

Web-aplikacije

ASP.NET Core MVC

2017/18.13

Primjer zaglavlje-stavke
Izrada izvješća

Napomena uz master-detail primjer

2

- Primjer za master-detail predstavlja složene koncepte koji uključuju korištenje CSS stilova i *jQueryja*
- Primjer je opsežan te pojedini dijelovi koda ne mogu u potpunosti stati na slajdove te se preporuča isprobati primjer i detaljno proučiti programski kôd pri čemu slajdovi mogu poslužiti kao nit vodilja kojim redom proučavati primjer

Pogled za prikaz svih dokumenata

3

➡ Kao i u primjeru s partnerima za dohvat svih dokumenata koristi se pogled iz baze podataka

➡ Značajno pojednostavljuje kôd za dohvat podataka u upravljaču

```
CREATE VIEW [dbo].[vw_Dokumenti] AS
SELECT  dbo.Dokument.IdDokumenta, dbo.Dokument.VrDokumenta,
        dbo.Dokument.BrDokumenta, dbo.Dokument.DatDokumenta,
        dbo.Dokument.IdPartnera, dbo.Dokument.IdPrethDokumenta,
        dbo.Dokument.PostoPorez, dbo.Dokument.IznosDokumenta,
        CASE dbo.Partner.TipPartnera
            WHEN 'O' THEN dbo.Osoba.PrezimeOsobe + ', ' + dbo.Osoba.ImeOsobe
            ELSE dbo.Tvrtka.NazivTvrtke END
            + ' (' + dbo.Partner.OIB + ')' AS NazPartnera
FROM    dbo.Dokument INNER JOIN dbo.Partner ON dbo.Dokument.IdPartnera =
        dbo.Partner.IdPartnera
LEFT OUTER JOIN dbo.Osoba
        ON dbo.Partner.IdPartnera = dbo.Osoba.IdOsobe
LEFT OUTER JOIN dbo.Tvrtka
        ON dbo.Partner.IdPartnera = dbo.Tvrtka.IdTvrtke
```

(Podsjetnik) Dodavanje pogleda u EF model (1)

4

➡ Potrebno definirati razred koji svojoj strukturom odgovara rezultatu pogleda


➡ Svojstva koja imaju set dio, a ne odgovaraju nekom stupcu iz rezultata označavaju se atributom *NotMapped*

➡ Primjer:  Web \ Firma.Mvc \ ViewModels \ ViewDokumentInfo.cs

```
public class ViewDokumentInfo {  
    public int IdDokumenta { get; set; }  
    public decimal PostoPorez { get; set; }  
    public int? IdPrethDokumenta { get; set; }  
    public DateTime DatDokumenta { get; set; }  
    public int IdPartnera { get; set; }  
    public string NazPartnera { get; set; }  
    public decimal IznosDokumenta { get; set; }  
    public string VrDokumenta { get; set; }  
    public int BrDokumenta { get; set; }  
  
    [NotMapped]  
    public int Position { get; set; } //Position in result  
}
```

(Podsjetnik) Dodavanje pogleda u EF model (2)

5

- U definiciju konteksta dodati novi *DbSet* za pogled i informaciju koje svojstvo jednoznačno određuje pojedini podatak iz novog skupa.
- Primjer:  Web \ Firma.Mvc \ Models \ FirmaContext.cs
 - Primijetiti da naziv svojstva u ovom primjeru ne odgovara nazivu pogleda (vidi slajdove koji slijede)

```
public partial class FirmaContext : DbContext {  
    ...  
    public virtual DbSet<ViewDokumentInfo> ViewDokumentInfo { get; set; }  
    ...  
    protected override void OnModelCreating(ModelBuilder modelBuilder) {  
        modelBuilder.Entity<ViewDokumentInfo>(entity =>  
            {  
                entity.HasKey(e => e.IdDokumenta);  
            });  
    }  
}
```

Korištenje vlastitog SQL upita za dohvat podataka

6

- Naziv pogleda nije isti kao naziv svojstva u kontekstu pa prilikom dohvata treba navesti odgovarajući SQL upit

➤ Primjer:  Web \ Firma.Mvc \ Controllers \ DokumentController.cs

```
public class DokumentController : Controller {  
    private readonly FirmaContext ctx;  
    private readonly AppSettings appData;  
    public DokumentController(FirmaContext ctx,  
                             IOptionSnapshot<AppSettings> options) {  
        this.ctx = ctx;  
        appData = options.Value;  
    }  
    public IActionResult Index(...) {  
        var query = ctx.ViewDokumentInfo.AsNoTracking()  
            .FromSql("SELECT * FROM vw_dokumenti");  
    }  
}
```

- Tablični prikaz dokumenata u pogledu s ikonama za ažuriranje i brisanje izveden kao i u ostalim primjerima (npr. prikaz država)
 - Model sadrži podatke o dokumentima te informacije o stranicenju i sortiranju te sadržajem po kojem treba filtrirati podatke

Filtriranje podataka

7

- Popis dokumenata moguće filtrirati po partneru, iznosu ili datumu.
- Za odabir partnera koristi se nadopunjavanje, a za odabir datuma prikazuje se kalendar
 - Izvedeno korištenjem alata jQueryUI (*autocomplete* i *datepicker*)
- Gumb na formi za unos kriterija šalje popunjene podatke na akciju *Filter* koja spaja kriterije u jedan string koji se šalje kao parametar akciji Index
 - Primjerice za odabir sa slike i aktiviranje filtera vrijednost parametra filter bit će `filter=0-01.01.2015-10.05.2017-300,00-500,00`
- Gumb za uklanjanje filtra je poveznica na akciju Indeks bez parametara
- Kriterij pretrage parcijalni pogled uključen u pogled Indeks
 - `@Html.Partial("KriterijPretrage", Model.Filter)`

IdPartnera: 0, Zadovoljni korisnik

Iznos: 300,00 - 500,00

Datum: 01.01.2015. - [Calendar: Svibanj 2017, 10 selected]

Iznos dokumenta

326,25 kn	[edit]	[delete]
382,50 kn	[edit]	[delete]
425,00 kn	[edit]	[delete]

1

Parametar ili sjednica?

8

- Prednosti korištenja paramet(a)ra za informaciju o filtriranju
 - Moguće imati nekoliko aktivnih filtara unutar više kartica istog preglednika
 - Moguće pohraniti poveznicu za buduće posjete
 - Informacija se ne gubi istekom sjednice
 - Može se koristiti ako server koristi *load balancing*
- Mane:
 - Potreba za korištenjem više parametara, odnosno pronalaskom efikasnog načina za spremanje više kriterija unutar istog stringa
 - Potrebno prenositi parametre u različite akcije
 - vidi primjer s državama i parametrima `page, sort ascending`
 - Treba obratiti pažnju na znakove koji mogu utjecati na rekonstrukciju vrijednosti pojedinog filtra
 - Prevelik broj kriterija i dugi tekstovi kriterija mogli bi uzrokovati adresu izvan raspona dopuštene duljine
 - Alternativa: kriterij pohraniti negdje (sjednica, BP) i pridružiti mu neku vrijednost koja bi se koristila unutar adrese

Razred za informacije o filteru (1)

9

➡ Za prihvatanje podataka o filteru koristi se razred *DokumentFilter*

➡ Osim svojstava za prihvatanje unesenih vrijednosti sadrži i nekoliko pomoćnih metoda kojima se olakšava rad s filtriranjem

➡ Primjer:  Web \ Firma.Mvc \ ViewModels \ DokumentFilter.cs

```
public class DokumentFilter {  
    public int? IdPartnera { get; set; }  
    public string NazPartnera { get; set; }  
    public DateTime? DatumOd { get; set; }  
    public DateTime? DatumDo { get; set; }  
    public decimal? IznosOd { get; set; }  
    public decimal? IznosDo { get; set; }  
    public bool IsEmpty() {  
        bool active = IdPartnera.HasValue  
            || DatumOd.HasValue || DatumDo.HasValue  
            || IznosOd.HasValue || IznosDo.HasValue;  
        return !active;  
    }  
    ...  
}
```

Razred za informacije o filtru (2)

10

➡ Uneseni podaci mogu se spojiti u string te rekonstruirati iz stringa

➡ Primjer:  Web \ Firma.Mvc \ ViewModels \ DokumentFilter.cs

```
public class DokumentFilter {  
    ...  
    public override string ToString() {  
        return string.Format("{0}-{1}-{2}-{3}-{4}",  
            IdPartnera, DatumOd?.ToString("dd.MM.yyyy"),  
            DatumDo?.ToString("dd.MM.yyyy"), IznosOd, IznosDo);  
    }  
    public static DokumentFilter FromString(string s) {  
        var filter = new DokumentFilter();  
        var arr = s.Split(new char[] { '-' }, StringSplitOptions.None);  
        filter.IdPartnera = string.IsNullOrEmpty(arr[0]) ?  
            new int?() : int.Parse(arr[0]);  
        filter.DatumOd = string.IsNullOrEmpty(arr[1]) ?  
            new DateTime?() : DateTime.ParseExact(arr[1],  
                "dd.MM.yyyy", CultureInfo.InvariantCulture);  
        ...  
        return filter;  
    }  
}
```

Razred za informacije o filtru (3)

11

➡ Upit za dohvat svih dokumenata filtrira se po odabranim kriterijima

➡ Primjer:  Web \ Firma.Mvc \ ViewModels \ DokumentFilter.cs

```
public class DokumentFilter {  
    ...  
    public IQueryable<ViewDokumentInfo> Apply(  
        IQueryable<ViewDokumentInfo> query) {  
  
        if (IdPartnera.HasValue)  
            query = query.Where(d => d.IdPartnera == IdPartnera.Value);  
  
        if (DatumOd.HasValue)  
            query = query.Where(d => d.DatDokumenta >= DatumOd.Value);  
  
        ...  
    }  
}
```

Aktiviranje kalendara za odabir datuma

12

- Kalendar će se aktivirati klikom na kontrolu kojoj je pridijeljen stil s nazivom *datum*

- Primjer:  Web \ Firma.Mvc \ wwwroot \ js \ autocomplete.js

```
$(function () {  
    $(".datum").datepicker({  
        // dateFormat: "dd.mm.yy"  
    });  
});
```

- Paket *jQueryUI* uključen u projekt korištenjem alata *Bower* te se referencira u pogledima koji imaju potrebu za *jQueryUI-em*

- Za prikaz kalendara koristi se lokalizirana verzija

- hrvatski nazivi mjeseca i format odabranog datuma oblika dd.mm.yy

- Primjer:  Web \ Firma.Mvc \ Views \ Dokument \ Index.cshtml

```
@section scripts{  
    <script src="~/lib/jquery-ui/jquery-ui.js"></script>  
    <script src="~/lib/jquery-ui/ui/i18n/datepicker-hr.js"></script>  
    <script src="~/js/autocomplete.js"></script>
```

Dohvat svih partnera za dinamičke padajuće liste

13

➤ Izvedeno korištenjem pogleda korištenog za pregled svih partnera

➤ Primjer:  Firma.Mvc \ Controllers \ AutoComplete \ DokumentController.cs

```
[HttpGet]
public IEnumerable<IdLabel> Get(string term) {
    var query = ctx.vw_Partner
        .Select(p => new IdLabel {
            Id = p.IdPartnera,
            Label = p.Naziv + " (" + p.OIB + ")"
        })
        .Where(l => l.Label.Contains(term));

    var list = query.OrderBy(l => l.Label)
        .ThenBy(l => l.Id)
        .ToList();

    return list;
}
```

Digresija: Dohvat svih partnera upitom bez pogleda

14


➡ Bez pogleda, kôd za upit korištenjem EF-a bi bio puno složeniji

➡ Primjer:  Firma.Mvc \ Controllers \ AutoComplete \ DokumentController.cs

```
var queryOsobe = ctx.Osoba.Select(o => new IdLabel {  
    Id = o.IdOsobe,  
    Label = o.PrezimeOsobe + ", " +  
    o.ImeOsobe + " (" + o.IdOsobeNavigation.Oib + ") "  
})  
    .Where(l => l.Label.Contains(term));  
  
var queryPartneri = ctx.Tvrtka.Select(t => new IdLabel {  
    Id = t.IdTvrtke,  
    Label = t.NazivTvrtke + ", " +  
    " (" + t.IdTvrtkeNavigation.Oib + ") "  
})  
    .Where(l => l.Label.Contains(term));  
  
var list = queryOsobe.Union(queryPartneri)  
    .OrderBy(l => l.Label)  
    .ThenBy(l => l.Id)  
    .ToList();
```

Aktiviranje nadopunjavanja za partnere

15


- Izvedeno slično kao kod padajuće liste za mjesta prilikom dodavanja novog partnera
 - Vlastiti skripta aktivira nadopunjavanje za sve kontrole koje imaju definiran atribut `data-autocomplete`
 - Unutar vlastite `data` oznake informacija o relativnoj adresi izvora podataka
 - Razlika je u tome što se id odabranog partnera vidi na formi
 - Nije skriveno polje, ali se ne može mijenjati (atribut `readonly`)
 - Primijetiti da se veže za svojstvo `NazPartnera` iz razreda `DokumentFilter`
 - Primjer:  Web \ Firma.Mvc \ Views \ Dokument \ KriterijPretrage.cshtml

```
<input asp-for="IdPartnera"  
  readonly="readonly"  
  class="form-control"  
  data-autocomplete-result="partner" />
```

```
<input data-autocomplete="partner"  
  class="form-control"  
  asp-for="NazPartnera" />
```

Identifikator i naziv partnera u filtru

16

- Naziv odabranog partnera dio razreda *DokumentFilter*, ali nije dio teksta koji se prenosi u adresi unutar parametra *filter*
 - Potrebno ga je obnoviti upitom temeljem *IdPartnera*
- Primjer:  Web \ Firma.Mvc \ Controllers \ DokumentController.cs

```
public IActionResult Index(string filter, ...  
    ...  
    DokumentFilter df = new DokumentFilter();  
    if (!string.IsNullOrEmpty(filter)) {  
        df = DokumentFilter.FromString(filter);  
        if (!df.IsEmpty()) {  
            if (df.IdPartnera.HasValue) {  
                df.NazPartnera = ctx.vw_Partner  
                    .Where(p => p.IdPartnera == df.IdPartnera  
                    .Select(vp => vp.Naziv)  
                    .FirstOrDefault();  
            }  
            query = df.Apply(query);  
        }  
    }
```


Forma za ažuriranje dokumenta

17

- Zaglavlje za ažuriranje podataka koji pripadaju tablici Dokument (*master*)
 - Odabir partnera i prethodnog dokumenta izveden nadopunjavanjem
- Detalji sa stavkama dokumenta (*detail*)
 - Moguće promijeniti vrijednost stavke (količina i rabat), obrisati je ili dodati novu (poseban redak nakon svih stavki)
- Zajednička akcija za spremanje podataka (zaglavlje + stavke)

Dokument br: 5555

Vrsta dokumenta: R-1 Porez (u %): 22 Datum: 05.11.2016.

Broj: 11250 Partner: 509 CETINA (5914624)

Prethodni dokument: Iznos: 2.414,38 kn

Artikl	Količina	Rabat [0-1]	Jedinična cijena	Iznos	
AC Adapter ASIIN Mitac - 442687900004	3,00000	0,00	106,00 kn	318,00 kn	✕
ArtiklProba F	3,00000	0,00	35,00 kn	105,00 kn	✕
AC/DC adapter za 3Pod KINGSINGTON, 70W (33335EU)	2,00000	0,00	778,00 kn	1.556,00 kn	✕

Programsko inženjerstvo, Fakultet strojarstva, računarstva i elektrotehnike Sveučilišta u Mostaru, ak. god. 2017/18

Model za rad s dokumentima (1)

18

► Prezentacijski model s validacijskim atributima

► Primjer:  Web \ Firma.Mvc \ ViewModels \ DokumentViewModel.cs

```
public class DokumentViewModel {  
    public int IdDokumenta { get; set; }  
    [Display(Name = "Vrsta dokumenta")]  
    [Required(ErrorMessage = "Potrebno je ... dokumenta")]  
    public string VrDokumenta { get; set; }  
    ...  
    [DisplayFormat(DataFormatString = "{0:dd.MM.yyyy.}")]  
    [Display(Name = "Datum")]  
    [Required(ErrorMessage = "Potrebno je odabrati datum")]  
    public DateTime DatDokumenta { get; set; }  
    [Display(Name = "Porez (u %)")]  
    [Required(ErrorMessage = "Potrebno je postotak poreza")]  
    [Range(0, 100, ErrorMessage = "Porez mora biti 0-100")]  
    public int StopaPoreza { get; set; }  
    public decimal PostoPorez {  
        get { return StopaPoreza / 100m; }  
        set { StopaPoreza = (int) (100m * value); }  
    }  
}
```

Model za rad s dokumentima (2)

19

- Osim atributa koji će u konačnici završiti u tablici *Dokument*, model sadrži i kolekciju stavki

➤ Primjer:  Web \ Firma.Mvc \ ViewModels \ DokumentViewModel.cs

```
public class DokumentViewModel {  
    ...  
    public IEnumerable<StavkaViewModel> Stavke { get; set; }  
    public DokumentViewModel() {  
        this.Stavke = new List<StavkaViewModel>();  
    }  
}
```

- Stavke prikazane vlastitim prezentacijskim modelom

Model za rad sa stavkama

20

➡ Novi razred kao model za rad sa stavkama da se izbjegnu problemi s vezom prema tablici Dokument


➡ Primjer:  Web \ Firma.Mvc \ ViewModels \ StavkaViewModel.cs

```
public class StavkaViewModel
{
    public int IdStavke { get; set; }
    public int SifArtikla { get; set; }
    public string NazArtikla { get; set; }
    public decimal KolArtikla { get; set; }
    public decimal JedCijArtikla { get; set; }
    public decimal PostoRabat { get; set; }

    public decimal IznosArtikla {
        get {
            return KolArtikla * JedCijArtikla * (1 - PostoRabat);
        }
    }
}
```

Priprema dokumenta za ažuriranje (1)


21

- Za prikaz dokumenta ili početak ažuriranja dokumenta i njegovih stavki potrebno iz BP dohvatiti osnovne podatke o dokumentu i popuniti model iz prethodnih slajdova
 - Isti programski kod za Show i Edit
 - Primjer:  Web \ Firma.Mvc \ Controllers \ DokumentController.cs

```
[HttpGet]
public IActionResult Show(int id ...) {
    var dokument = ctx.Dokument.AsNoTracking()
        .Where(d => d.IdDokumenta == id)
        .Select(d => new DokumentViewModel {
            BrDokumenta = d.BrDokumenta,
            DatDokumenta = d.DatDokumenta,
            IdDokumenta = d.IdDokumenta,
            IdPartnera = d.IdPartnera,
            IdPrethDokumenta = d.IdPrethDokumenta,
            IznosDokumenta = d.IznosDokumenta,
            PostoPorez = d.PostoPorez,
            VrDokumenta = d.VrDokumenta
        })
        .FirstOrDefault();
}
```

Priprema dokumenta za ažuriranje (2)

22

- Odabir partnera i prethodnog dokumenta vrši se nadopunjavanjem, pa ne treba pripremati podatke za padajuće liste
- Ipak, ... potrebno dohvatiti nazive partnera i dokumenta kako bi se prikazali korisniku
 - Za partnera provjeriti tip i dohvatiti ime i prezime, odnosno naziv tvrtke
 - Slično za dokument
 - Potrebno koristiti isti format koji se koristi u upravljaču koji služi kao izvor za nadopune
- Primjer:  Web \ Firma.Mvc \ Controllers \ DokumentController.cs
 - Programski kod prevelik za slajd, ali relativno jednostavan...

Priprema dokumenta za ažuriranje (3)

23

➡ Nakon osnovnih podataka vrši se dohvat stavki dokumenta


➡ Primjer:  Web \ Firma.Mvc \ Controllers \ DokumentController.cs

```
[HttpGet]
public IActionResult Edit(int id ...) {
    ...
    var stavke = ctx.Stavka.Where(s => s.IdDokumenta ==
        dokument.IdDokumenta)
        .OrderBy(s => s.IdStavke)
        .Select(s => new StavkaViewModel {
            IdStavke = s.IdStavke,
            JedCijArtikla = s.JedCijArtikla,
            KolArtikla = s.KolArtikla,
            NazArtikla = s.SifArtiklaNavigation.NazArtikla,
            PostoRabat = s.PostoRabat,
            SifArtikla = s.SifArtikla
        })
        .ToList();

    dokument.Stavke = stavke;
    return View(dokument);
}
```

Pogled za ažuriranje dokumenta

24

- Dio sa zaglavljem nalik ostalim formama za ažuriranje pojedinačnog podatka
 - Input kontrole + kontrole za nadopunjavanje
 - Gumb za povratak na popis dokumenata, osvježavanje trenutnog dokumenta i spremanje promjena
- Na dnu forme se nalazi poziv parcijalnog pogleda koji će prikazati sve stavke dokumenta
 - Primjer:  Web \ Firma.Mvc \ Views \ Dokument \ Edit.cshtml

```
@model DokumentViewModel
...
<form id="form" method="post" asp-action="Edit" ... >
    <input type="hidden" asp-for="@Model.IdDokumenta"/>
    ...
    <input asp-for="StopaPoreza" class="form-control" />
    ...
    <button id="save" type="submit" title="Spremi">...</button>
    ...
    @Html.Partial("Stavke", Model.Stavke)
```


Povezivanje kolekcije podataka

25

- U slučaju da postoji više kontrola s istim atributom name unesene vrijednosti se na upravljaču prihvaćaju u istoimenom argumentu koji je polje nekog tipa vrijednosti.
- Što ako treba povezati više složenih podataka (npr. više stavki)?
 - Za atribut name se koristi oblik NazivKolekcije[indeks].NazivSvojstva
 - npr. name="Stavke[0].IdStavke", name="Stavke[1].IdStavke", name="Stavke[2].IdStavke" ...
 - Indeksi moraju biti bez prekida počevši od 0
- Što ako nije moguće osigurati neprekinuti niz indeksa?
 - Mogu se koristiti bilo koje oznake (čak ni ne moraju biti brojevi), ali dodatno treba stvoriti (skriveno) kontrole oblika Stavka.Index čija je vrijednost jednaka korištenoj oznaci

```
<input type="hidden" name="Stavke.Index" value="knjiga" />  
<input type="text" name="Stavke[knjiga].IdStavke" ... />
```

```
<input type="hidden" name="Stavke.Index" value="5" />  
<input type="text" name="Stavke[5].IdStavke" ... />
```

Parcijalni pogled za ažuriranje stavki

26

➡ *DokumentViewModel* sadrži svojstvo *Stavke*

➡ Povezivanje kolekcije stavki i njenih svojstva ostvareno varijantom sa *Stavke.Index*

➡ Kao indeks koristi se šifra artikla

➡ Primjer:  Web \ Firma.Mvc \ Views \ Dokument \ Stavke.cshtml

```
@model DokumentViewModel
...
@foreach (var stavka in Model) {
    <input type="hidden" name="Stavke.Index" value="@stavka.SifArtikla"/>

    <input type="hidden"
        name="Stavke[@stavka.SifArtikla].IdStavke"
        value="@stavka.IdStavke" />
    ...
    <input name="Stavke[@stavka.SifArtikla].KolArtikla"
        value="@stavka.KolArtikla"/>
    ...
}
```

Brisanje stavke

27

- ➡ Gumb za brisanje stavke (prepoznaje se po stilu *deleterow*) uklanja stavku iz strukture HTML dokumenta.


➡ Primjer:  Web \ Firma.Mvc \ wwwroot \ js \ dokument.js

```
$(document).on('click', '.deleterow', function () {  
    event.preventDefault();  
    var tr = $(this).parents("tr");  
    tr.remove();  
  
    //očisti staru poruku da je dokument uspješno snimljen  
    $("#tempmessage").siblings().remove();  
    $("#tempmessage").removeClass("alert-success");  
    $("#tempmessage").removeClass("alert-danger");  
    $("#tempmessage").html('');  
});
```

- ➡ Posljedično bit će izbrisana iz dokumenta nakon klika na gumb za snimanje

Izmjena dokumenta i njegovih stavki (1)

28


- Prilikom dohvata dokumenta potrebno je dohvatiti i sve njegove stavke
- Kao prvi korak vrši se kopiranje podataka dokumenta iz povezanog modela u objekt koji je dohvaćen iz BP
 - Primjer:  Web \ Firma.Mvc \ Controllers \ DokumentController.cs

```
[HttpPost]
[ValidateAntiForgeryToken]
public IActionResult Edit(DokumentViewModel model, ...
    var dokument = ctx.Dokument
        .Include(d => d.Stavka)
        .Where(d => d.IdDokumenta == model.IdDokumenta)
        .FirstOrDefault();

... Kopiraj podatke o dokumentu iz modela u objekt dokument
... dokument.PostoPorez = model.StopaPoreza / 100m;
    dokument.VrDokumenta = model.VrDokumenta;
```

Izmjena dokumenta i njegovih stavki (2)

29


- U listu cijelih brojeva spremaju se identifikatori povezanih stavki
 - Predstavlja stavke koje nisu uklonjene iz HTML-a prije snimanja
 - Iz konteksta se izbacuju sve stavke dokumenta koje se ne nalaze u toj listi
 - Primjer:  Web \ Firma.Mvc \ Controllers \ DokumentController.cs

```
[HttpPost]
[ValidateAntiForgeryToken]
public IActionResult Edit(DokumentViewModel model, ...
...
    List<int> idStavki = model.Stavke
                                .Where(s => s.IdStavke > 0)
                                .Select(s => s.IdStavke)
                                .ToList();

    //izbaci sve koje su nisu više u modelu
    ctx.RemoveRange(dokument.Stavka
                    .Where(s => !idStavki.Contains(s.IdStavke)));
```

Izmjena dokumenta i njegovih stavki (3)

30

- Stavke koje se nalaze u povezanom modelu su ili nove stavke ili neke od postojećih
 - Postojeće treba dohvatiti iz objekta dohvaćenog iz baze podataka
 - Nove treba stvoriti i dodati u dokument
 - U oba slučaja u novi objekt kopirati svojstva iz povezanog modela
- Primjer:  Web \ Firma.Mvc \ Controllers \ DokumentController.cs

```
foreach (var stavka in model.Stavke) {  
    Stavka novaStavka;  
    if (stavka.IdStavke > 0) {  
        novaStavka = dokument.Stavka  
            .First(s => s.IdStavke == stavka.IdStavke);  
    }  
    else {  
        novaStavka = new Stavka();  
        dokument.Stavka.Add(novaStavka);  
    }  
    novaStavka.SifArtikla = stavka.SifArtikla;  
    novaStavka.KolArtikla = stavka.KolArtikla;  
    kopiranje ostalih vrijednosti
```

Izmjena dokumenta i njegovih stavki (4)

31

➡ Potrebno izračunati iznos dokumenta i spremiti promjene




➡ Primjer:  Web \ Firma.Mvc \ Controllers \ DokumentController.cs

```
[HttpPost]
[ValidateAntiForgeryToken]
public IActionResult Edit(DokumentViewModel model, ...
    ...
    dokument.IznosDokumenta =
        (1 + dokument.PostoPorez) * dokument.Stavka
                                   .Sum(s =>
                                       s.KolArtikla
                                       * (1 - s.PostoRabat)
                                       * s.JedCijArtikla);

    ctx.SaveChanges();
```



Odabir artikla za novu stavku

32

- Poseban redak na dnu postojećih stavki u kojem se nadopunjavanjem bira artikl
 - Povratni podaci sa servera sadrže šifru artikla, naziv artikla i cijenu artikla
 - Cijena artikla se kopira u odgovarajuće polje
- Pogledati programski kôd u sljedećim datotekama:
 -  Web \ Firma.Mvc \ Controllers \ AutoComplete \ Artikl.cs
 -  Web \ Firma.Mvc \ Controllers \ AutoComplete \ ArtiklController.cs
 -  Web \ Firma.Mvc \ wwwroot \ js \ autocomplete.js
 - Programski kod za selektor \$("[data-autocomplete-artikl]")


Dodavanje nove stavke

33

- Skrivena tablica s identifikatorom *template* koja služi za dodavanje novog retka na osnovu vrijednosti iz posebnog retka na dnu stavki
 - Provjera ispravnosti vrijednosti, provjera postoji li duplikat artikla i slično
 - Dodavanje novog retka
 - Ponašanje tipke ENTER
- Redak s predloškom se uklanja neposredno prije snimanja forme kako ne bi sudjelovao u povezivanju
- Pogledati programski kôd u sljedećim datotekama:
 -  Web \ Firma.Mvc \ Views \ Dokument \ Stavke.cshtml
 -  Web \ Firma.Mvc \ wwwroot \ js \ dokumenti.js

Dodatna funkcionalnost (1)

34

- Na stranici za ažuriranje dokumenta poveznica za povratak na listu svih dokumenata
 - Pamti se prethodna stranica i način sorta
- Dodatno, poveznica na prethodni i sljedeći dokument
 - Svaki dokument ima svoju poziciju unutar baze podataka u ovisnosti o trenutnom sortu i filtru
 - Inicijalno pridijeljeno prilikom dohvata dokumenata
 - Primjer:  Web \ Firma.Mvc \ Controllers \ DokumentController.cs

```
public IActionResult Index(string filter, int page = 1...  
  
    ... Dohvati dokumente u ovisnosti o filtru i sortu ...  
  
    for(int i=0; i<dokumenti.Count; i++)  
        dokumenti[i].Position = (page - 1) * pagesize + i;
```

Traženje sljedbenika i prethodnika

35

► Kombinacijom Skip i Take

► Primjer:  Web \ Firma.Mvc \ Controllers \ DokumentController.cs


```
private void SetPreviousAndNext(int position, string filter,
                                int sort, bool ascending) {

    var query = ctx.ViewDokumentInfo.AsNoTracking()
        .FromSql(Constants.SqlViewDokumenti);
    ... proširi upit sa filtrom i sortiranjem ...
    if (position > 0)
        ViewBag.Previous = query.Skip(position - 1)
                                .Select(d => d.IdDokumenta)
                                .First();

    if (position < query.Count() - 1) {
        ViewBag.Next = query.Skip(position + 1)
                            .Select(d => d.IdDokumenta)
                            .First();
    }
}
```

Kreiranje novog dokumenta

36

- Kao početni model za unos novog dokumenta stvara se novi objekt kojem se postavlja trenutni datum i sljedeći broj dokumenta
 - Samo kao primjer početne inicijalizacije, jer u slučaju istovremenog dodavanja novih dokumenata može doći do ponavljanja istog broja
- Primjer:  Web \ Firma.Mvc \ Controllers \ DokumentController.cs

```
[HttpGet]
public IActionResult Create() {
    int maxbr = ctx.Dokument.Max(d => d.BrDokumenta) + 1;
    var dokument = new DokumentViewModel {
        DatDokumenta = DateTime.Now,
        BrDokumenta = maxbr
    };
    return View(dokument);
}
```

- Ostatak izveden slično kao kod ažuriranja (razlika u tome što su sve stavke nove).

Izrada izvješća

Alat za izradu izvješća

38

- Za izradu izvješća u PDF formatu koristit će se alat PdfReport.Core

- <https://github.com/VahidN/PdfReport.Core>



PdfRpt.Core by Vahid Nasiri

📄 v1.1.4

PdfReport.Core is a code first reporting engine, which is built on top of the iTextSharp.LGPLv2.Core and EPPlus.Core libraries.

- Primjeri sadrže 3 vrsta izvješća:
 - Jednostavni tablični izvještaj
 - Popis svih država
 - Tablični izvještaj sa slikama i sumom vrijednosti nekog stupca
 - Prikaz 10 najskupljih artikala sa slikom
 - Master-detail primjer koji demonstrira grupiranje elemenata
 - N najboljih kupnji (tj. dokumenata s najvećom vrijednosti zajedno s popisom stavki)

Inicijalne postavke za izvješća (1)

39

- Razred za rad s izvješćima je *PdfReport* koji sadrži postupke koji vraćaju referencu na vlastiti objekt, pa se postupci mogu vezati
 - Inicijalno se postavljaju metapodaci (autori, naslov, ...) i orijentacija
 - Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs

```
private PdfReport CreateReport(string naslov) {  
    var pdf = new PdfReport();  
    pdf.DocumentPreferences(doc =>  
    {  
        doc.Orientation(PageOrientation.Portrait);  
        doc.PageSize(PdfPageSize.A4);  
        doc.DocumentMetadata(new DocumentMetadata {  
            Author = "FER-ZPR", Application = "Firma.MVC Core",  
            Title = naslov  
        });  
        doc.Compression(new CompressionSettings {  
            EnableCompression = true,  
            EnableFullCompression = true  
        });  
    });  
}
```

Inicijalne postavke za izvješća (2)

40

- ➡ Postavlja se vrsta predloška, automatska širina stupaca, može se ograničiti broj redaka po stranici, broj grupa po stranici i slično.

➡ Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs

```
private PdfReport CreateReport(string naslov) {  
    var pdf = new PdfReport();  
    ...  
    .MainTableTemplate(template => {  
        template.BasicTemplate(BasicTemplate.ProfessionalTemplate);  
    })  
    .MainTablePreferences(table => {  
        table.ColumnsWidthType(TableColumnWidthType.Relative);  
        //table.NumberOfDataRowsPerPage(20);  
        table.GroupsPreferences(new GroupsPreferences {  
            GroupType = GroupType.HideGroupingColumns,  
            RepeatHeaderRowPerGroup = true,  
            ShowOneGroupPerPage = true  
        });  
        table.SpacingAfter(4f);  
    });  
}
```


Zaglavlje i podnožje jednostavnih izvješća

41

➤ Jednostavna izvješća će u zaglavlju imati naslov izvješća, a u podnožju trenutni datum

➤ Potrebno postaviti smjer teksta na LeftToRight

➤ Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs

```
public async Task<IActionResult> Drzave() {  
    ...  
    PdfReport report = CreateReport(naslov);  
    report.PagesFooter/footer => {  
        footer.DefaultFooter(DateTime.Now.ToString("dd.MM.yyyy."));  
    })  
    .PagesHeader/header => {  
        header.CacheHeader(cache: true);  
        header.DefaultHeader(defaultHeader => {  
            defaultHeader.RunDirection(  
                PdfRunDirection.LeftToRight);  
            defaultHeader.Message(naslov);  
        });  
    });  
};
```

Izvor podataka za izvješće

42

➤ U svim primjerima kao izvor podataka služit će lista čiji su elementi objekti nekog prezentacijskog modela ili anonimnog razreda

➤ Podaci se u izvještaj dodaju redom kojim se nalaze u listi

➤ Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs

```
public async Task<IActionResult> Drzave() {  
    var drzave = await ctx.Drzava  
        .AsNoTracking()  
        .OrderBy(d => d.NazDrzave)  
        .ToListAsync();  
}
```

```
public async Task<IActionResult> Artikli() {  
    var artikli = await ctx.Artikl  
        .AsNoTracking()  
        .Where(a => a.SlikaArtikla != null)  
        .OrderByDescending(a => a.CijArtikla)  
        .Select(a => new {  
            a.SifArtikla, a.NazArtikla,  
            a.CijArtikla, a.SlikaArtikla  
        })  
        .Take(10).ToListAsync();  
}
```

Postavljanje izvora podataka

43

- Izvor se postavlja pozivom postupka *MainTableDataSource* i pisanjem odgovarajućeg delegata tipa *Action<MainTableDataSourceBuilder>*
 - Nad podatkom tipa *MainTableDataSourceBuilder* poziva se postupak *StronglyTypedList* te se kao argument predaje pripremljena lista
 - Alternative: postupci *SqlDataReader*, *CustomDataSource*, ...
- Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs

```
public async Task<IActionResult> Drzave() {  
    var drzave = await ctx.Drzava  
        .AsNoTracking()  
        .OrderBy(d => d.NazDrzave)  
        .ToListAsync();  
  
    report.MainTableDataSource(dataSource =>  
        dataSource.StronglyTypedList(drzave));  
}
```

Definiranje stupaca

44

➡ Željeni stupci se navode u postupku *MainTableColumns*

➡ Stupcima se postavlja poredak (ako se izostavi koristi se redoslijed navođenja) relativna širina, vidljivost, zaglavlje te naziv svojstva

➡ Moguće imati redak s rednim brojem

➡ Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs (Drzave)

```
report.MainTableColumns(columns => {  
    columns.AddColumn(column => {  
        column.IsRowNumber(true);  
        column.CellsHorizontalAlignment(HorizontalAlignment.Right);  
        column.IsVisible(true); column.Order(0); column.Width(1);  
        column.HeaderCell("#",  
            horizontalAlignment: HorizontalAlignment.Right);  
    });  
    columns.AddColumn(column => {  
        column.PropertyName(nameof(Drzava.OznDrzave));  
        column.Order(1); column.Width(2);  
        column.HeaderCell("Oznaka države");  
    });  
});
```

Preuzimanje izvješća

45

➤ Binarni sadržaj izrađenog izvješća vraća se korisniku korištenjem postupka File (naslijeđen iz razreda *Controller*)

1. Preglednik nudi odabir mape za snimanje datoteke
2. Sugerirati pregledniku da isporučeni sadržaj prikaže unutar preglednika

➤ content-disposition:inline

➤ Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs

```
public async Task<IActionResult> Drzave() {  
    ...  
    byte[] pdf = report.GenerateAsByteArray();  
    if (pdf != null) {  
        Response.Headers.Add("content-disposition",  
            "inline; filename=drzave.pdf");  
        return File(pdf, "application/pdf");  
        //return File(pdf, "application/pdf", "drzave.pdf");  
        // samo gornji redak otvara Save as dialog  
    }  
    else return NotFound();  
}
```

Stupci s posebnim predloškom

46


➡ Za svaki artikl prikazana slika u izvješću

➡ Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs

```
public async Task<IActionResult> Artikli()  
...  
report.MainTableColumns(columns => {  
    columns.AddColumn(column => {  
        column.PropertyName(nameof(Artikl.SlikaArtikla));  
        ...  
        column.ColumnItemsTemplate(t =>  
            t.ByteArrayImage(string.Empty,  
                fitImages: true));  
    });  
...  
}
```

Stupci sa sumom vrijednosti u podnožju

47

- Postavlja se konačna suma svih artikala i tekst ispred sume
 - moguće postaviti i međusume za svaku stranicu *PageSummarySettings*
 - Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs

```
public async Task<IActionResult> Artikli()  
...  
report.MainTableSummarySettings(summarySettings => {  
    summarySettings.OverallSummarySettings("Ukupno");  
});  
report.MainTableColumns(columns => {  
    columns.AddColumn(column => {  
        column.PropertyName<Artikl>(x => x.CijArtikla);  
        column.ColumnItemsTemplate(template => {  
            template.TextBlock();  
            template.DisplayFormatFormula(obj =>  
                string.Format("{0:C2}", obj));  
        });  
        column.AggregateFunction(aggregateFunction => {  
            aggregateFunction.NumericAggregateFunction(  
                AggregateFunction.Sum);  
            aggregateFunction.DisplayFormatFormula(...
```


Primjer izvještaja oblika zaglavlje stavke

- Za svaki dokument ispisani osnovni podaci nakon čega slijede sve stavke
 - Nakon ispisa stavki ispisana suma bez PDV-a
- Stupac *Ukupno* je izračunati stupac

10 najvećih kupnji					
Id dokumenta: 2237		Datum dokumenta: 11.02.2014			
Partner: KBC		Iznos dokumenta: 2.178.811,73 kn			
#	Naziv artikla	Količina	Jedinična cijena	Rabat	Ukupno
1	CD/MP3 radio kazetofon, PHILIP MORRIS PH-AZ1038, 1 x kazeta	1,00	289,00 kn	5,00%	274,55 kn
2	DVD/CD/MP3/DivX player, linijski, BIONEER DV-300-K + 7 crtiča, crni	10,00	549,00 kn	9,00%	4.995,90 kn
3	GPS uređaj GURMAN GPS 72	1,00	1.291,00 kn	7,00%	1.200,63 kn
4	Knjiga "Access 2003 za Windows"	2,00	157,00 kn	9,00%	285,74 kn
5	Knjiga "Skok u Linux"	2,00	199,00 kn	2,00%	390,04 kn
6	MBO EVGA, s. 775, 650i Ultra, NVIDIA nForce 650i Ultra, BUS 1333 MHz, serial ATA II, RAID, 7.1 zvuk, 1Gbps, SSSR 2, ATX 2	1,00	684,00 kn	1,00%	677,16 kn
7	Mikrovalna pećnica LGG MS-1924V, 19 litara	1,00	419,00 kn	1,00%	414,81 kn
8	Mobilni uređaj MOTOCOLA V3, GPRS, EMS, Bluetooth, ekran u boji	1,00	999,00 kn	2,00%	979,02 kn
9	Mobilni uređaj NUKIA N73, GPRS, Bluetooth, MMS, FM Radio, kamera, ekran u boji, crveni	2,00	3.499,00 kn	1,00%	6.928,02 kn
10	MP3/WMA player, DESTRUCTIVE Zen Vision M 60 GB, crni	3,00	2.299,00 kn	6,00%	6.483,18 kn
11	MP3/WMA/FM player, CS, MC309F, 1 GB	5,00	403,00 kn	1,00%	1.994,85 kn
12	Notebook ACER Aspire 5610 NWLMi, Kintel Solo Core T1350(1.83ghz), TFT 15.4" WXGA, 1GB (512MB + 512MB POKLON), DVD±RW, HDD 60 GB, bežična mreža, Linux (LX.AU60C.004)	4,00	4.799,00 kn	7,00%	17.852,28 kn
13	Notebook PH Compaq 6715b (GB833EA), KPD Sempron 3800+ (2.2GHz), 15.4" WXGA (1280x800), RAM 1GB SSSR2, HDD 120GB SATA, DVD+RW DL, ATI Radeon X1250, modem, LAN, WLAN, BT, Windows Vista Basic	4,00	5.192,00 kn	8,00%	19.106,56 kn
14	PC računalo HGspotVR PREMIUM II + Windows VISTA Home Premium, KPD Athlon 64 X2 4400+, 1 GB SSSR2, HDD 250 GB SATA-II, GeForce 8500 GT 256 MB, DVD±RW, TV Tuner+zvučnici 2.1 + 19" LCD BENQ monitor + poklon Chat Pack i BITDEFENDER Internet Security 2008	2,00	5.199,00 kn	10,00%	9.358,20 kn
15	RAM, 1 GB, SSSR 2, PC-5300, 667 MHz,	10,00	178,00 kn	2,00%	1.744,40 kn
16	Torba za notebook Marasst MFL30	4,00	141,00 kn	8,00%	518,88 kn
17	Torbica za Strukkle triPOD trio nano 3-1, kožna, siva	3,00	114,00 kn	4,00%	328,32 kn
18	TV LCD 82 cm, SHARP LC32WD1E, 16:9, stereo, HD Ready, 2 x HDMI, DVB -T	1,00	7.399,00 kn	9,00%	6.733,09 kn
19	TV LCD 94 cm, PHILIP MORRIS 37PFL9732, FULL HD, DVB-T	100,00	16.999,00 kn	1,00%	1.682.901,00 kn
Ukupno					763.166,63 kn

Oblik podataka za izvješća s grupiranjem

49

➤ Potrebno koristiti denormalizirane podatke

➤ Primjer:  Web \ Firma.Mvc \ ViewModels \ StavkaDenorm.cs

```
public class StavkaDenorm {  
    public string OIB { get; set; }  
    public string NazPartnera { get; set; }  
    public int IdDokumenta { get; set; }  
    public DateTime DatDokumenta { get; set; }  
    public decimal IznosDokumenta { get; set; }  
    public int IdStavke { get; set; }  
    public int SifArtikla { get; set; }  
    public decimal KolArtikla { get; set; }  
    public decimal JedCijArtikla { get; set; }  
    public decimal PostoRabat { get; set; }  
    public string NazArtikla { get; set; }  
    [NotMapped] public string UrlDokumenta { get; set; }  
}
```

➤ Svojstvo *UrlDokumenta* predstavljat će adresu dokumenta

➤ Formirat će se nakon dohvata na osnovu usmjeravanja i *IdDokumenta*

➤ Ne postoji u bazi podataka pa se mora označiti s *NotMapped*

Funkcija za dohvat n najboljih kupnji

50


➤ Ispis n dokumenata s najvećim iznosom

- Složen upit te je izveden korištenjem funkcije na BP
- U kontekst potrebno dodati DbSet<StavkaDenorm> i navesti podatak koji jednoznačno određuje podatak

```
CREATE FUNCTION [dbo].[fn_NajveceKupnje] (@N int)
RETURNS TABLE AS
RETURN(
    SELECT P.OIB, P.Naziv AS NazPartnera,
           D.IdDokumenta, D.DatDokumenta, D.IznosDokumenta,
           S.IdStavke, S.SifArtikla, S.KolArtikla,
           S.JedCijArtikla, S.PostoRabat, A.NazArtikla
    FROM
        (SELECT TOP (@N) * FROM Dokument ORDER BY IznosDokumenta DESC) D
    INNER JOIN vw_Partner P ON P.IdPartnera = D.IdPartnera
    LEFT OUTER JOIN Stavka S ON S.IdDokumenta = D.IdDokumenta
    LEFT OUTER JOIN Artikl A ON A.SifArtikla = S.SifArtikla
)
```

Poziv funkcije s parametrima


51

- Za dohvat iz DbSet-a koji se odnosi na funkciju, potrebno napisati SQL upit (u ovom slučaju parametrizirani)
- Podaci moraju biti poredani po stupcima po kojima se želi izvršiti grupiranje
 - Poredak nakon toga po želji
- Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs

```
public async Task<IActionResult> Dokumenti() {  
    int n = 10;  
    var param = new SqlParameter("N", n);  
    string naslov = $"{n} najvećih kupnji";  
    var stavke = await ctx.StavkaDenorm  
        .AsNoTracking()  
        .FromSql(  
            "SELECT * FROM fn_NajveceKupnje(@N)", param)  
        .OrderBy(s => s.IdDokumenta)  
        .ThenBy(s => s.NazArtikla)  
        .ToListAsync();  
}
```

Postavljanje adrese za ažuriranje dokumenta


52

- Unutar PDF-a bit će dodana poveznica za ažuriranje dokumenta
- Potrebno pripremiti svojstvo s adresom
 - Akcija Edit na upravljaču Dokument s parametrom id
 - Sprema se u svojstvo UrlDokumenta za svaki element dohvaćene liste
- Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs

```
public async Task<IActionResult> Dokumenti() {  
    ...  
    var stavke = await ctx.StavkaDenorm  
        ...  
        .ToListAsync();  
  
    stavke.ForEach(s => s.UrlDokumenta = Url.Action("Edit",  
                                                    "Dokument",  
                                                    new { id = s.IdDokumenta }  
    ));  
}
```

Definiranje stupaca po kojima se vrši grupiranje

53

- Prilikom definiranja stupaca u prikazu, prvo se navode stupci po kojima se vrši grupiranje
 - Neće biti prikazani u tablici, već iznad nje
- Potrebno napisati kriterij pripadaju li dvije vrijednosti tog „stupca” istoj grupi
- Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs

```
public async Task<IActionResult> Artikli()  
...  
report.MainTableColumns(columns => {  
    columns.AddColumn(column => {  
        column.PropertyName("IdDokumenta");  
        column.Group(  
            (val1, val2) => {  
                return (int)val1 == (int)val2;  
            });  
    });  
});
```

Stupac s automatskim izračunom vrijednosti

54


➡ Vrijednost stupca s ukupnom cijenom neke stavke računa se na osnovu ostalih vrijednosti iz trenutnog retka

➡ Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs

```
public async Task<IActionResult> Artikli()  
...  
report.MainTableColumns(columns => {  
    columns.AddColumn(column => {  
        column.CalculatedField(  
            list => {  
                if (list == null) return string.Empty;  
                decimal kolArtikla = (decimal)list.GetValueOf("KolArtikla");  
                decimal postoRabat = (decimal)list.GetValueOf("PostoRabat");  
                decimal jedCijArtikla = (decimal)list.GetValueOf("JedCijArtikla");  
                var iznos = jedCijArtikla * kolArtikla * (1 - postoRabat);  
                return iznos;  
            });  
    });  
});
```

Zaglavlje grupe

55

- Kao predložak za zaglavlje pojedine grupe potrebno napisati vlastiti predložak kojim se definira kako izgleda zaglavlje izvještaja i zaglavlje pojedine grupe
- Predložak se postavlja postupkom *CustomHeader* u *PagesHeader*
 - Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs

```
public async Task<IActionResult> Artikli()  
...  
report.PagesHeader(header => {  
    header.CacheHeader(cache: true);  
    header.CustomHeader(new MasterDetailsHeaders(naslov)  
    {  
        PdfRptFont = header.PdfFont  
    });  
});
```

Predložak za zaglavlje izvještaja

56

➡ Zaglavlje izvještaja sastoji se od jednog retka s naslovom

➡ Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs

```
public class MasterDetailsHeaders : IPageHeader {
    private string naslov;
    public MasterDetailsHeaders(string naslov) {
        this.naslov = naslov;
    }
    public IPdfFont PdfRptFont { set; get; }

    public PdfGrid RenderingReportHeader(...
    {
        var table = new PdfGrid(numColumns: 1) { WidthPercentage = 100 };
        table.AddSimpleRow(
            (cellData, cellProperties) => {
                cellData.Value = naslov;
                ...
            });
        return table.AddBorderToTable();
    }
}
```


Predložak za zaglavlje grupe

57

➤ Vrijednosti grupe mogu se dobiti iz ulaznog argumenta tipa *ICollection<CellData>*

➤ Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs

```
public class MasterDetailsHeaders : IPager {
    public PdfGrid RenderingGroupHeader(...
        ICollection<CellData> newGroupInfo ...){

        var idDokumenta = newGroupInfo
            .GetStringValueOf("IdDokumenta");

        ...
        var iznosDokumenta = (decimal) newGroupInfo
            .GetValueOf("IznosDokumenta");
    }
}
```

Predložak za zaglavlje grupe

58

➤ Zaglavlje grupe definirano kao tablica s 4 stupca s podacima grupe

➤ Svaka ćelija može sadržavati fiksni tekst ili biti vezana za neku vrijednost

➤ Primjer:  Web \ Firma.Mvc \ Controllers \ ReportController.cs

```
public class MasterDetailsHeaders : IPageHeader {
    public PdfGrid RenderingGroupHeader( ...
        ...
        var table = new PdfGrid(relativeWidths: new[]
            { 2f, 5f, 2f, 3f }) { WidthPercentage = 100 };

        table.AddSimpleRow(
            ...
            (cellData, cellProperties) => {
                cellData.Value = datDokumenta;
                cellProperties.PdfFont = PdfRptFont;
                cellProperties.HorizontalAlignment =
                    HorizontalAlignment.Left;

                cellProperties.DisplayFormatFormula =
                obj => ((DateTime)obj).ToString("dd.MM.yyyy");
            });
    }
```