**SPECIFIKACIJA PROJEKTNE DOKUMENTACIJE**



**Fakultet tehničkih nauka Novi Sad**

**TURISTIČKA AGENCIJA**

**Marina Cvijetić**

**IT60/2019**

**SADRŽAJ:**

1. **Opis realnog sistema ....................................................................... 3**
2. **Zadatak ............................................................................................ 3**
3. **Opis tehnologije**
   1. **GenMyModel ..................................................................... 4**
   2. **SQL ..................................................................................... 4**
   3. **SQL Server Manegement Studio ..................................... 4**
   4. **Microsoft Visual Studio .................................................... 5**
4. **UML**
   1. **Dijagram slučajeva upotrebe ........................................... 6**
   2. **Opis dijagrama slučajeva upotrebe ................................. 7**
   3. **Dijagram klasa .................................................................. 10**
   4. **Dijagram sekvence ............................................................ 11**
5. **Baza podataka**
   1. **Kreiranje baze podataka .................................................. 12**
   2. **Upiti i DML naredbe**
      1. **Insert naredba ...................................................... 20**
      2. **Update naredba .................................................... 21**
      3. **Delete naredba ...................................................... 21**
      4. **Select naredba ....................................................... 22**
      5. **Select naredba + join ............................................ 23**
6. **Kreiranje WPF aplikacije**
   1. **Dodavanje ........................................................................... 27**
   2. **Izmena ................................................................................. 30**
   3. **Brisanje ............................................................................... 33**
   4. **Izgled aplikacije ................................................................. 35**
7. **Testiranje aplikacije ....................................................................... 37**
8. **Zaključak ........................................................................................ 41**

**OPIS REALNOG SISTEMA**

Turistička agencija je sistem čija je svrha da se olakša i ubrza vršenje poslovnih zadataka u okviru turističkih agencija. Aplikacija omogućava kreiranje rezervacija (odabir destinacije, vrste transporta, nacina placanja, odabir hotela, rezervacija karata itd) , evidenciju podataka o mušterijama, zaposlenima kao i kompletne podatke o rezervacijama kreiranim na osnovu zahteva mušterija.

Zaposleni se loguje u sistem koristeći svoj username i password ukoliko je već registrovan, u suprotnom neophodno je da se prvo registruje kako bi bio evidentiran u bazi podataka. Prvi korak kreiranja rezervacije jeste evidencija ličnih podataka mušterije kao i zaposlenog kako bi za svaku rezervaciju bio poznat kreator. Nakon toga vrši se evidencija broja osoba, datuma rezervacije, destinacije. Vrši se rezervacija soba u odabranom hotelu. Mušterija ima mogućnost izbora sopstvenog transporta, autobusa ili aviona. Ukoliko se opredeli za autobus ili avion, zaposleni vrši rezervaciju karata. Kako bi rezervacija bila kompletna, neophodno je da se evidentira i odabran način plaćanja (keš ili kartica).

**ZADATAK**

Cilj zadatka jeste kreiranje WPF (engl. Windows Presentation Foundation) aplikacije, povezane sa bazom podataka koja simulira rad turističke agencije. Cilj je da aplikacija omogući prikaz svih evidentiranih podataka o mušterijama, zaposlenima, rezervacijama. Takođe neophodno omogućiti kreiranje rezervacija, dodavanja novih podataka, izmena postojećih kao i brisanja željenih podataka.

**OPIS TEHNOLOGIJA**

**GenMyModel**

**GenMyModel** predstavlja online alat na čijoj se platformi mogu isprojektovati dijagrami različitih vrsta, poput UML, BPMN, Archimate, RDS, JPA, DMN, EMF, Flowchart koji nalaze široku primenu u predstavljanju realnih sistema I njegovom razumevanju. Alat je generisan kodom za Javu, Spring i SQL. Svi modeli su pravljeni na UML2 metamodelu i mogu se izvesti u više formata. U konkretnom primeru korišćen je dijagram slučajeva upotrebe i dijagram klasa.

**SQL**

**SQL** (engl. Structured Querz Language) je rwlacioni upitni jezik. Uniforman je jer se svi podaci i rezultati operacija prikazuju u vidu tabele i omogućava interaktivno i klasično programiranje. Ovaj jezik je neproceduralan, odnosno njime se specificiralo ŠTA, a ne i KAKO nešto treba uraditi. Jednostavan je za korišćenje i relacije se kreiraju jednom naredbom.

**SQL Server Management Studio**

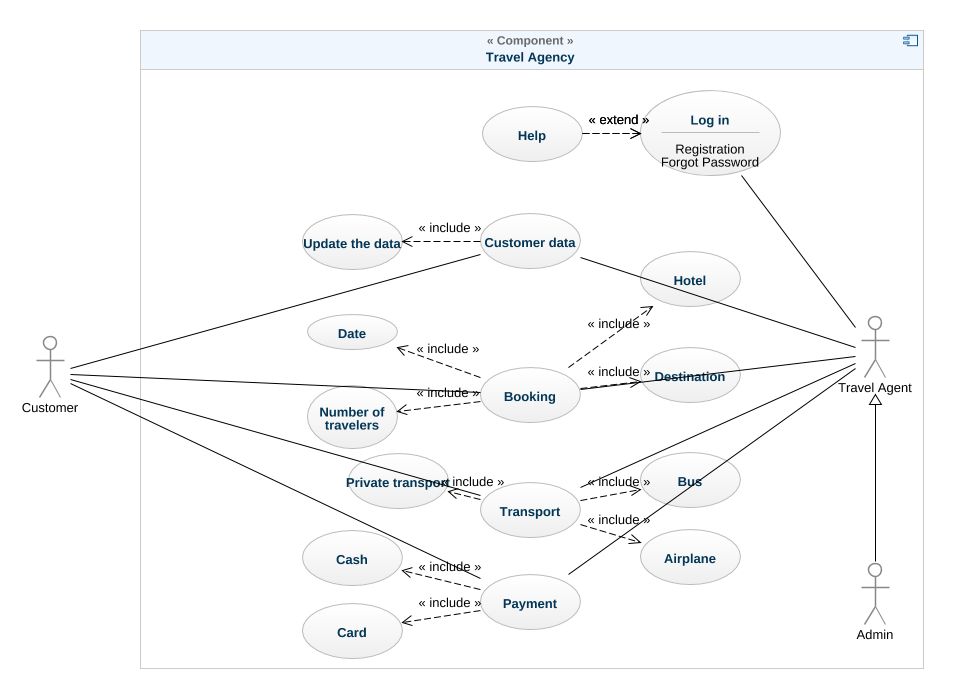
**SQL Server Management Studio** (SSMS) je softverska aplikacija koja se koristi za konfigurisanje, upravljanje i administraciju svih komponenti u okviru Microsoft SQL servera. Uključuje grafičke alate i skripte koji rade sa objektima i performansama servera. Centralna karakteristika SSMS-a je Object Explorer, koji omogućava korisniku da pretražuje, bira i deluje na bilo koji objekat unutar servera. Postoji posebno Express izdanje koje se može besplatno preuzeti i lako koristiti. Za izradu moje baze, koristila sam Microsoft SQL Server Management Studio 2012.

**Microsoft Visual Studio**

**Microsoft Visual Studio** je integrisano programsko okruženje, programirano od strane kompanije Microsoft. Visual Studio se koristi za programiranje računarskih igara, programa, veb sajtova, veb-servisa i veb-aplikacija na Microsoft-Windows-u. Visual Studio podržava 36 različitih programskim jezika, uključujući i C# koji smo mi koristili. Microsoft pruža i besplatnu verziju programa Visual Studio pod nazivom Community Edition koja podržava dodatke i dostupna je bez ikakvih troškova.

**UML**

**Dijagram slučajeva upotrebe**

****

**Opis dijagrama slučajeva upotrebe**

**Slučaj upotrebe: Logovanje**

**Kratak opis**: Radnik turističke agencije se loguje u sistem.

**Učesnici**: Radnik turističke agencije

**Uslovi** **koji moraju biti zadovoljeni pre izvršavanja**: Radnik ima mogućnost pristupa sistemu i zna svoj username i password.

**Opis**: Radnik pristupa sistemu, gde se prilikom pristupa traži njegovo korisničko ime i lozinka. Nakon unošenja korisničkog imena i lozinke, sistem proverava da li uneto korisničko ime postoji i da li je lozinka ispravna. Izuzetak: korisničko ime ne postoji i lozinka nije ispravna. Kada sistem utvrdi tačnost podataka, radnik pristupa sistemu.

**Izuzetak (Problemi pri lo**govanju): Otvara se pomoćni prozor.

**Uslovi koji moraju biti zadovoljeni nakon izvršenja**: Radnik je ulogovan u sistem.

**Slučaj upotrebe: Podaci mušterije**

**Kratak opis**: Radnik unosi podatke mušterije

**Učesnici**: Radnik turističke agencije, mušterija

**Uslovi** **koji moraju biti zadovoljeni pre izvršavanja**: Radnik ima mogućnost pristupa sistemu, mušterija ima validnu ličnu dokumentaciju

**Opis**: Mušterija prilaže validnu dokumentaciju radniku turističke agencije, nakon čega radnik unosi te podatke u sistem ili ih ažurira.

**Uslovi koji moraju biti zadovoljeni nakon izvršavanja**: Podaci o mušteriji su uneti u sistem, ili ažurirani ukoliko je to bilo potrebno.

**Slučaj upotrebe: Rezervacija**

**Kratak opis**: Radnik vrši rezervacije u skladu sa zahtevima mušterije

**Učesnici**: Radnik turističke agencije, mušterija

**Uslovi koji moraju biti zadovoljeni pre izvršavanja**: Radnik ima mogućnost pristupa sistemu, mušterija je izabrala destinaciju, hotel i datume prijavljivanja i odjavljivanja za željeni broj osoba, podaci su uneti u sistem.

**Opis**: Na osnovu svih podataka i zahteva mušterije o destinaciji, hotelu, datumima prijavljivanja i odjavljivanja i broju osoba, radnik rezerviše sobe u hotelu za željeni period i broj osoba.

**Uslovi koji moraju biti zadovoljeni nakon izvršavanja**: Rezervacije su potvrđene.

**Slučaj upotrebe: Transport**

**Kratak opis**: Radnik unosi u sistem vrstu prevoza koju mušterija bira i rezerviše prevozne karte ukoliko je to potrebno

**Učesnici:** Radnik turističke agencije, mušterija

**Uslovi koji moraju biti zadovoljeni pre izvršavanja**: Radnik ima mogućnost pristupa sistemu, mušterija je izabrala vrstu prevoza, radnik unosi podatke u sistem i rezerviše prevozne karte ukoliko mušterija nije izabrala sopstveni prevoz.

**Opis**: Na osnovu izabranog prevoza, radnik unosi podatke u sistem i po potrebi rezerviše avionske ili autobuske karte, kao i mesta.

**Uslovi koji moraju biti zadovoljeni nakon izvršavanja**: Podaci su uneti u sistem i karte su rezervisane.

**Slučaj upotrebe: Plaćanje**

**Kratak opis**: Mušterija plaća troškove rezervacije kao i usluge agencije

**Učesnici**: Radnik turističke agencije, mušterija

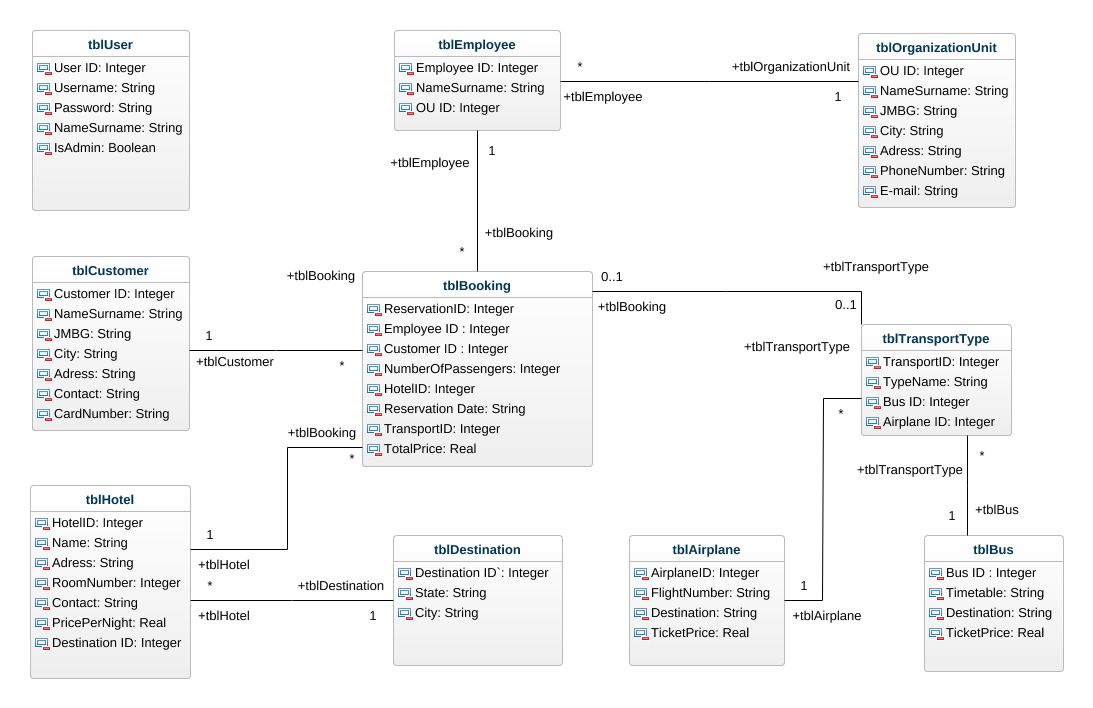
**Uslovi koji moraju biti zadovoljeni pre izvršavanja**: Radnik ima mogućnost pristupa sistemu, mušterija je izabrala način plaćanja, ukoliko je zabrala plaćanje karticom radnik mora imati validne podatke o kreditnoj kartici mušterije. Podaci uneti u sistem. Mušterija ima dovoljno sredstava za plaćanje

**Opis**: Nakon rezervacije mušterija plaća sve neophodne troškove i to može učiniti preko kreditne kartice, koja zahteva unos njenih podataka radi isplate, ili može platiti u kešu. Mušterija mora da zna potrebne podatke o kartici i da ima dovoljno sredstava za plaćanje.

**Uslovi koji moraju biti zadovoljeni nakon izvršavanja**: Troškovi su plaćeni

**Dijagram klasa**

**Dijagram klasa** prikazuje osobine i funkcionalnosti kreiranih klasa, njihove međusobne direktne i indirketne veze. Pravi se na osnovu dijagram slučajeva upotrebe.

****

**Dijagram sekvence**

**BAZA PODATAKA**

**Kreiranje baze podataka**

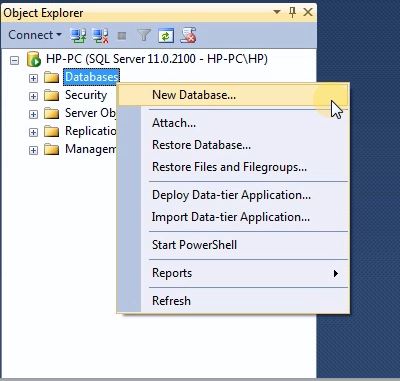
Pre samog kreiranja baze, neophodno je izvršiti konektovanje na izabrani server, pri čemu svaki računar ima svoj server kom pristupa. Otvaranjem Microsoft SQL Server Management Stidio softverske aplikacije, automatski se otvara prozor gde se zahteva izbor servera i konekcija.



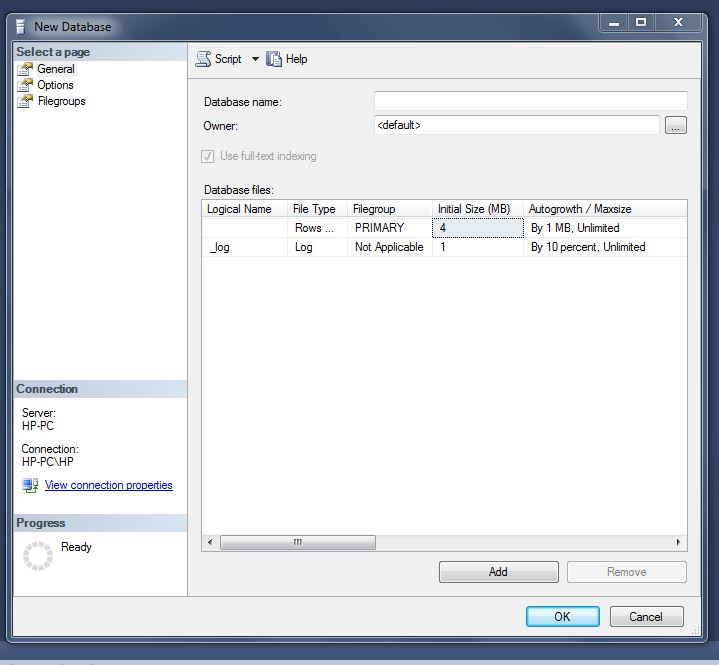
Klikom na *Connect*, izvršeno je konektovanje i računar je pristupio izabranom serveru.

Nakon toga, kreira se baza podataka koju je neophodno povezati sa aplikacijom. Kreiranje baze realizuje se pomoću sledećih koraka:

1. U okviru prozora *Object Explorer* desni klik na *Databases*
2. Otvara se lista opcija. Potrebno je kliknuti na *New Database.*



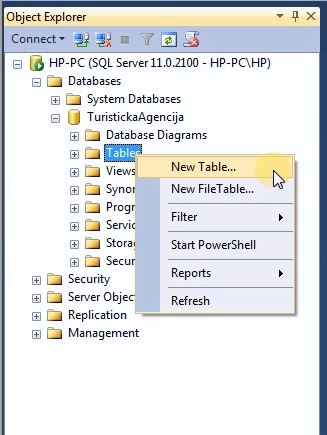
1. Otvara se prozor (pogledati sliku br. 6 na strani 14). U polje *Database name* upisuje se naziv baze podataka.



1. Klikom na dugme *OK* baza je kreirana.

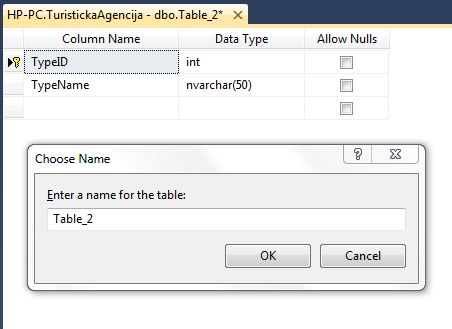
Nakon kreiranja baze, u skladu sa njenim dijagramom klasa kreiraju se tabele koje će odgovarati klasama u dijagramu.

1. U okviru prozora *Object Explorer* dvoklikom na:
   1. *Databases -> TuristickaAgencija -* navedeni folderi se prosiruju
   2. U okviru foldera baze (*TuristickaAgencija*) desnim klikom na folder *Tables* otvara se lista.
   3. Klikom na *New Table* kreira se nova tabela.



Nakon što je tabela kreirana u nju se unose odgovarajući atributi koji će opisivati unešene podatke.

Ime tabele se dodeljuje prilikom čuvanja, pri čemu je praksa da naziv tabele u sebi sadrži prefiks *tbl* , na primer ovde bi naziv tabele bio *tblType.*



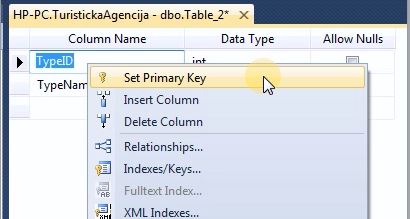
Neophodno je popuniti polja *Column Name* (naziv atributa), *Data Type* (tip podatka)i *Allow Nulls* (dozvola za null vrednost).

*Column Name*  sadrži atribute klase. *Data Type* kolona određuje tip podatka tog atributa (int, decimal, nvachar...), a *Allow Nulls* određuje da li pri popunjavanju tabele, može da se izostavi unos konkretnog atributa.

Svaka tabela u bazi ima svoj *Primary Key* odnosno primarni ključ koji može da se setuje na bilo koji atribut, ali najlogičnije je da primarni ključ tabele bude ID te tabele. Primarni ključ je tipa int i ne dozvoljava null vrednost.

Primarni ključ se dodeljuje na sledeći način:

1. Desnim klikom na željenu kolonu u tabeli otvara se lista
2. Klik na *Set Primary Key*



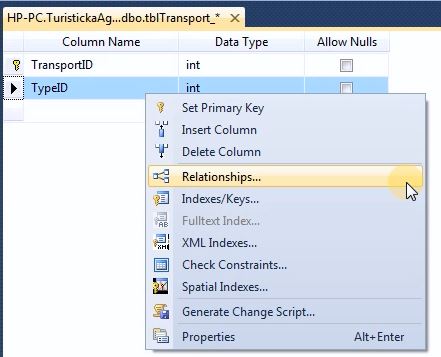
1. Zatim, u okviru prozora *Column Properties,* proširiti opciju *Identity Specification*
2. Postaviti opciju *Is Identity* na Yes
3. Opcija *Identity Increment* označava za koliko se uvećava sledeća vrednost ključa i ona ostaje 1, a *Identity Seed* označava od koje vrednosti počinju da se memorišu ključevi.

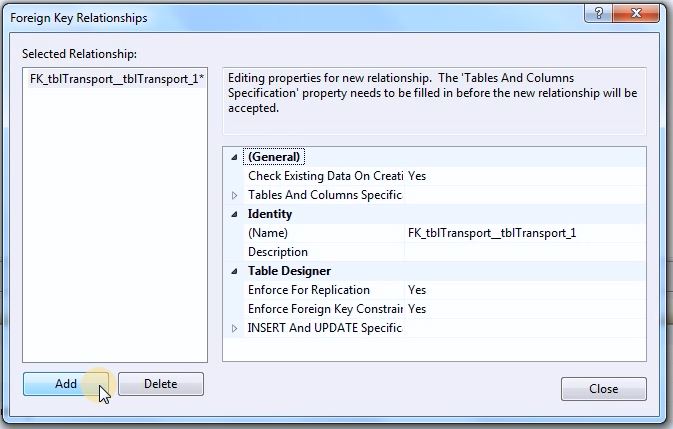
Neophodno je kreirati i ostale tabele kako bi baza bila kompletna, podesiti atribute i dodeliti kako primarne tako i strane ključeve koji povezuju odgovarajuće tabele.

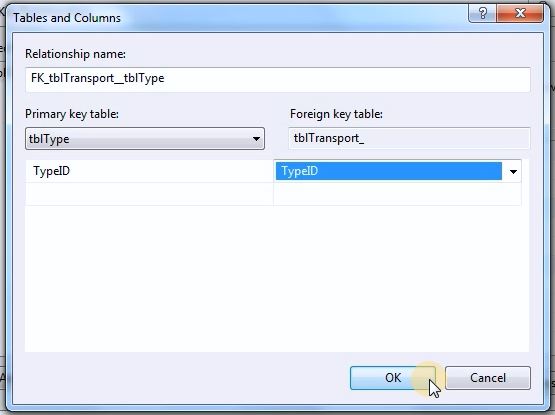
U bazama je neophodno da odredjene tabele sadrže **strane ključeve** (*Foreign key*) odnosno primarne ključeve drugih tabela jer se na taj način povezuju. Strani ključ u nekoj tabeli, omogućava pristup i manipulaciju podacima tabele čiji je strani ključ, iz tabele u kojoj se strani ključ nalazi.

Setovanje stranog ključa se vrši na sledeći način:

1. Desni klik na kolonu koju želimo da označimo kao *Foreign Key* a zatim *Relationships*



1. Zatim se otvara prozor u okviru kog je neophodno kliknuti na dugme *Add,* pri čemu će se pojaviti strani ključ koji treba podesiti.
2. Klik na *Tables And Columns Specific* a zatim na tri tačkice koje će se pojaviti sa desne strane nakon klika. Otvoriće se sledeći prozor.



1. Podesiti ključeve a zatim *OK*. Strani ključ je kreiran!

**UPITI I DML NAREDBE**

DML naredbe predstavljaju zahteve koji se upućuju bazi podataka i imaju za cilj manipulaciju podacima u bazi. Odgovor zahtevima predstavljaju podaci u odgovarajućem obliku.

**Insert naredba**

U okviru insert naredbe se navodi u koju tabelu unosimo podatke I koje vrednosti unosimo.. Zagrada sa atributima nije nužna, ali se pribegava njenom korišćenju kako bi se ispoštovao redosled unošenja podataka, ali I zbog sigurnosti prilikom unošenja. U okviru *values* upisuju se one vrednosti koje treba proslediti tabeli, vodeći računa o format I tipu podataka.

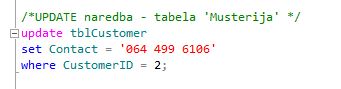
Insert naredbe baze *TuristickaAgencija:*

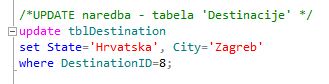


**Update naredba**

Update naredba služi kako bi ažurirali već postojeće podatke u bazi. U okviru sintakse navodi se tabela čije podatke ažuriramo, *set* služi za izmenu konkretnog podatka, a pomoću *where* definišemo gde se tačno vrši promena.

Update naredbe baze *TuristickaAgencija:*

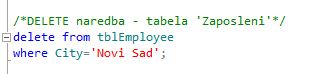


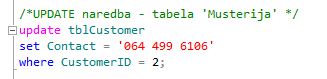
****

**Delete naredba**

Delete naredba služi kako bi obrisali željene podatke u tabelama baze. Navodimo tabelu u okviru koje želimo da brišemo podatke i takođe se koristi *where* kako bi definisali pripadajući ID podataka koji se brišu.

Delete naredbe baze *TuristickaAgencija:*

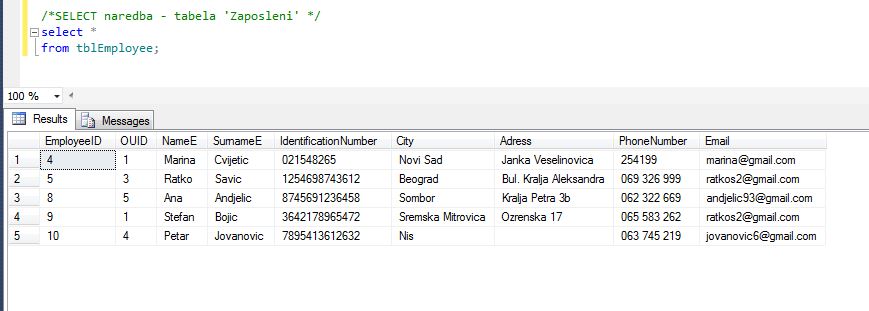


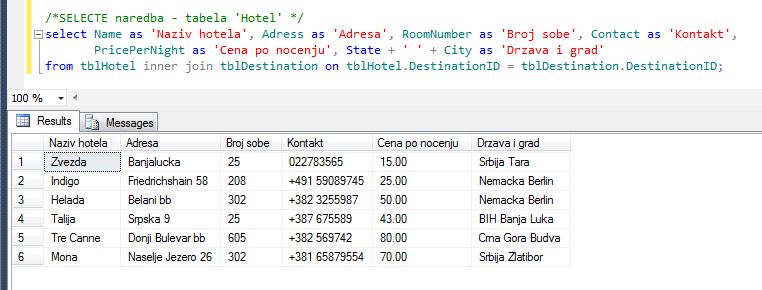


**Select naredba**

Select naredba omogućava iščitavanje svih ili pojedinih podataka tabela. Ukoliko želimo da iščitamo sve podatke iz tabele navodi se \* pri čemu nema mogućnosti menjanja naziva kolona u prikazu, a zatim se uz pomoć *from* definiše tabela čije podatke iščitavamo. Ukolikilo želimo samo pojedine kolone tabele da iščitamo, navodimo te kolone i moguće je menjanje naziva kolona u prikazu

Select naredbe baze *TuristickaAgencija:*

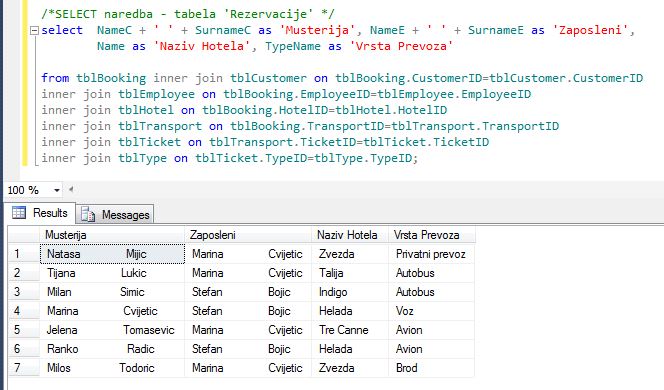
**



**Select + join**

Select naredba sa udruživanjem tabela, omogućava prikaz podataka ne samo iz jedne tabele, već iz svih tabela koje su međusobno povezane. Za to se koristi *Join.*

Select + join naredba baze *TuristickaAgencija:*



**KREIRANJE WPF APLIKACIJE**

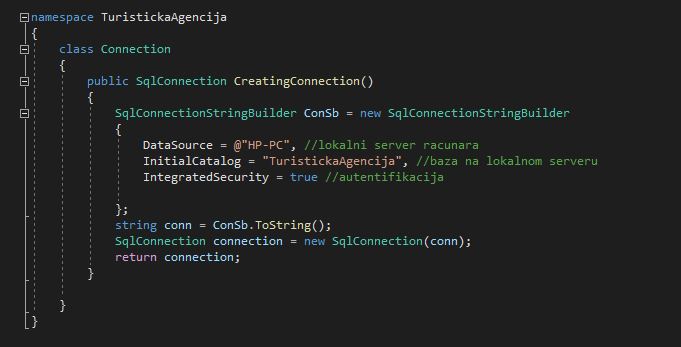
Nakon što je Visual Studio pokrenut, aplikacija se kreira na sledeći način:

1. *File*
2. *New Project*
3. *Windows Classic Desktop*
4. *WPF*
5. Imenovati aplikaciju

**WPF** aplikacija se sastoji iz nekoliko ključnih klasa. To su klasa konekcije, glavnog prozora i formi. Kreiranje konekcije ka bazi podataka je neophodno jer na taj način aplikacija se povezuje sa bazom i koristi njene podatke. U klasi *Connection* se kreira statička metoda kojom se ostvaruje veza sa bazom.

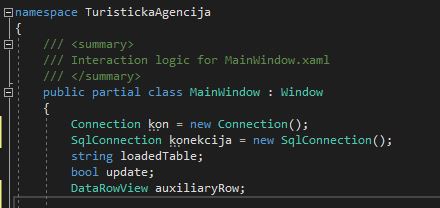
Kreiranje klase se vrši na sledeći način:

1. U okviru prozora *Solution Explorer* desni klik na projekat
2. *Add*
3. *Class* i novokreiranoj klasi dodeljujemo ime *Connection*



U *MainWindow.xaml.cs* koristimo klasu *Connection* koju smo mi kreirali kao na prethodnoj slici i kojoj prosledjujemo atribute pristupa bazi, kao i ugrađenu klasu *SqlConnection* koja kreira instancu konekcije u aplikaciji po osnovu prosleđenih atributa.

Takođe definišemo promenljive *loadedTable* tipa string, *update* tipa bool i *auxiliaryRow* tipa DataRowView. *LoadedTable* je promenljiva u kojoj će se čuvati podaci tabele koju je potrebno iščitati, dok *update* služi za proveru da li je u pitanju dodavanje ili ažuriranje podataka. *AuxiliaryRow* je promenljiva u kojoj će se čuvati selektovani red prilikom ažuriranja podataka.

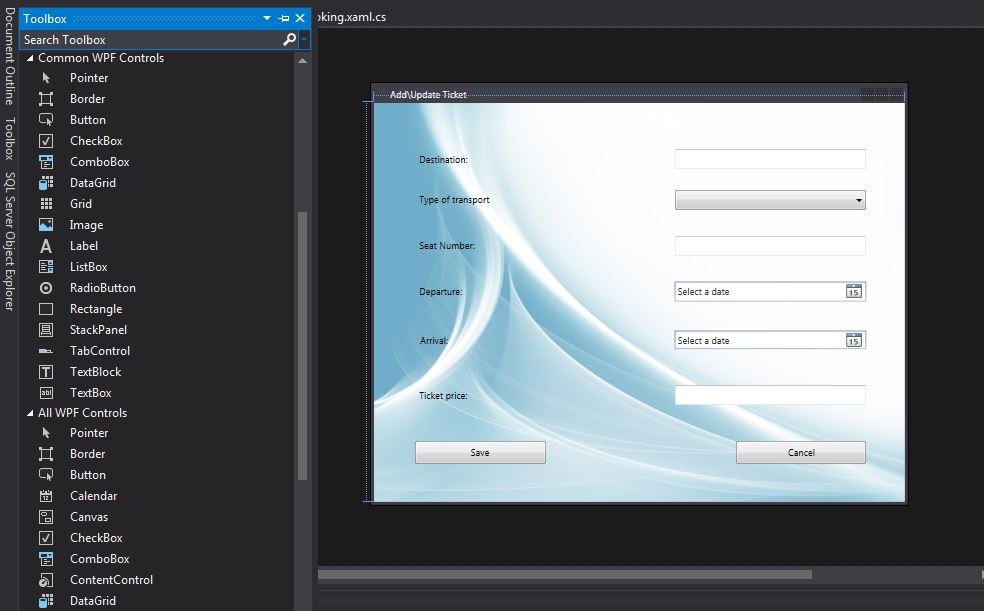


Kreiranje formi za dodavanje objekata u bazu podataka je sledeći korak. U okviru foldera *Forms* biće smeštene sve forme. Broj formi treba da bude jednak broju tabela u bazi.

WPF prozor za svaku formu kreira se desnim klikom na folder *Forms,* zatim *Add,* i na kraju *Window*. Imenovati prozor.

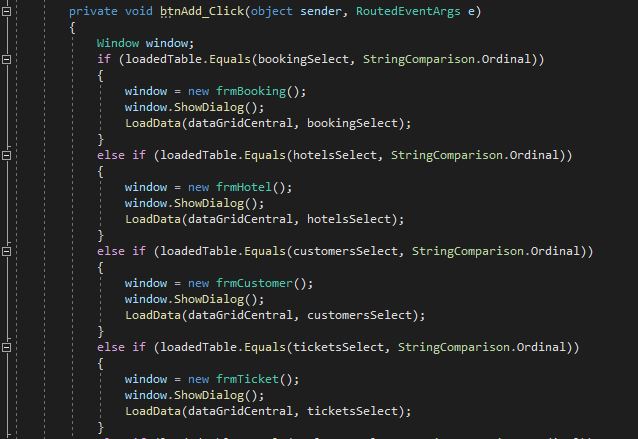
Nakon kreiranja prozora za svaku formu, neophodno je da se vizuelno uredi i da se dodaju svi neophodni elementi. U okviru *Toolbox-a* nalaze se sve WPF kontrole i elementi (dugmići, textbox polja, checkbox polja, labele itd).

Jedna od formi za odgovarajuću tabelu baze:

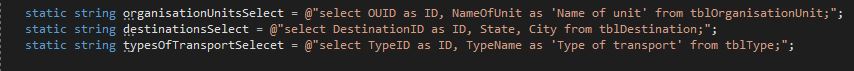


**Dodavanje**

Dvoklikom na dugme *Add* kreira se *btnAdd\_Click* funkcija. U njoj If uslovnom naredbom proveravamo koja tabela je učitana u centralnom gridu na osnovu select upita i u skladu sa tim otvara se forma učitane tabele.

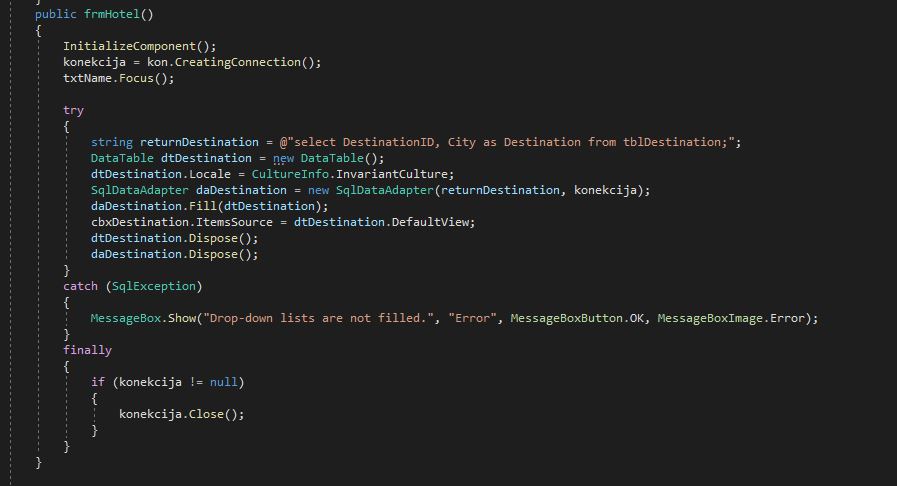


Select upit je prosledjuje bazi kako bi podaci iz baze mogli da se učitaju u centralnom gridu.



U behind kodu za svaku formu kreira se metoda za učitavanje padajućih lista (potrebnih za dodavanje i ažuriranje) ukoliko tabela sadrži strane ključeve. Takodje prosledju se select naredbe bazi kako bi definisali kojim podacima iz tabele popunjavamo padajuću listu. Svaka naredba nad bazom u kodu radi se u *Try, Catch, Finally* bloku. *Try* sadrži kod koji izvršava popunjavanje padajućih lista, *Catch* hvata SQL exception ukoliko postoji neka greška pri povezivanju sa bazom a u *Finally* se zatvara konekcija.

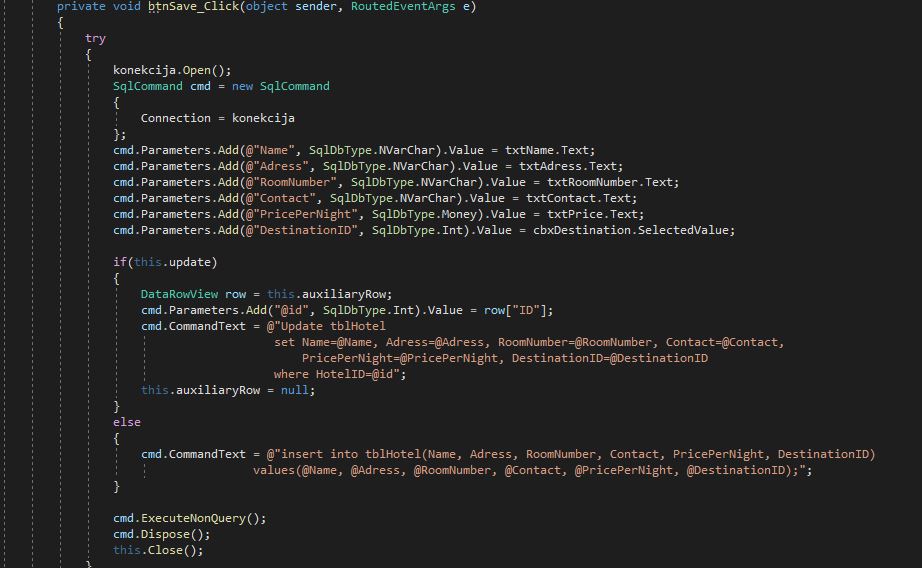
NAPOMENA: U xaml kodu za forme sa stranim ključevima, *SelectValuePath* mora da se poklapa sa nazivom ID-a tabele, a *DisplayMemberPath* mora da se poklapa sa nazivom kolone.



xaml:



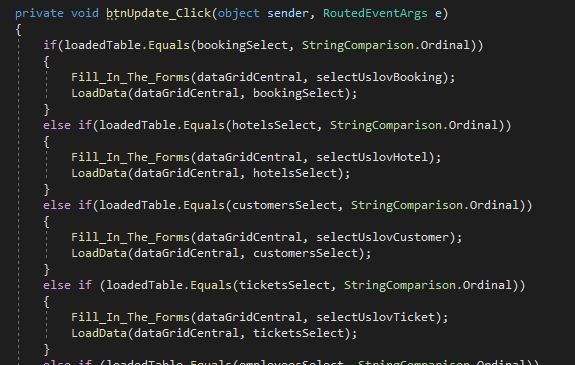
Nakon toga, kreira se kod koji nove podatke prosleđuje bazi kada se klikne na dugme *Save*. Takodje se sastoji iz *Try, Catch, Finally* bloka, ali *Try* kod je drugačiji. Prvo je potrebno ostvariti konekciju sa bazom, zatim se kreira komanda koja unešene podatke iz forme, prosleđuje bazi. If uslov proverava da li je u pitanju Ažuriranje podataka ili dodavanje novih. Pošto je u pitanju dodavanje i vrednost promenljive *update* je *false*, izvršava se naredba *insert*. Neophodno je pre navodnika dodati karakter @ kako bi naredba bila prosleđena bazi kao string.



*Catch i Finally* su isti kao i na prethodnoj slici.

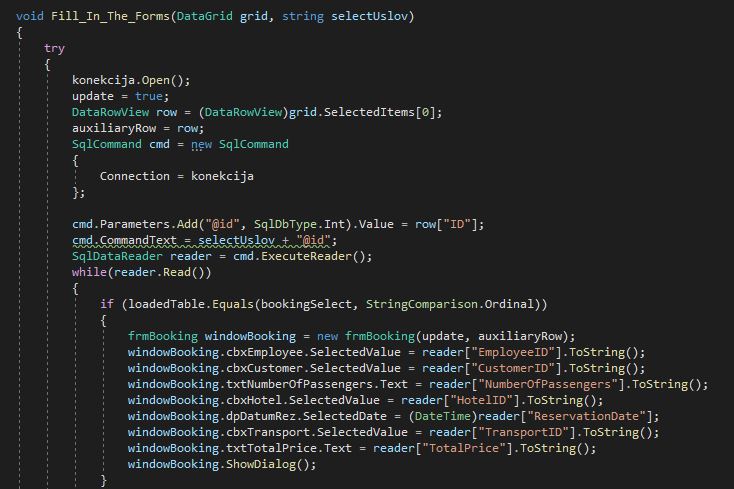
**Izmena**

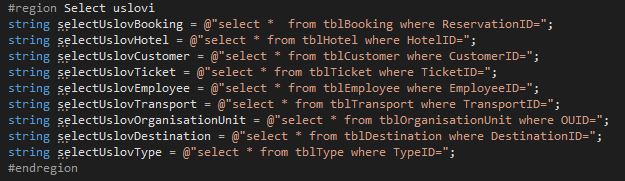
Proces izmene podataka se započinje izborom tabele i klikom na dugme *Update* u formi glavnog prozora koji se prvi otvara prilikom pokretanja aplikacije. U mainu se dvoklikom na dugme *Update* kreira funkcija *btnUpdate\_Click* u kojoj se kao i za dodavanje prvo proverava koja je tabala učitana.



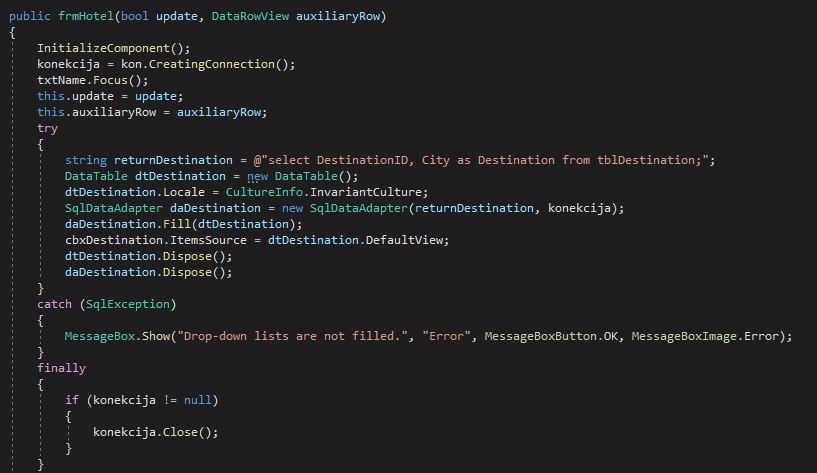
U if uslovu se proverava koja je tabela učitana pomoću promenljive *loadedTable* i select upita. Zatim se poziva funkcija koja popunjava formu sa podacima iz tabele.

Funkcija koja će popuniti formu sa podacima reda koji se nalaze u tabeli baze se naziva *Fill\_in\_the\_forms*. Prvo se otvara konekcija nad bazom. Vrednost *update* promenljive setuje se na *true* jer kada se izvršava ažuriranje, na taj način program zna koju naredbu treba da izvršava. Vrednost pomoćnog reda se setuje na početni ID (*[0]*) i pomoću njega preuzima se ID selektovanog reda iz aplikacije, a zatim se ID tog reda iz baze, konkatanacijom lepi na selekt uslov namenjen za operaciju ažuriranja. Koristi se čitač kako bi se iščitali podaci i popunili u formu u skladu sa izabranim redom.

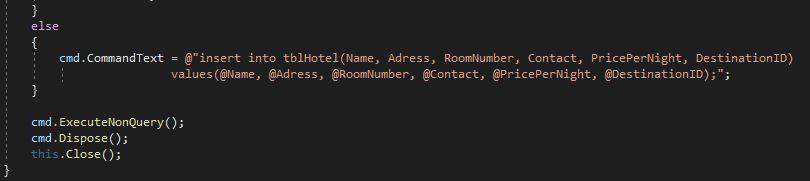




U svakoj formi se kreira ista metoda za padajuće liste kao i u slučaju dodavanja, ali se sada prosleđuju parametri promenljiva *update* i *auxiliryRow* odnosno pomoćni red.

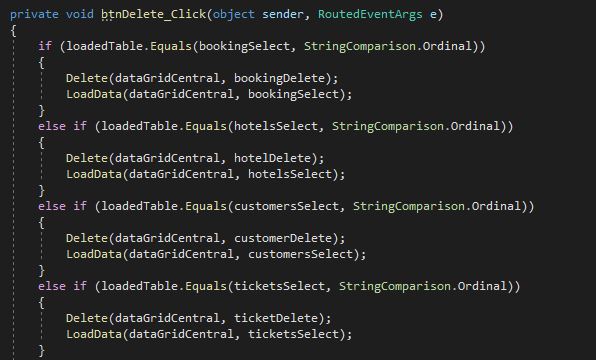


Kada korisnik popuni formu izmenjenim podacima i klikne dugme *Save* izvršava se *Update* naredba umesto insert naredbe i podaci se ponovo iščitavaju nakon zatvaranja forme i promena se prikazuje.

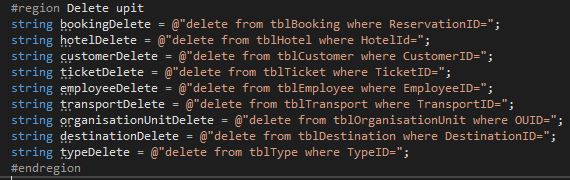


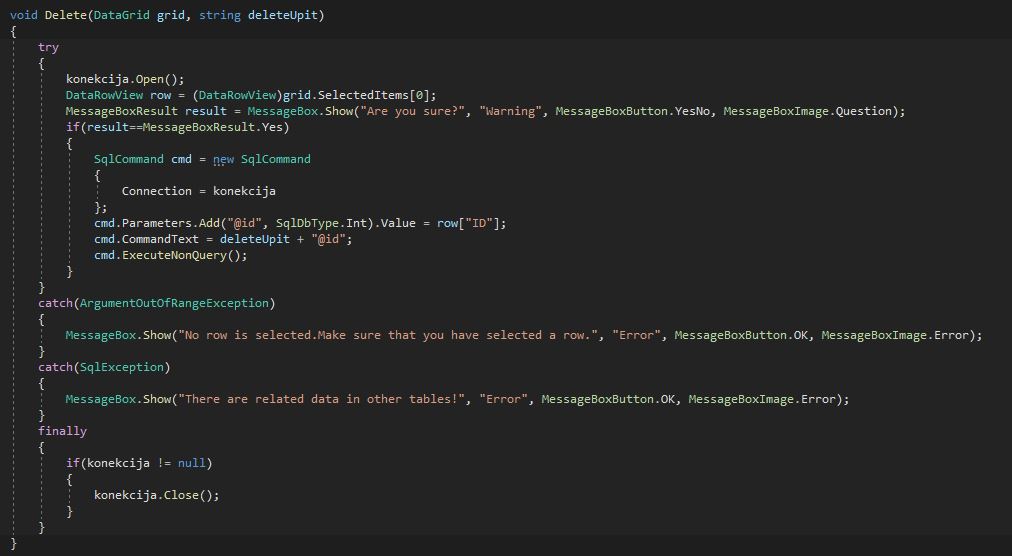
**Brisanje**

Isto kao i u prethodnim slučajevima za dodavanje i brisanje, dvoklikom na dugme *Delete* kreira se funkcija *btnDelete\_Click* koja proverava koja je tabela učitana i poziva funkciju *Delete.*



Funkcija *Delete* ima prosleđene parametre *grid* i *deleteUpit*. Takodje se koristi *Try,Catch,Finally* blok. Red mora biti selektovan i promenljiva *row* tipa *DataRowView* čuva koji je red selektovan. Konkatanacijom se ID selektovanog reda lepi na delete upit koji se prosleđuje bazi.

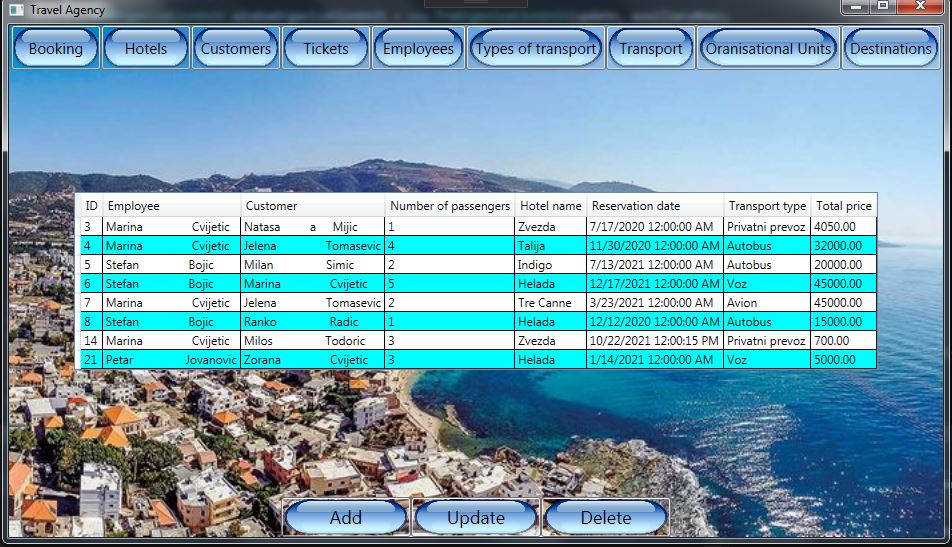


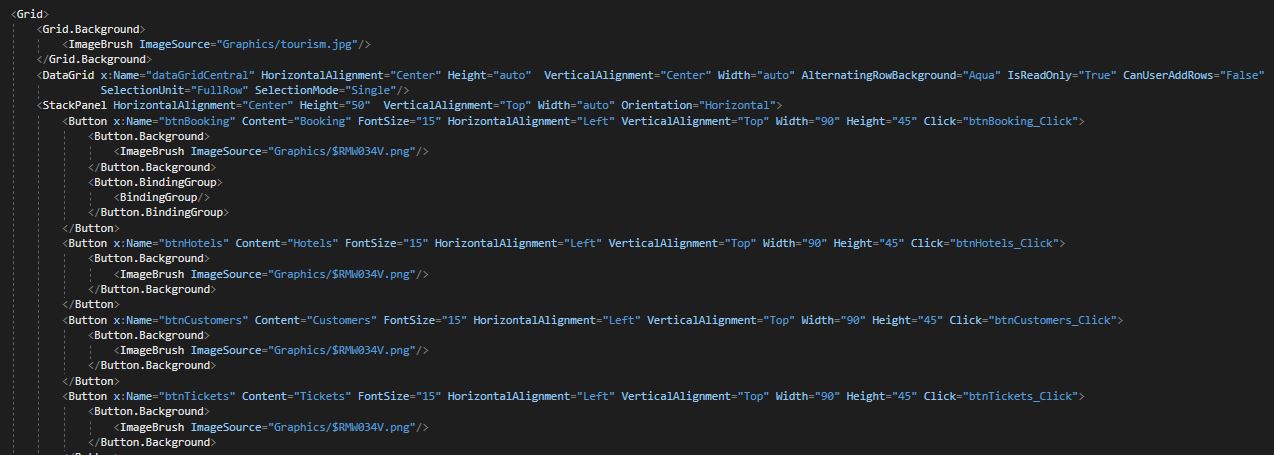


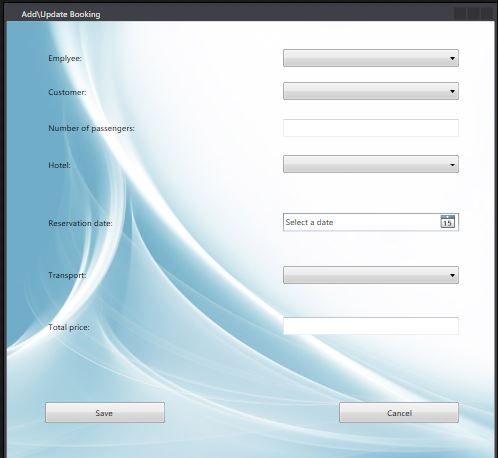
Pre brisanja pojavljuje se poruka koja traži potvrdu brisanja. Takodje postoji poruka upozorenja da red mora biti selektovan. Ukoliko su podaci povezani sa drugim tabelama, pojavljuje se poruka da brisanje nije moguće.

**IZGLED APLIKACIJE**

Izgled aplikacije se definiše kodom u xaml prozoru, kao što sam već rekla pomoću toolBox-a i mnogih drugih funkcionalnosti i podešavanja.







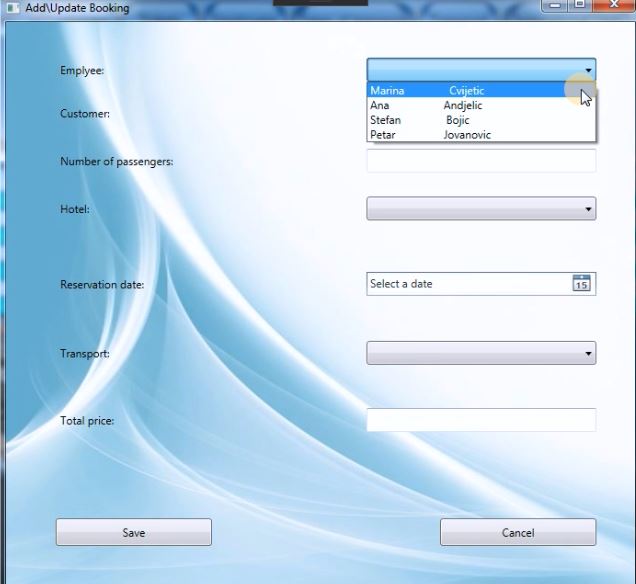


**TESTIRANJE APLIKACIJE**

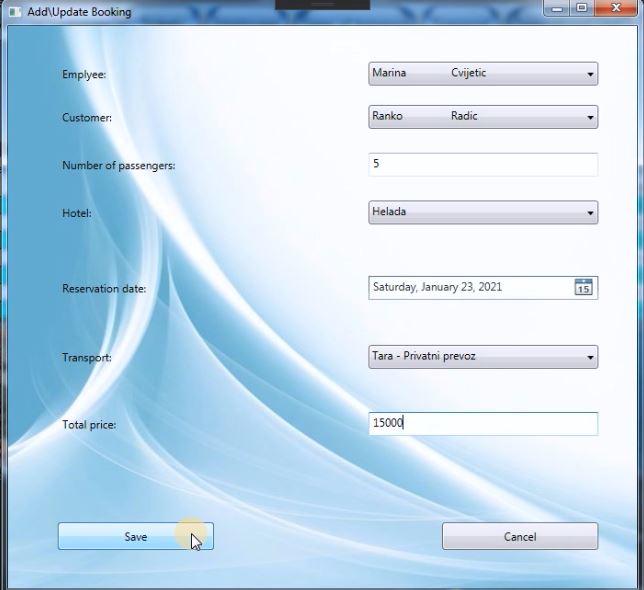
Dodavanje:



Pomoću dugmića u gornjem delu prozora biramo tabelu u koju želimo da unesemo podatke. Klikom na dugme *Add* započinje se dodavanje.



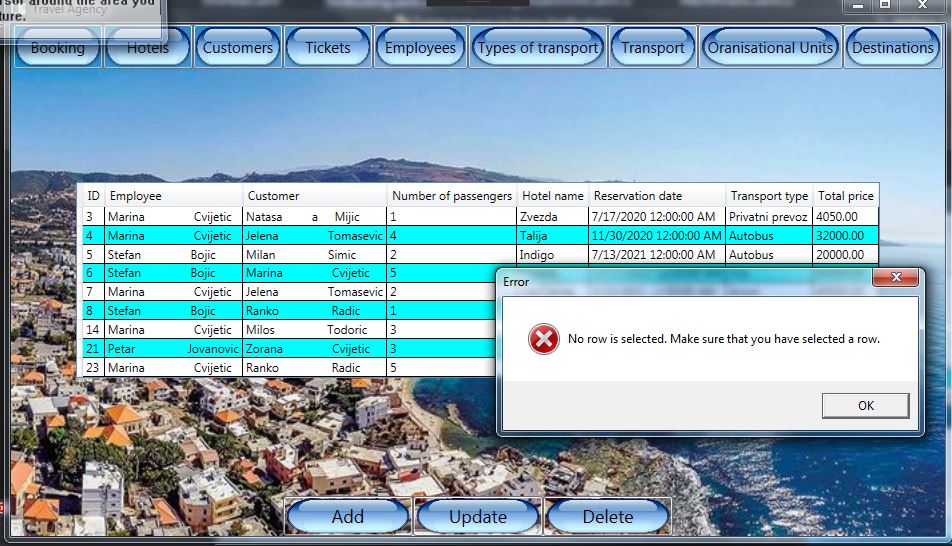
Neophodno je popuniti sva polja, u suprotnom pojavljuje se prozor greške.



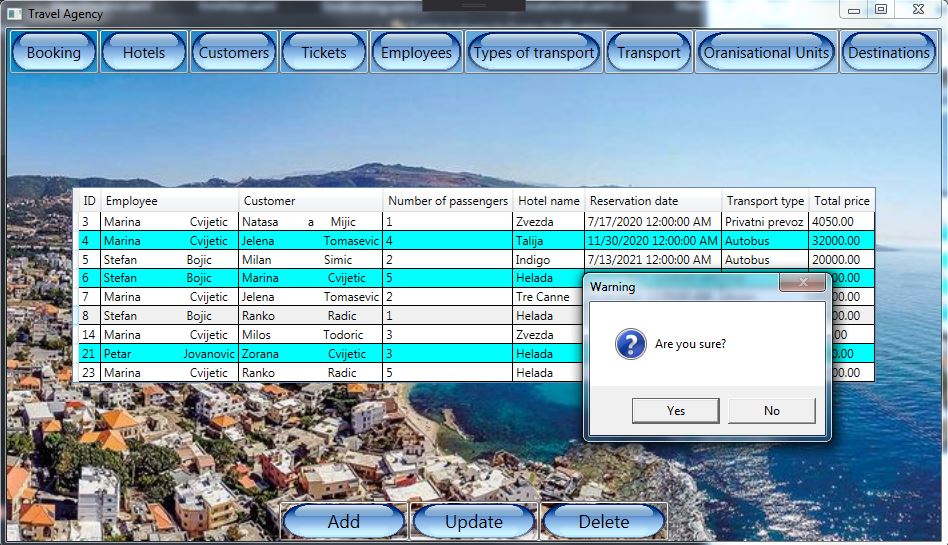


Kada smo popunili sva polja u formi, klikom na *Save* podaci se dodaju u tabelu u bazi a zatim se iščitava ažurirana tabela sa dodatim podacima.

Ukoliko pokusamo da pokrenemo ažuriranje a nismo selektovali red dešava se sledeće:



A ukoliko zelimo da obrisemo selektovani red, potvrdom na *Yes* red ce biti obrisan.



**ZAKLJUČAK**

Analizom svakog koraka kreiranje aplikacije i baze, počevši od ideje, planiranja i postepene realizacije, najpre dijagrama upotrebe slučajeva, dijagrama klasa i sekvence, zatim i same baze i na kraju aplikacije, objasnili smo kako aplikacija funkcioniše. Kreiran je informacioni sistem u obliku WPF aplikacije i koji treba da olakša poslovanje u turističkim agencijama. Aplikaciju je neophodno usavršiti, kako bi u potpunosti bila pogodna za lako i brzo korišćenje. Dokumentacija aplikacije treba da olakša razumevanje njenog funkcionisanja i rada, ali može da posluži i kao priručnik za kreiranje aplikacije namenjene potpuno drugoj oblasti poslovanja.