**Notes from “Better Device List View” video**

ctrl-k, ctrl-d will format your code

F12 open definition

Goals: 1) Display Device link only if user logged in

2) Add a Details button to the device list view

3) Add a Claim button to the device view and the details view

**1) Display Device link only if user logged in**

WebUi/Views/Home/Index.cshtml

removed stub code and added conditional code

@{

ViewBag.Title = "Home Page";

}

<div class="jumbotron">

@if (!Request.IsAuthenticated)

{

<h2>Please log in to the Customer Care Center</h2>

}

else

{

<h2>Welcome to the Customer Care Center</h2>

}

</div>

WebUi/Views/Shared/\_Layout.cshtml

Change header bar to “Customer Care Center” @Html.ActionLink("Customer Care Center",

Delete About and Contact links

Add Devices link

@if (Request.IsAuthenticated)

{

<li>@Html.ActionLink("Devices", "Index", "Device")</li>

}

Change footer to <p>&copy; @DateTime.Now.Year - Customer Care Center, Admn Insurance Company</p>

**2) Add a Details button to the device list view**

WebUi/Views/Device/Index.cshtml

Changed title from Device to Devices

Changed tag line from “Devices covered by your policy” to “Your insured devices”

Removed column labels

Removed serial number

Removed bullet point &#8226;&nbsp;

Removed style="width:50px" from button column

Added Details button

@model EDeviceClaims.WebUi.Models.DeviceListViewModel

@{

ViewBag.Title = "Devices";

}

<h2>@ViewBag.Title.</h2>

<h3>Your insured devices.</h3>

<hr />

<table>

@foreach (var thing in Model)

{

<tr>

<td style="width:200px">@thing.Name</td>

<td>

@Html.ActionLink("Details", "Details", "Device", new { @id = thing.PolicyId }, new { @class = "btn btn-default" })

@Html.Partial("\_ClaimButtons", thing) 🡸 actually pertains to goal 3

</td>

</tr>

}

</table>

WebUi/Controllers/DeviceController.cs

added using System using System;

added method:

The domain model already existed, but we needed to add a new method GetById()

public ActionResult Details(Guid id)

{

var domainModel = \_policyService.GetById(id); 🡸 this is a new method

var model = new DeviceViewModel(domainModel);

The view model also already existed

return View(model);

}

Domain/Services/Class1.cs

add method signature in IPolicyService PolicyDomainModel GetById(Guid id); 🡸so you have to add it to the interface

implement the method in PolicyService:

public PolicyDomainModel GetById(Guid id) 🡸and the class

{

var entity = GetPolicyInteractor.GetById(id);

return entity == null ? null : new PolicyDomainModel(entity);

}

We needed to make a new view to display the fields in the existing view model

WebUi/Views/Device/Details.cshtml

create new file Details.cshtml with the following content:

@model EDeviceClaims.WebUi.Models.DeviceViewModel

@{

ViewBag.Title = "Device";

}

<h2>@ViewBag.Title.</h2>

<h3>Device details.</h3>

<hr />

<span style="color:#616fe0"><strong>Device</strong></span>&nbsp;&nbsp;@Model.Name<br /><br />

<span style="color:#616fe0"><strong>Serial Number</strong></span>&nbsp;&nbsp;@Model.SerialNumber<br /><br />

<span style="color:#616fe0"><strong>Device ID</strong></span>&nbsp;&nbsp;@Model.PolicyId<br /><br />

<span style="color:#616fe0"><strong>Policy Number</strong></span>&nbsp;&nbsp;@Model.PolicyNumber

**3) Add a Claim button to the device view and the details view**

WebUi/Views/Device/Details.cshtml

add a Claim button:

@model EDeviceClaims.WebUi.Models.DeviceViewModel

@{

ViewBag.Title = "Device";

}

<h2>@ViewBag.Title.</h2>

<h3>Device details.</h3>

<hr />

<span style="color:#616fe0"><strong>Device</strong></span>&nbsp;&nbsp;@Model.Name<br /><br />

<span style="color:#616fe0"><strong>Serial Number</strong></span>&nbsp;&nbsp;@Model.SerialNumber<br /><br />

<span style="color:#616fe0"><strong>Device ID</strong></span>&nbsp;&nbsp;@Model.PolicyId<br /><br />

<span style="color:#616fe0"><strong>Policy Number</strong></span>&nbsp;&nbsp;@Model.PolicyNumber<br /><br />

@Html.ActionLink("Go Back", "Index", "Device", new { @class = "btn btn-default" })

@Html.Partial("\_ClaimButtons", Model)

WebUi/Views/Device/\_ClaimButton.cshtml

create a new file \_ClaimButton.cshtml with the following content:

@model EDeviceClaims.WebUi.Models.DeviceViewModel

@if (!Model.HasClaim)

{

@Html.ActionLink("Start a Claim", "Create", "Claim", new { @id = Model.PolicyId }, new { @class = "btn btn-default" })

}

else

{

@Html.ActionLink("Check Claim Status", "Status", "Claim", new { @id = Model.PolicyId }, new { @class = "btn btn-default" })

}

WebUi/Models/DeviceViewModel.cs

Add property public bool HasClaim { get; set; }

WebUi/Controllers/ClaimController.cs

create a new file ClaimController.cs with the following content:

has a Create() method and a Status() method . . .

Domain/Services/ClaimService.cs

Create a new file ClaimService.cs with the following content:

Domain/Models/ClaimDomainModel.cs

Create new file ClaimDomainModel.cs with the following content:

WebUi/Models/ClaimViewModel.cs

Create new file ClaimViewModel.cs with the following content:

WebUi/Views/Claim/Create.cshtml

Create a new file Create.cshtml with the following content:

Interactors/CreateClaimInteractor.cs

Create a new file CreateClaimInteractor.cs with the following content:

Entities/ClaimEntity.cs

Create a new file ClaimEntity.cs with the following content:

Repositories/ClaimRepository.cs

Create a new file ClaimRepository.cs with the following content:

Repositories/Contexts/EDeviceClaimsContext.cs

Add a new property public DbSet<ClaimEntity> Claims { get; set; }

Package Manager Console Repositories

PM> add-migration –m “createClaimsEntity” this creates class datetime\_createClaimsEntity.cs in Repositories/Migrations

PM> add-migration createClaimsEntity

No migrations configuration type was found in the assembly 'EDeviceClaims.WebUi'. (In Visual Studio you can use the Enable-Migrations command from Package Manager Console to add a migrations configuration).

PM> Enable-Migrations -EDeviceClaims.WebUi

No context type was found in the assembly 'EDeviceClaims.WebUi'.

When I try to build I get:

*The model backing the 'EDeviceClaimsContext' context has changed since the database was created.*

PM> update-database this creates a Claims table in WebUi/App\_Data, refresh circular arrow, see e\_device\_claims\_dev.mdf, double click it, see Server Explorer open, open Tables folder, see claims table (which is new?)