

Web-aplikacije

ASP.NET Core MVC

2019/20.13

Primjer zaglavlje-stavke

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

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

Filtriranje podataka

6

- 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

```
<partial name="KriterijPretrage"
        model="Model.Filter" />
```

IdPartnera: 0 Zadovoljni korisnik

Iznos: 300,00 - 500,00

Datum: 01.01.2015. -

1

Iznos dokumenta

326,25 kn		
382,50 kn		
425,00 kn		

Svibanj 2017

Po	Ut	Sr	Če	Pe	Su	Ne
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

Parametar ili sjednica?

7

- 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)

8

➡ 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)

9

➡ 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)

10

➡ 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

11

- 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 *libman* 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

12

➡ 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

13


➡ 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

14


- 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-placeholder="partner" />
```

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

Identifikator i naziv partnera u filtru

15

- 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

16

- 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. 2019/20

Model za rad s dokumentima (1)

17

➤ 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)

18

- 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

19

➡ 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)


20

- 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)

21

- 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)

22

➡ 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

23

- 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>
    ...
    <partial name="Stavke" model="Model.Stavke" />
```

Povezivanje kolekcije podataka

24

- 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 (skrivenne) 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

25

➡ *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

26

- 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)

27


- 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)

28


- 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)

29

- 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)

30

➡ 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

31

- 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


Dodavanje nove stavke

32

- 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)

33

- 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

34

➡ 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

35

- 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).