IT2030 Assignment 10 Matt Brown

**ToDoViewModel.cs**

namespace ToDoList.Models

{

public class ToDoViewModel

{

public ToDoViewModel()

{

CurrentTask = new ToDo();

}

public Filters Filters { get; set; }

public List<Status> Statuses { get; set; }

public List<Category> Categories { get; set; }

public Dictionary<string, string> DueFilters { get; set; }

public List<ToDo> Tasks { get; set; }

public ToDo CurrentTask { get; set; }

}

}

**HomeController.cs**

namespace ToDoList.Controllers

{

public class HomeController : Controller

{

private ToDoContext context;

public HomeController(ToDoContext ctx) => context = ctx;

public IActionResult Index(string id)

{

ToDoViewModel model = new ToDoViewModel();

// load current filters and data needed for filter drop downs in ViewBag

var filters = new Filters(id);

model.Filters = new Filters(id);

model.Categories = context.Categories.ToList();

model.Statuses = context.Statuses.ToList();

model.DueFilters = Filters.DueFilterValues;

// get ToDo objects from database based on current filters

IQueryable<ToDo> query = context.ToDos

.Include(t => t.Category).Include(t => t.Status);

if (filters.HasCategory) {

query = query.Where(t => t.CategoryId == filters.CategoryId);

}

if (filters.HasStatus) {

query = query.Where(t => t.StatusId == filters.StatusId);

}

if (filters.HasDue) {

var today = DateTime.Today;

if (filters.IsPast)

query = query.Where(t => t.DueDate < today);

else if (filters.IsFuture)

query = query.Where(t => t.DueDate > today);

else if (filters.IsToday)

query = query.Where(t => t.DueDate == today);

}

var tasks = query.OrderBy(t => t.DueDate).ToList();

model.Tasks = tasks;

return View(model);

}

public IActionResult Add()

{

ToDoViewModel model = new ToDoViewModel();

model.Categories = context.Categories.ToList();

model.Statuses = context.Statuses.ToList();

return View(model);

}

[HttpPost]

public IActionResult Add(ToDoViewModel model)

{

if (ModelState.IsValid)

{

context.ToDos.Add(model.CurrentTask);

context.SaveChanges();

return RedirectToAction("Index");

}

else

{

model.Categories = context.Categories.ToList();

model.Statuses = context.Statuses.ToList();

return View(model);

}

}

[HttpPost]

public IActionResult Filter(string[] filter)

{

string id = string.Join('-', filter);

return RedirectToAction("Index", new { ID = id });

}

[HttpPost]

public IActionResult Edit([FromRoute]string id, ToDo selected)

{

if (selected.StatusId == null) {

context.ToDos.Remove(selected);

}

else {

string newStatusId = selected.StatusId;

selected = context.ToDos.Find(selected.Id);

selected.StatusId = newStatusId;

context.ToDos.Update(selected);

}

context.SaveChanges();

return RedirectToAction("Index", new { ID = id });

}

}

}

**Add.cshtml**

@model ToDoViewModel

<h2>New Task</h2>

<div asp-validation-summary="All" class="text-danger"></div>

<form asp-action="Add" method="post">

<div class="form-group">

<label asp-for="CurrentTask.Description">Description:</label>

<input type="text" asp-for="CurrentTask.Description" class="form-control">

</div>

<div class="form-group">

<label asp-for="CurrentTask.CategoryId">Category:</label>

<select asp-for="CurrentTask.CategoryId" class="form-control"

asp-items="@(new SelectList(Model.Categories, "CategoryId", "Name"))">

<option value=""></option>

</select>

</div>

<div class="form-group">

<label asp-for="CurrentTask.DueDate">Due Date:</label>

<input type="text" asp-for="CurrentTask.DueDate" class="form-control">

</div>

<div class="form-group">

<label asp-for="CurrentTask.StatusId">Status:</label>

<select asp-for="CurrentTask.StatusId" class="form-control"

asp-items="@(new SelectList(Model.Statuses, "StatusId", "Name"))">

<option value=""></option>

</select>

</div>

<button type="submit" class="btn btn-primary">Add</button>

<a asp-action="Index" class="btn btn-primary">Cancel</a>

</form>

**Index.cshtml**

@model ToDoViewModel

@{

string Overdue(ToDo task) => task.Overdue ? "bg-warning" : "";

}

<div class="row">

<div class="col-sm-2">

<form asp-action="Filter" method="post">

<div class="form-group">

<label>Category:</label>

<select name="filter" class="form-control"

asp-items="@(new SelectList(Model.Categories,

"CategoryId", "Name", Model.Filters.CategoryId))">

<option value="all">All</option>

</select>

</div>

<div class="form-group">

<label>Due:</label>

<select name="filter" class="form-control"

asp-items="@(new SelectList(Model.DueFilters,

"Key", "Value", Model.Filters.Due))">

<option value="all">All</option>

</select>

</div>

<div class="form-group">

<label>Status:</label>

<select name="filter" class="form-control"

asp-items="@(new SelectList(Model.Statuses,

"StatusId", "Name", Model.Filters.StatusId))">

<option value="all">All</option>

</select>

</div>

<button type="submit" class="btn btn-primary">Filter</button>

<a asp-action="Index" asp-route-id="" class="btn btn-primary">Clear</a>

</form>

</div>

<div class="col-sm-10">

<a asp-action="Add"><b>Add new task</b></a>

<table class="table table-bordered table-striped mt-2">

<thead>

<tr>

<th>Description</th>

<th>Category</th>

<th>Due Date</th>

<th>Status</th>

<th class="w-25"></th>

</tr>

</thead>

<tbody>

@foreach (ToDo task in Model.Tasks)

{

string overdue = Overdue(task);

<tr>

<td>@task.Description</td>

<td>@task.Category.Name</td>

<td class="@overdue">@task.DueDate?.ToShortDateString()</td>

<td class="@overdue">@task.Status.Name</td>

<td>

<form asp-action="Edit" method="post"

asp-route-id="@Model.Filters.FilterString"

class="mr-2">

<input type="hidden"

name="@nameof(ToDo.Id)" value="@task.Id" />

<button type="submit"

name="@nameof(ToDo.StatusId)" value="closed"

class="btn btn-primary btn-sm">Completed

</button>

<button type="submit"

class="btn btn-primary btn-sm">Delete

</button>

</form>

</td>

</tr>

}

</tbody>

</table>

</div>

</div>