SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Seminar iz kolegija

OBJEKTNO OBLIKOVANJE

**AVESOBZ**

**Aplikacija za vođenje evidencije studenata Osječko-baranjske županije**

Case study

Mario Petričević

Zagreb, siječanj, 2021.

Sadržaj

[1. Uvod 3](#_Toc63035596)

[1.1. Design story 3](#_Toc63035597)

[2. Zahtjevi 5](#_Toc63035598)

[2.1. Razrada pojedinih zahtjeva 5](#_Toc63035599)

[2.1.1. USE CASE 1 – Podnošenje zahtjeva za izradu pametne kartice 5](#_Toc63035600)

[2.1.2. USE CASE 2 – Kreiranje profila 6](#_Toc63035601)

[2.1.3. USE CASE 7 – Izdavanje karte 6](#_Toc63035602)

[3. Opis (objektnog) modela domene 8](#_Toc63035603)

[4. Arhitektura 10](#_Toc63035604)

[5. O/R mapiranje za laboratorij profila 14](#_Toc63035605)

[5.1. Usporedba NHibernatea i Entity Frameworka 14](#_Toc63035606)

[5.2. Ostvarivanje mapiranja 15](#_Toc63035607)

# Uvod

Cilj seminara je prezentirati na praktičnom primjeru usvojeno znanje koje smo obrađivali tijekom semestra razvojem case study-a na izabranu temu. Potrebno je primijeniti tehnike objektnog oblikovanja te razvoja softvera. Tema ovog case study-a je Aplikacija za vođenje evidencije studenata Osječko-baranjske županije. Aplikacija će omogućiti Hrvatskim željeznicama jednostavnije vođenje evidencije studenata u svrhu izdavanja besplatnih karti za putovanje od mjesta studiranja do mjesta prebivališta i obratno.

## Design story

Od 2021. godine Hrvatske željeznice u suradnji s Osječko baranjskom županijom sufinanciraju prijevoz studentima Osječko baranjske županije. Sufinanciranje se odnosi na prijevoz vlakom iz mjesta studiranja u mjesto boravišta i obratno.

Do sada, student su mogli kupiti karte u iznosu od 50% njihove cijene. Od ove godine cijene karata za student potpuno su besplatne što uvelike olakšava život studentima. Kako bi studenti ostvarili popust moraju izraditi pametnu karticu te u uredu Osječko baranjske županije izvaditi potvrdu da su studenti na području županije. Kako bi student kupio kartu besplatno, svaki put pri kupnji karte na blagajni kolodvora mora ostaviti navedenu potvrdu te priložiti pametnu karticu. Vidimo jasan problem gdje se studente opterećuje sa stalnim printanjem potvrda te također gomilanje papirologije na blagajnama kolodvora. Zbog toga studenti su se odlučili odužiti HŽ-u te im napraviti aplikaciju koja će pomoći pri vođenju evidencije studenata te će se značajno olakšati rad djelatnicima HŽ-a.

Student najprije dolazi na blagajnu kolodvora te podnosi zahtjev za izradu pametne kartice u svrhu besplatnog prijevoza. Djelatnik unosi podatke o studentu te u aplikaciji kreira profil za studenta unoseći njegove osobne podatke te otvara zahtjev za izradu pametne kartice koja će se izraditi u roku od par dana. Student također ostavlja svoj kontakt broj te mu nakon što je kartica gotova stiže obavijest.

Student mora dostaviti potvrdu da je student na području Osječko baranjske županije samo jednom, te prilikom dostavljanja potvrde djelatnik unosi u aplikaciju da je potvrda zaprimljena te datum do kojega potvrda traje. Potvrda traje godinu dana te nakon isteka datuma student ju može obnoviti, ako ju ne obnovi gubi pravo na popust od 100%. Osim vođenja evidencije aplikacija također pruža mogućnost izdavanja karata. Prilikom kupnje karata student mora imati pametnu karticu koju pokazuje djelatniku. Ukoliko ju nema, neće moći ostvariti popust. Djelatnik tada skenira karticu i provjerava podatke o studentu. Ako student ima karticu, a nije dostavio potvrdu, ostvarit će popust od 50% te će morati platiti kartu. Ukoliko student ima evidentiranu potvrdu te je ona valjana, ostvaruje popust od 100% te kartu plaća besplatno. Pri izdavanju karte, djelatnik unosi mjesto polazišta i odredišta te aplikacija automatski određuje cijenu karte ovisno o tome koji kriteriji su zadovoljeni.

Hrvatske željeznice isporučuju izvještaj Osječko baranjskoj županiji o kupljenim studentskim kartama s obračunatim popustima te broj izrađenih studentskih profila. Izvještaj se isporučuje u svrhu izrade godišnjeg proračuna te konačnog računa koji HŽ ispostavlja županiji.

# Zahtjevi

Sustav ima nekolicinu zahtjeva te funkcionalnosti za ispravan rad. Neke funkcionalnosti opisane su i razrađene preko use-case-ova dok su neke implementirane u krajnjoj aplikaciji.

Zahtjevi su:

1. Podnošenje zahtjeva za izradu pametne kartice
2. Kreiranje profila
3. Brisanje profila
4. Ažuriranje profila
5. Dostava potvrde
6. Evidentiranje potvrde
7. Izdavanje karte
8. Pregled izvještaja za određeno vremensko razdoblje

## Razrada pojedinih zahtjeva

### USE CASE 1 – Podnošenje zahtjeva za izradu pametne kartice

Primary actor: Student

Stakeholders: Student – želi ostvariti pravo na popust pri putovanju vlakom u mjesto studiranja

Preconditions: Student mora imati status studenta( predočenje iksice)

Postconditions: Zahtjev za izradu pametne kartice uspješno podnesen

Glavni uspješni scenarij:

1. Student dolazi na blagajnu podnijeti zahtjev za izradu pametne kartice
2. Student predočuje studentsku karticu
3. Djelatnik otvara zahtjev za izradu pametne kartice
4. Student ostavlja kontakt broj na kojega dolazi obavijest kada je kartica gotova

Alternativni scenarij:

2.a) Student trenutno nema studentski karticu

1.Student će nastaviti podnošenje zahtjeva za izradom kartice nakon što pronađe iksicu.

### USE CASE 2 – Kreiranje profila

Primary actor: Djelatnik

Stakeholders: Djelatnik – unosi podatke o studentu

Student – daje svoje podatke

Preconditions: Djelatnik ima pristup aplikaciji te ima prava izraditi novi profil studentu

Postconditions: Studentu je kreiran profil te je unesen u aplikaciju. Omogućeno mu je kupovanje karata s popustom.

Glavni uspješni scenarij:

1. Student daje svoje osobne informacije djelatniku
2. Djelatnik unosi ime i prezime studenta
3. Djelatnik unosi kućnu adresu studenta
4. Djelatnik unosi kontakt broj studenta
5. Djelatnik potvrđuje podatke klikom na kreiraj profil

Alternativni scenarij:

2.a) Student je već unesen u sustav. Obustavlja se kreiranje novog profila jer profil već postoji.

5.a) Aplikacija iz nekog razloga ne može kreirati novi profil. Djelatnik se obraća administratoru koji će osvježiti aplikaciju.

### USE CASE 7 – Izdavanje karte

Primary actor: Djelatnik

Stakeholders: Djelatnik – izdaje kartu

Student – kupuje kartu do mjesta studiranja ili do mjesta prebivališta

Preconditions: Student posjeduje pametnu karticu uz koju će moći kupiti kartu s popustom. Također student mora imat izrađen profil

Postconditions: Djelatnik na kolodvoru izdaje kartu studentu te mu obračunava popust

Glavni uspješni scenarij:

1. Student želi kupiti kartu do mjesta studiranja ili do mjesta prebivališta
2. Student predočuje pametnu karticu
3. Djelatnik skenira karticu te provjerava aktivnost profila
4. Djelatnik unosi mjesto do kojega student putuje
5. Djelatnik provjerava je li student dostavio potvrdu
6. Djelatnik odobrava popust od 100% te izdaje kartu studentu besplatno

Alternativni scenarij:

2.a) Studentov profil nije aktivan te ga student može ponovno aktivirati

5.a) Student nije dostavio potvrdu da je student Osječko baranjske županije te na osnovu toga ne može dobiti besplatnu kartu. Student plaća kartu

6.a) Student nije dostavio potvrdu te mu djelatnik ne može izdati besplatnu kartu. Djelatnik studentu izdaje kartu u popustu od 50%

# Opis (objektnog) modela domene

****

**Slika 1. Dijagram razreda modela domene**

Slika 1. prikazuje dijagram razreda modela domene. Profile predstavlja profil svakog studenta. Definiran je entityrazredom Profile. Razred Profile sadrži atribute Ime, Kontakt, Prezime, Potvrda, Mjesto te konstruktor. ProfileRepository je razred koji sadrži sve profile studenata te metode za dohvaćanje, dodavanje, brisanje profila te njihovih svojstava. ProfileRepository realiziran je kao Singleton odnosno postojat će samo jedna instanca tog repozitorija. Razred Profile također nasljeđuje apstraktnu klasu EntityBase koja služi za postavljanje Id-a profila.

ProfileFactory je tvornica za kreiranje profila. Također WindowsFormsFactory je tvornica za kreiranje novih instanci windows formi. Pozivanjem odgovarajućih metoda kreiraju se forme za unos, brisanje profila i karata te za prikaz popisa profila i izdanih karata. Sve forme nasljeđuju osnovnu klasu Form.

Entity razred Ticket predstavlja kartu koja se izdaje studentima. Sadrži atribute Mjesto1(predstavlja polazište), Mjesto2(predstavlja odredište), Price(predstavlja cijenu karte). Također implementira sučelje ITicketRepository te njegove metode za izdavanje karte te dohvat popisa svih izdanih karti.

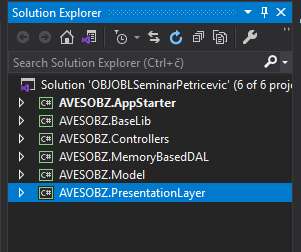
MainFormController razred sadrži reference na sučelja IMainFormController, IWindowFormsFactory, IProfileRepository, ITicketRepository. Kontroler je glavni kontroler koji sadrži metode za dodavanje novih profila, izradu karte te njihov prikaz. Metode pomoću tvornice formi kreiraju novu formu te pozivaju odgovarajući kontroler(ProfileController, TicketController).

# Arhitektura

Uz pomoć MVC( Model-View-Controller) patterna izradili smo AVESOBZ aplikaciju.

Aplikacije koje slijede MVC uzorak podijeljene su na odgovarajuće cjeline. Te cjeline omogućavaju da se složena aplikacija podjeli u manje dijelove gdje svaki dio ima svoju svrhu.

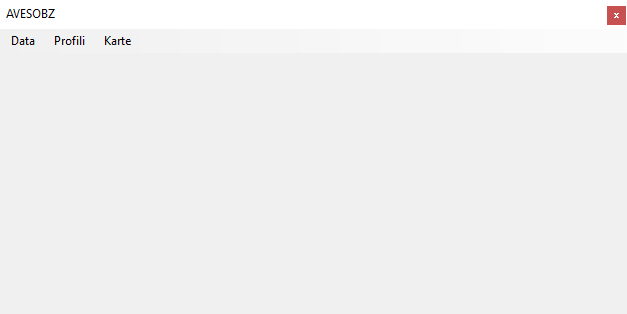
Kontroler se brine za obradu svih zahtjeva koje korisnik šalje aplikaciji. Kontroler provjerava informacije u modelu. U našem slučaju npr. ProfileController će dohvaćati podatke od modela Profile. Model se bavi unosom, uređivanjem i brisanjem podataka. Kontroler tada dobivene podatke od modela prezentira korisniku u viewu. View se brine samo za prikaz podataka korisniku. Slika 2 prikazuje logičku strukturu projekta.



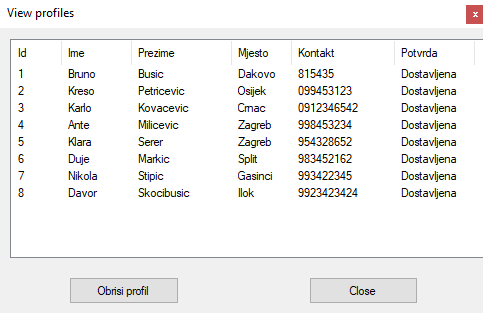
Slika 2. Struktura projekta

AppStarter predstavlja početnu točku za pokretanje aplikacije. Unutar tog projekta nalazi se main funkcija odgovorna start aplikacije. BaseLib je kolekcija sučelja za kontrolere. U projektu Controllers nalaze se svi kontroleri. MemoryBasedDAL sadrži sve potrebne repozitorije za odgovarajuće entitete. Model sadrži modele domene. PresentationLayer predstavlja view odnosno prezentacijski sloj krajnjem korisniku.

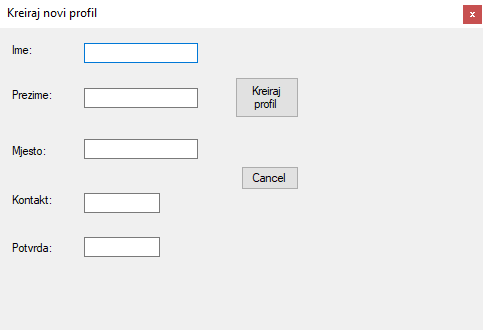
Sve forme prezentacijskog sloja funkcionalnosti prosljeđuju MainControlleru na način da pozivaju metode preko njegovog sučelja. U sljedećem dijelu pregledat ćemo osnovne forme za rad aplikacije.



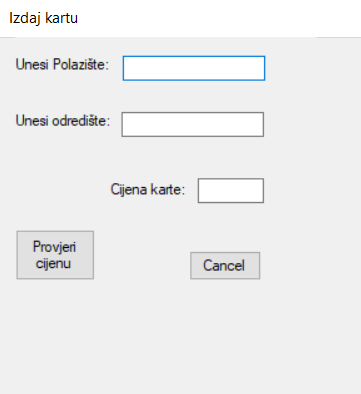
Slika 3. Početna forma aplikacije



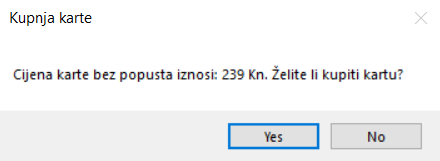
Slika 4. Forma koja prikazuje popis izrađenih profila



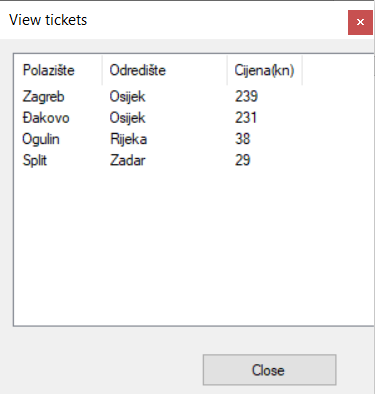
Slika 5. Forma za unos novog profila



Slika 6. Forma za izdavanje nove karte



Slika 7. Klikom na Provjeri cijenu gumb, otvara se dijalog za potvrdu kupnje karte



Slika 8. Forma koja prikazuje sve izdane karte

# O/R mapiranje za laboratorij profila

U sklopu seminara za labos profila potrebno je realizirati O/R mapper. Potrebno je definirati mapiranje za sve relevantne klase u modelu domene te podržati nekoliko scenarija use caseova.

O/R mapiranje ostvarit ćemo korištenjem Entity Frameworka. U sljedećem poglavlju dana je usporedba NHibernatea i Entity Frameworka.

## Usporedba NHibernatea i Entity Frameworka

NHibernate i Entity Framework koriste se za O/R mapiranje. Pružaju funkcionalnost izrade baze podataka na temelju objektnog modela.

NHibernate ima jako fleksibilne mogućnosti mapiranja. On kao ulaz uzima C# klasu sa svojstvima koja treba pohranit u bazu i kasnije dohvatit te podatke iz baze. Kao izlaz daje automatski generirani SQL query na zahtjev da nam učita odnosno pronađe objekt. NHibernate ne zahtjeva nasljeđivanje nikakvih posebnih baznih klasa ili sučelja. Objekti ne znaju za postojanje data store-a( što nije slučaj kod Entity Frameworka ali o tome nešto kasnije). Zbog toga klase je jednostavno testirati. NHibernate vrši mapiranje uz pomoć podataka o mapiranju u XML datoteci( za usporedbu s Entity Frameworkom koji ne zahtjeva nikakve XML podatke već automatski generira potrebne entitete odnosno klase). Potreban je jedan XML file po klasi. Na temelju tih fajlova se generiraju SQL upiti. Također, potrebno je postaviti globalnu konfiguraciju za koju se također koristi XML datoteka.

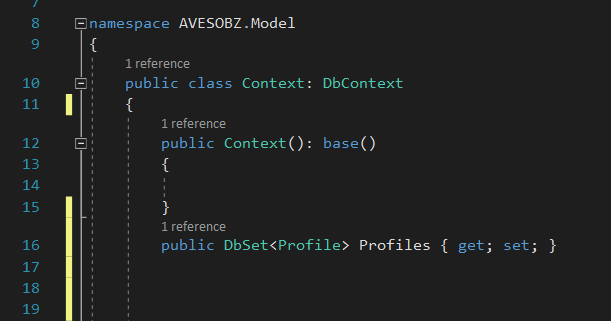
Entity Framework je O/R mapper kao nadogradnja nad ADO.NET-om. Vrši preslikavanje između modela i podataka u bazi podataka. Entity Framework automatski stvara odgovarajuće SQL upite. Kod Entity Frameworka postoji više načina kreiranja modela. Neki od njih su: Model First( model se dizajnira kroz grafičko sučelje, a baza podataka nastaje na osnovu modela), Database First( baza postoji, model nastaje reverznim inženjerstvom), Code First( model opisan kroz ručno izrađene razrede, a baza nastaje na temelju tih razreda), Code First from existing database( code First ali za postojeću bazu podataka). Ovisno o odabranom načinu kreirana izvršava se mapiranje.

Osnovna klasa biti će Context koja nasljeđuje klasu DbContext. Na ovaj način EntityFramework zna koje modele treba mapirati u bazu. Unutar te klase nalaze se podaci pohranjeni u skupu entiteta tipa DbSet. Generirani razredi su parcijalni pa se njihove definicije mogu nalaziti u više datoteka. Upiti nad EntityFramework modelom izvršavaju se dohvatom ili slanjem prvog podatka. Nakon uspješnog upita EntityFramework automatski vrši dohvat primarnog ključa koji je definiran kao tip identity. EntityFramework ne podržava dodavanje procedura u model.

## Ostvarivanje mapiranja

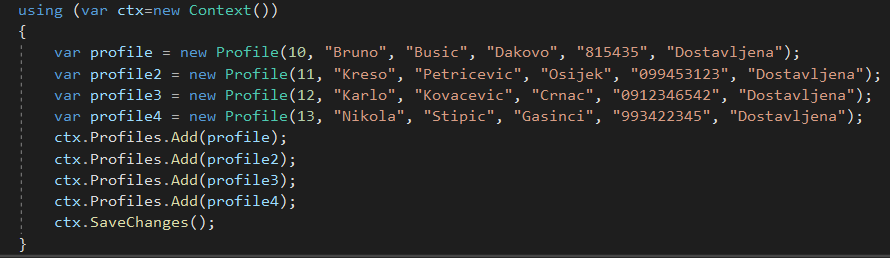
Najprije je potrebno dodati EntityFramework u naš projekt. To je moguće pomoću NuGeta.

Zatim u modele dodajemo klasu Context koja nasljeđuje klasu DbContext. Unutar ove klase dodajemo metode povratnog tipa DbSet<T> koji će se kasnije preslikati u entitete u bazi, a gdje T predstavlja model odnosno entitet. Isječak koda vidim na slici 9.



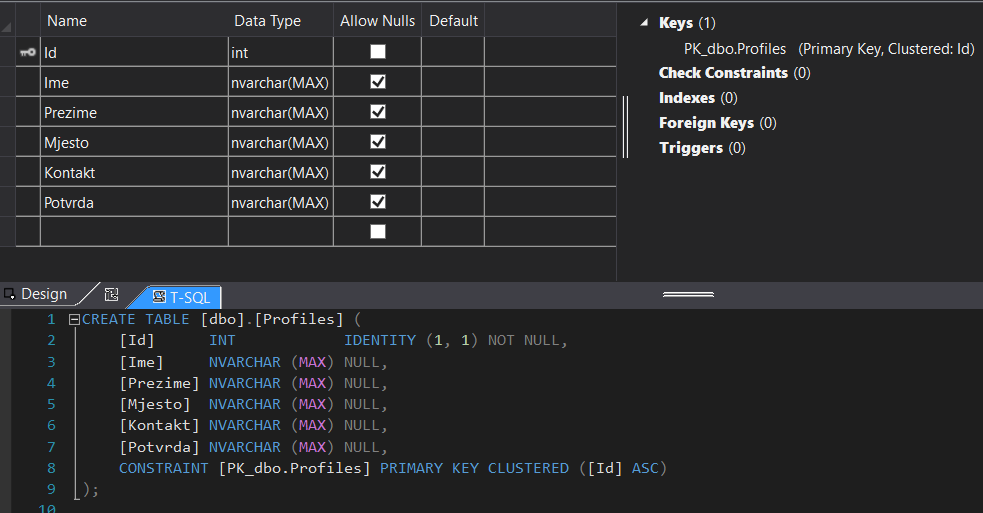
Slika 9. Klasa Context

Budući da je MainFormController klasa odgovorna za dodavanje profila u repozitorij, unutar te klase instanciramo objekt klase Context te ga punimo podacima. Kada nad navedenim objektom pozovemo metodu Add, automatski se generira upit bazi kojoj se šalju podaci. Ukoliko baza podataka ne postoji, ona je automatski kreirana, te su kreirane sve potrebne tablice sa svojim atributima i ključevima. Za bazu podataka koristimo SQLExpress koja dolazi kao dodatak u Visual Studio-u. Pozivom metode SaveChanges podaci su spremljeni u bazu te su potrebne tablice( tablica Profiles) generirane. Svi potrebni atributi biti će generirani u bazi automatski.



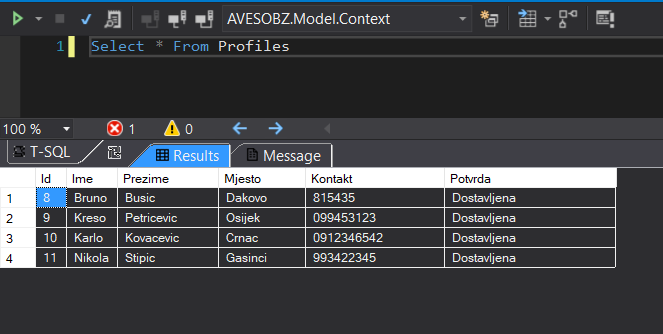
Slika 10. Isječak koda za kreiranje upita bazi te mapiranje entiteta

Slika 11. prikazuje automatski generiranu tablicu Profiles koja se generirala slanjem upita. Vidimo kako je EntityFramework sam prepoznao atribute te njihove DataTypove koje im pridružuje u bazi.



Slika 11. Automatski generirana tablica Profiles

Upitom Select nad tablicom Profiles dobivamo rezultat pohranjenih podataka u tablici. Vidimo kako je EntityFramework uspješno spremio podatke u automatski generiranu bazu.



Slika 12. Podaci u tablici Profiles