Orders afwerken

We hebben er in voorgaande paragraaf voor gezorgd dat ons winkelmandje uiteindelijk werd omgezet in een bestelling.
Wanneer in de pagina Confirmation.cshtml op de knop “Bestelling bevestigen” werd geklikt hebben we er in de OnPost() voor gezorgd dat :

- Er in de tabel Order 1 record werd aangemaakt met daarin het Id van de klant (UserId) die de bestelling plaatste, het moment (DateTimeStamp) waarop de bestelling werd geplaatst en werd aan het betrokken record een autonummeringswaarde toegekend als primaire sleutel (Id).
- Er in de tabel OrderDetail evenveel records werden toegevoegd als er verschillende artikels werden besteld (dus in het winkelmandje) en aan elk record in de OrderDetail tabel werden volgende zaken toegekend :
    - Een primaire sleutel Id (autonummering)
    - Een verwijzing naar de bestelling waar dit record deel van uit maakt (OrderId)
    - Het Id van het artikel dat besteld werd (ArticleId)
    - Het aantal van het artikel dat besteld werd (Count)
    - De verkoopsprijs zoals die was op het moment van de bestelling (SalesPrice)

We gaan er nu voor zorgen dat een beheerder van onze website de bestellingen kan raadplegen. Om rechtstreeks vanuit het menu bovenaan (in _Layout) deze bestellingen te kunnen raadplegen voegen we aan deze pagina dus een extra menu item toe.

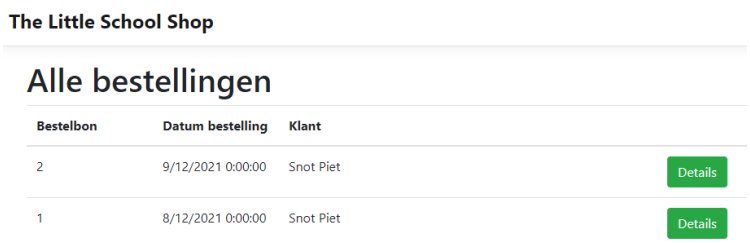
/Pages/Shared/_Layout.cshtml

Het menu item voegen we toe onder de links naar merken en categoriën (+/- regel 38) :

```
        <div class="dropdown-menu" aria-labelledby="navbarDropdown">
            <a class="dropdown-item" href="/brands/index">Merken</a>
            <a class="dropdown-item" href="/Categories/index">Categoriën</a>
            <a class="dropdown-item" href="/Orders/index">Bestellingen</a>
        </div>
```

/Pages/Orders/Index.cshtml

In deze pagina gaan we voor onze beheerders een overzicht aanbieden van alle bestellingen, aflopend gesorteerd op het bestelbonnummer (= het veld Order.Id).
We voorzien hier geen mogelijkheden om nieuwe bestellingen toe te voegen of bestellingen te wijzigen of te verwijderen, enkel om bestellingen te raadplegen.



Neem in deze pagina de (ondertussen gekende) onderstaande code over :

```
@page
@model SchoolShop.Pages.Orders.IndexModel

@{
    ViewData["Title"] = "Bestellingen";
    ViewData["Email"] = "";
    ViewData["Loginstyle"] = "visibility: visible; ";
    ViewData["Logoutstyle"] = "visibility: hidden; ";
    ViewData["Configstyle"] = "visibility: hidden; ";
    if (!string.IsNullOrEmpty(Model.Availability.Email))
    {
        ViewData["Email"] = Model.Availability.Email;
        ViewData["BasketCount"] = Model.Availability.BasketCount;
        ViewData["Loginstyle"] = "visibility: hidden; ";
        ViewData["Logoutstyle"] = "visibility: visible; ";
        if (Model.Availability.IsAdmin)
        {
            ViewData["Configstyle"] = "visibility: visible; ";
        }
    }
}


<h1>Alle bestellingen</h1>

<table class="table">
    <thead>
        <tr>
            <th>Bestelbon</th>
            <th>
                Datum bestelling
            </th>
            <th>
                Klant
            </th>

            <th></th>
        </tr>
    </thead>
    <tbody>
        @foreach (var item in Model.Order)
        {
        <tr>
            <td class="col-2">
                @Html.DisplayFor(modelItem => item.Id)
            </td>
            <td class="col-2">
                @Html.DisplayFor(modelItem => item.DateTimeStamp)
            </td>
            <td class="col-6">
                @Html.DisplayFor(modelItem => item.User.Name)
                @Html.DisplayFor(modelItem => item.User.FirstName)
            </td>
            <td class="col-2">
                <a asp-page="./Details" asp-route-id="@item.Id" class="btn btn-
success">Details</a>
            </td>
        </tr>
        }
    </tbody>
</table>
```

Je zal uiteraard een aantal fouten krijgen, maar dat komt omdat we in de bijhorende CS file (Index.cshtml.cs) nog geen (of de juiste) code hebben voorzien.
Let ondertussen wel al op de knop waarmee de gebruiker de details van de bestelling zal kunnen bekijken : we voorzien het attribuut asp-route-id wat er voor zal zorgen dat we straks in de Details.cshtml.cs pagina in het OnGet event het id zullen kunnen opvragen van deze bestelling

zodat we daar dan zullen weten welke records er uit de tabel OrderDetail moeten opgevraagd worden.

/Pages/Orders/Index.cshtml.cs

In deze pagina gebeurt er geen postback (commit), er is enkel de detail-knop die niets anders is dan een hyperlink naar een andere pagina (/Details.cshtml) : we hebben hier dus enkel een OnGet() methode nodige.
We zorgen er voor dat – net als bij vele andere pagina's – dat wanneer we merken dat de ingelogde gebruiker geen beheerder is, we onmiddellijk de bezoeker naar de artikel-pagina brengen.
In de OnGet() methode gaan we de property Order (een List<Order>) vullen met alle bestelling aflopend gesorteerd op het Id van de bestellingen.
We zorgen er ook voor dat naast de pure order-gegevens (inhoud van de records van de tabel Order) er meteen ook de gegevens van de klant (User) opgenomen worden.

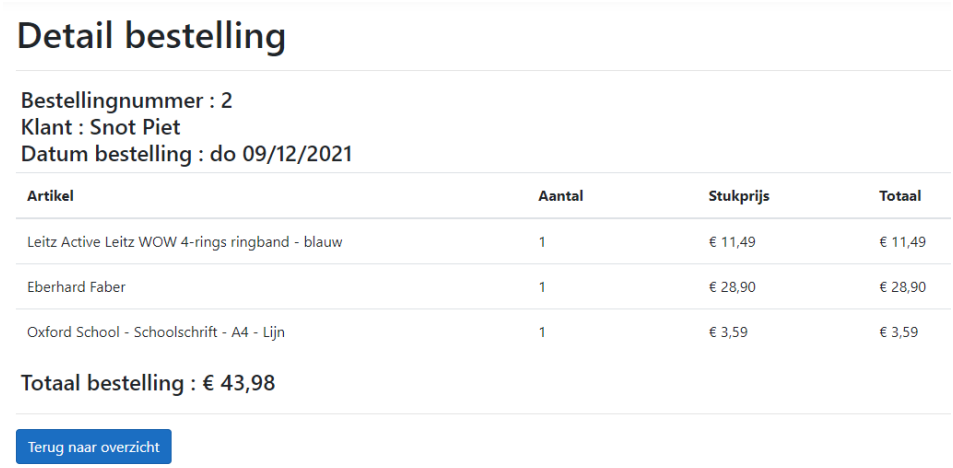
Vervang de volledige code door onderstaande :

```
    public class IndexModel : PageModel
    {
        private readonly SchoolShop.Data.SchoolShopContext _context;
        public Availability Availability { get; set; }
        public IndexModel(SchoolShop.Data.SchoolShopContext context)
        {
            _context = context;
        }

        public IList<Order> Order { get;set; }

        public void OnGet()
        {
            Availability = new Availability(_context, HttpContext);
            if (!Availability.IsAdmin)
            {
                RedirectToPage("../Articles/Index");
            }
            IQueryable<Order> query = _context.Order
                .Include(b => b.User)
                .OrderByDescending(b=>b.DateTimeStamp);
            Order = query.ToList();
        }
    }
```

/Pages/Orders/Details.cshtml



Detail bestelling

Bestellingnummer : 2
Klant : Snot Piet
Datum bestelling : do 09/12/2021

Artikel	Aantal	Stukprijs	Totaal
Leitz Active Leitz WOW 4-rings ringband - blauw	1	€ 11,49	€ 11,49
Eberhard Faber	1	€ 28,90	€ 28,90
Oxford School - Schoolschrift - A4 - Lijn	1	€ 3,59	€ 3,59

Totaal bestelling : € 43,98

Terug naar overzicht

Ook de details.cshtml pagina is opnieuw een opbouw die we kennen uit het verleden.
Let er op dat we helemaal bovenaan deze pagina een drietal variabelen declareren :

- Displaydata : hiermee vragen we dat DateTimeStamp prop op die van het type DateTime is en we converteren die (met de nodig formaat karakters) naar een string die straks een keurige datum op ons scherm laat afbeelden
- lineTotal : met deze variabele gaan we lijn per lijn gaan berekenen wat het resultaat is van de totale lijn (aantal * prijs).
- orderTotal : met deze variabele gaan we het totaal van de bestelling laten berekenen.

De volledige code van deze pagina :

```
@page
@model SchoolShop.Pages.Orders.DetailsModel

@{
    ViewData["Title"] = "Detail bestelling";
    ViewData["Email"] = "";
    ViewData["Loginstyle"] = "visibility: visible; ";
    ViewData["Logoutstyle"] = "visibility: hidden; ";
    ViewData["Configstyle"] = "visibility: hidden; ";
    string displaydate = Model.Order.DateTimeStamp.ToString("ddd dd/MM/yyyy");
    decimal lineTotal = 0M;
    decimal orderTotal = 0M;
    if (!string.IsNullOrEmpty(Model.Availability.Email))
    {
        ViewData["Email"] = Model.Availability.Email;
        ViewData["BasketCount"] = Model.Availability.BasketCount;
        ViewData["Loginstyle"] = "visibility: hidden; ";
        ViewData["Logoutstyle"] = "visibility: visible; ";
        if (Model.Availability.IsAdmin)
        {
            ViewData["Configstyle"] = "visibility: visible; ";
        }
    }
}


<h1>Detail bestelling</h1>

<div>
    <hr />
    <div class="row h4" style="padding-left:20px;">
        Bestellingnummer : @Html.DisplayFor(model => model.Order.Id)
        <br />
        Klant : @Html.DisplayFor(model => model.Order.User.Name)
```

```
        @Html.DisplayFor(model => model.Order.User.FirstName)
        <br />
        Datum bestelling : @displaydate
    </div>
    <table class="table">
        <thead>
            <tr>
                <th>
                    Artikel
                </th>
                <th>
                    Aantal
                </th>
                <th>
                    Stukprijs
                </th>
                <th>
                    Totaal
                </th>
            </tr>
        </thead>
        <tbody>
            @foreach (var item in Model.OrderDetails)
            {
                lineTotal = item.SalesPrice * item.Count;
                <tr>
                    <td class="col-6">
                        @Html.DisplayFor(modelItem => item.Article.ArticleName)
                    </td>

                    <td class="col-2">
                        @Html.DisplayFor(modelItem => item.Count)
                    </td>
                    <td class="col-2">
                        @Html.DisplayFor(modelItem => item.SalesPrice)
                    </td>
                    <td class="col-2">
                        € @lineTotal.ToString("#,##0.00")
                    </td>
                </tr>
        orderTotal += lineTotal;
    }
        </tbody>
    </table>
    <div class="row h4" style="padding-left:20px;">
        Totaal bestelling : € @orderTotal.ToString("#,##0.00")
        <hr />
    </div>
</div>
<hr />
<div>
    <a asp-page="./Index" class="btn btn-primary">Terug naar overzicht</a>
</div>
```

/Pages/Orders/Details.cshtml.cs

Je herinnert je nog dat wanneer we in de Index pagina op de knop details klikten we het ID van de betrokken bestelling gingen doorgeven : `asp-route-id="@item.Id"`

In de OnGet methode van deze pagina komt deze waarde dus binnen als parameter (int? id) .
We gebruiken deze waarde (= het Id van de betrokken bestelling) om enkel die detaillijnen af te beelden die deel uitmaken van de betrokken bestelling.
Omdat we op onze pagina ook nog eens alle globale bestellingsgegevens wensen af te beelden (dus de inhoud van het record uit Order) vragen we dat record ook nog even op.

```
public class DetailsModel : PageModel
{
    private readonly SchoolShop.Data.SchoolShopContext _context;
    public Availability Availability { get; set; }
    public DetailsModel(SchoolShop.Data.SchoolShopContext context)
    {
        _context = context;
    }

    public Order Order { get; set; }
    public List<OrderDetail> OrderDetails { get; set; }

    public IActionResult OnGet(int? id)
    {
        Availability = new Availability(_context, HttpContext);
        if (!Availability.IsAdmin)
        {
            return RedirectToPage("../Articles/Index");
        }
        if (id == null)
        {
            return NotFound();
        }

        Order = _context.Order
            .Include(o => o.User).FirstOrDefault(m => m.Id == id);
        if (Order == null)
        {
            return NotFound();
        }
        IQueryable<OrderDetail> query = _context.OrderDetail
            .Include(b => b.Article);
        query = query.Where(b => b.OrderId.Equals(Order.Id));
        OrderDetails = query.ToList();



        return Page();
    }
}
```