize \* (int)page).Take(pageSize).ToList();

**return** View(piratesList);

}

*/// <summary>*

*/// Returns the number of pages given the page size*

*/// </summary>*

*/// <param name="pageSize">The number of pirates per page</param>*

*/// <returns>The number of pages</returns>*

**public** int GetPiratePagesCount(int pageSize)

{

float pageCount = (float)db.Pirates.ToList().Count / (float)pageSize;

**if** ((int)pageCount < pageCount)

{

**return** ((int)(pageCount + 1));

}

**return** ((int)pageCount);

}

*/// <summary>*

*/// Initial http get for the create pirate page.*

*/// </summary>*

*/// <returns>The create pirate view</returns>*

**public** ActionResult CreatePirate()

{

**return** View();

}

*/// <summary>*

*/// The post for the create pirate page. Adds the newPirate into the database.*

*/// </summary>*

*/// <param name="newPirate">The pirate which to add into the database</param>*

*/// <returns>The details of the new pirate</returns>*

[HttpPost]

**public** ActionResult CreatePirate(Pirate newPirate)

{

**if** (ModelState.IsValid && newPirate.DateConscripted < DateTime.Now)

{

db.Pirates.Add(newPirate);

db.SaveChanges();

**return** RedirectToAction("PiratesView");

}

**return** View(newPirate);

}

*/// <summary>*

*/// Called by ajax to get json information about the total booty counts*

*/// </summary>*

*/// <returns>The json information</returns>*

**public** JsonResult GetBooty()

{

var bootyList = db.Pirates.Select(p => **new** {Name = p.Name, Booty = p.Crews.Select(c => c.Booty).Sum()})

.OrderByDescending(pnb => pnb.Booty);

var data = **new**

{

bootyVal = bootyList,

bootySize = bootyList.Count(),

};

**return** Json(data, JsonRequestBehavior.AllowGet);

}

*/// <summary>*

*/// The read pirates page*

*/// </summary>*

*/// <param name="id">The id of the pirate displaying details for</param>*

*/// <returns>The read pirate view</returns>*

**public** ActionResult ReadPirate(int? id)

{

**if** (id == **null**)

{

**return** RedirectToAction("PiratesView");

}

Pirate pirate = db.Pirates.Find(id);

**if** (pirate == **null**)

{

**return** RedirectToAction("PiratesView");

}

**return** View(pirate);

}

*/// <summary>*

*/// Initial update pirate page*

*/// </summary>*

*/// <param name="id">The id of the pirate</param>*

*/// <returns>The view of the update pirate page</returns>*

**public** ActionResult UpdatePirate(int? id)

{

Pirate pir = db.Pirates.Find(id);

**return** View(pir);

}

*/// <summary>*

*/// Update a pirate with new information*

*/// </summary>*

*/// <param name="id">The id of the pirate updating</param>*

*/// <param name="form">Used to separate from http get update pirate page</param>*

*/// <returns>The pirate list view or the updated pirate (if valid)</returns>*

[HttpPost]

**public** ActionResult UpdatePirate(int? id, FormCollection form)

{

DateTime testDate = DateTime.Parse(form["DateConscripted"]);

**if** (testDate < DateTime.Now)

{

Pirate pirateToUpdate = db.Pirates.Find(id);

TryUpdateModel(pirateToUpdate, "", **new** string[] { "Name", "DateConscripted" });

db.SaveChanges();

**return** RedirectToAction("PiratesView");

}

**else**

{

**return** (View(db.Pirates.Find(id)));

}

}

*/// <summary>*

*/// Delete a pirate given it's id*

*/// </summary>*

*/// <param name="id">The id of the pirate to delete</param>*

*/// <returns>The pirates view page</returns>*

**public** ActionResult DeletePirate(int? id)

{

var crewsToDelete = db.Crews.Where(c => c.PirateID == id);

**foreach** (Crew c **in** crewsToDelete)

{

db.Crews.Remove(c);

}

Pirate pirateToDelete = db.Pirates.Find(id);

db.Pirates.Remove(pirateToDelete);

db.SaveChanges();

**return** RedirectToAction("PiratesView");

}

}

}

Scripts/BootyCalc.js

*// Get json booty data from server*

**function** GetBootyValues()

{

$.ajax({

type: "GET",

dataType: "json",

url: "/Home/GetBooty/",

data: { },

success: DisplayData

});

}

*// Display the json booty data*

**function** DisplayData(data)

{

document.getElementById("bootyList").innerHTML = "";

**for** (**var** index = 0; index < data.bootySize; index++)

{

**var** item = data.bootyVal[index];

**var** itemDesc = "" + item.Name + " - " + item.Booty;

document.getElementById("bootyList").innerHTML += "<li class=\"list-group-item\">" + itemDesc + "</li>";

}

}

Views/Index.cshtml

@{

ViewBag.Title = "Index";

}

**<script** type="text/javascript" src="~/Scripts/BootyCalc.js"**></script>**

**<div** class="container"**>**

**<div** class="jumbotron"**>**

**<h1>**AVAST This is yer database o pirates.**</h1>**

**<p>**

Start yer pirate searchin.

**</p>**

**</div>**

**</div>**

**<h3>**

Booty Stats!

**</h3>**

**<button** class="btn btn-default" onclick="GetBootyValues()"**>**Calculate Booty**</button>**

**<ul** id="bootyList" class="list-group"**>**

**</ul>**