**Beschrijving opdrachten**

**Makkelijke Oefeningen:**

Oefening 1 (Models, Views, Controllers):

Stappen  
- Geef het model genaamd "Oefening1Model" onder Models/Oefening1Model 3 properties met getters en setters.  
- Maak de constructor af in het model.  
- Voeg in de controller genaamd "Oefening1Controller" de using   
"using Makkelijke\_Oefeningen.Models;" toe.  
- Maak in de controller een instantie van het "Oefening1Model" en geef hier waardes aan mee.  
- Geef deze instantie mee aan de "return View()".  
- Zet in de view "Index.cshtml" onder Views/Oefening1/Index.cshtml de namen van de gemaakte properties in de <th> tags van de tabel.  
- Zet in deze view ook de waardes van de variabelen in door middel van een "@Model.{Naam van propertie}" in de <td> tags van de tabel.  
- Maak een view aan onder Views/Oefening1. Deze kan genoemd worden hoe jij zelf wilt, en zet hier wat html in bijvoorbeeld "<h2>Test</h2>" om iets op het scherm te kunnen tonen.  
- Voeg hierna in de view "Index.cshtml" onder Views/Oefening1 in de <h2> tags "@Html.ActionLink("Test", "{Naam van de nieuwe view}", "Oefening1")" toe.  
- Als laatst moet je in de controller genaamd "Oefening1Controler" een nieuw ActionResult aanmaken met als naam de naam van de view die je hebt aangemaakt.

Oefening 2 (Get en Post):

Stappen  
- Zet in de controller genaamd “Oefening2Controller” boven de ActionResult genaamd “Index” de tag “[HttpGet]”.  
- Geef een nieuwe instantie van het OefeningModel mee aan de view met waardes die je zelf bepaald.  
- Maak ik de conroller ook een nieuw ActionResult aan genaamd “Index” en geef als paramaters 3 strings mee genaamd “username, password en email” deze moeten alle 3 strings zijn string.   
- Geef aan de view een nieuwe instantie van het OefeningModel mee waarin de waardes van de parameters worden gezet.  
- Plaats boven deze methode de tag “[HttpPost]”.  
- Plaatst in de view genaamd “Index.cshtml” onder Views/Oefening1 de waardes van de variabelen met “@Model.{Naam van propertie}”

Oefening 3 (Razor):

Stappen  
- Plaatst in de view genaamd “Index.cshtml” onder View/Oefening3 in de eerste @{} tags een array van integers met lengte die je zelf bepaald.  
- Vul de array met nummers maakt niet uit welke.  
- Plaats in de tweede tags een loop waarin je de tabel vult met de nummers in de array die je net hebt aangemaakt met @{Naam van variabele}  
- Zet in de @{} tag in de view genaamd “List.cshtml” onder View/Oefening3 een loop waarin word geloop door het model.  
- Toon de waardes in de tabel

Oefening 4 (Html helpers):

Stappen  
- Plaats in de view genaamd “Index.cshtml” onder View/Oefening4 de regel “@Html.Textbox(“{Waarde de jij wilt}”)”. Hierdoor zal er een textbox worden getoond wanneer de pagina wordt geladen.  
- Maak in het project een map aan genaamd “Helpers”.  
- Plaats in deze map een class genaamd “LabelHelper”.  
- Voeg een static methode toe genaamd “Label” die een HtmlString teruggeeft met de parameters string target en string text.  
- Geef een instantie van een HtmlString terug met daarin de html voor een label waarin de waardes van de parameters worden gezet.  
- Voeg bovenaan in de view genaamd “Index.cshtml” onder View/Oefening4 de using   
“using Makkelijke\_Oefeningen.Helpers” toe.  
- Plaats in de view de regel “@LabelHelper.Label(“{Eeerste waarde}”, “{Tweede waarde}”)”.  
- Test of er een label verschijnt wanneer de view geladen wordt.  
- Voeg in de LabelHelper onder Helpers de using “using System.Web.Mvc;” toe.  
- Voeg een static methode toe genaamd “Label2” die een MvcHtmlString teruggeeft met de parameters string target en string text.  
- Geef een MvcHtmlString terug met daarin de html voor een label waarin de waardes van de parameters worden gezet. Door de regel “MvcHtmlString.Create({Html met waardes})”.  
- Plaats in de view genaamd “Index.cshtml” onder View/Oefening4 de regel “@LabelHelper.Label2(“{Eeerste waarde}”, “{Tweede waarde}”)”.  
- Test of er een tweede label verschijnt wanneer de view geladen wordt.

**Moeilijkere Oefeningen:**

Oefening 1 (jQuery Ajax Get en Post):

Stappen  
- Plaats in de controller genaamd “Oefening1Controller” een methode genaamd “AjaxGet” die een JsonResult teruggeeft.  
- Voeg de tag “[HttpGet]” toe boven deze methode.  
- Maak een instantie aan van het OefenModel en geef deze waardes. Welke waardes maakt niet uit.  
- Geef deze instantie terug met de regel   
“return Json({Naam van instantie}, JsonRequestBehavior.AllowGet)”.  
- Plaats in deze controller ook een methode genaamd “AjaxPost” die een JsonResult teruggeeft en de parameters (string username, string password, string email) ontvangt.  
- Voeg de tag “[HttpPost]” toe boven deze methode.  
- zet hierin de regel  
“string html = String.Format("<td>{0}</td><td>{1}</td><td>{2}</td>", username, password, email);”  
- Geef deze string terug met de regel   
“return Json({Naam van string}, JsonRequestBehavior.AllowGet)”.  
- Plaats in de view genaamd “Get.cshtml”onder Views/Oefening1 in de <input> tag “id=btnGet” dit zal er voor zorgen dat de jquery functie die je onderaan in de html ziet staan zal worden aangeroeen als je op de knop klikt.  
- Vul in de jQuery functie bij type: “Get” in, bij url: “/Oefening1/AjaxGet” en bij datatype: “json”.  
- Test het resultaat als het goed is zal de knop van tekst veranderen als je er op klikt.  
- Plaats in de view genaamd “Post.cshtml”onder Views/Oefening1 bij data:   
'{username: "' + $("#txb\_Username").val() + '", password: "' + $("#txb\_Password").val() + '", email: "' + $("#txb\_Email").val() + '"}'.  
- Test het resultaat als het goed is zal als je op de knop klikt de ingevuld waardes in de textboxen worden gezet in de tabel.

Oefening 2 (Partial Loading met en zonder Ajax):

Stappen  
- Plaats in de controller genaamd “Oefening2Controller” een methode genaamd “RenderPartial” die een ActionResult teruggeeft.  
- Geef in deze methode een partial view terug met de regel  
“return PartialView("\_TablePartial");”  
- Aan de PartailView zou je als tweede parameter ook een model van een object mee kunnen geven. Maar aangezien het model dat we gebruiken in de view hetzelfde is als in de partial view hoet dat in deze instantie niet.  
- Plaats in de view genaamd “Index.cshtml” onder Views/Oefening2 in de eerste <div> tag de regel  
@Html.Partial("\_TablePartial").  
- Via deze manier kun je een partial inladen met een html helper.  
- Plaats in de jQuery function onderaan in de html bij datatype: “html”.  
- Via deze manier kun je een partial inladen met een Ajax call.  
- Test het resultaat er wordt standaard één keer de tabel ingeladen die in de partial view staat. Als je op de knop klikt zou er nog een tabel moeten verschijnen door middel van een Ajax call.

Oefening 3 (Sessions):

Stappen  
- Plaats in de controller genaamd “Oefening3Controller” in de methode Login de regel   
“Session["OefeningModel"] = oefeningModel;” en vervang de null naar de naam van de instantie die je wilt teruggeven.  
- Maak ik de methode SessionView een instantie van het OefeningModel genaamd “oefeningModel” en zet hier de waarde in van de session die je in de andere methode aan hebt gemaakt door deze te casten naar een OefeningModel.  
- Geef de instantie mee terug aan de view.  
- Test het resultaat door waardes in te vullen in de pagina en daarna op login te klikken. Klik dan vervolgens op de link “Go To Session” en kijk of de waardes die jij net hebt ingevuld in de pagina staan.