Microsoft tehnologije za pristup podacima
Projekti #1

1. ADO.NET Disconnected klase

Pomoću objekata DataTable, DataSet, DataColumn, DataRow i DataRelation kreirati relacioni model po sledećoj specifikaciji:

Entitet Kupci sa sledećim atributima i njihovim svojstvima:

- **KupacID**, int, autoincrement (počev od 1, korak 1), nisu dozvoljene NULL vrednosti, primarni ključ
- **NazivKupca**, string maksimalne dužine 50 karaktera, nisu dozvoljene NULL vrednosti
- **Adresa**, string maksimalne dužine 200 karaktera, dozvoljene su NULL vrednosti

Entitet Fakture sa sledećim atributima i njihovim svojstvima:

- **FakturaID**, int, autoincrement (počev od 1, korak 1), nisu dozvoljene NULL vrednosti, primarni ključ
- **KupacID**, int, nisu dozvoljene NULL vrednosti, spoljni ključ - referiše se na entitet **Kupci**, kolonu **KupacID**
- **Datum**, datetime, nisu dozvoljene NULL vrednosti, default vrednost je trenutni datum i vreme

Entitet FaktureStavke sa sledećim atributima i njihovim svojstvima:

- **FakturaID**, int, nisu dozvoljene NULL vrednosti, spoljni ključ - referiše se na entitet **Fakture**, kolonu **FakturaID**
- **NazivStavke**, string maksimalne dužine 40 karaktera, nisu dozvoljene NULL vrednosti
- **CenaStavke**, decimal, nisu dozvoljene NULL vrednosti
- Primarni ključ ovog entiteta je kompozitni i sastoji se od atributa **FakturaID** i **NazivStavke**

Specifikacije:

- Za obe relacije postaviti UPDATE i DELETE RULE na **none**.
- U **Windows Forms** tipu aplikacije pomoću **DataGridView** kontrola, prikazati podatke iz sva tri entiteta. Sve uraditi na jednoj formi.
- Omogućiti dodavanje novih slogova za sva tri entiteta preko odgovarajućeg korisničkog interfejsa.
- Svuda uraditi obradu greške i odgovarajuće notifikacije u slučaju greške.
- Omogućiti serijalizaciju sva tri entiteta u jedan XML fajl koji treba da sadrži XML šemu i podatke.

2. ADO.NET Connected klase

Koristi se baza podataka TSQL i Windows Forms aplikacija.

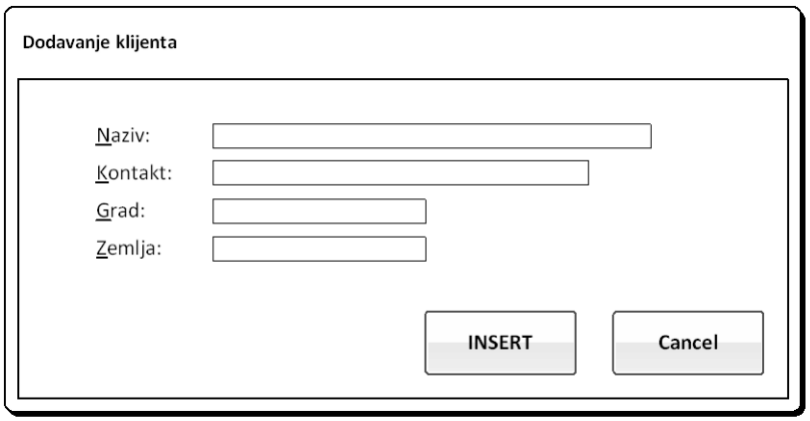
Kreirati novu tabelu u TSQL bazi podataka i napuniti je podacima pomoću sledećeg TSQL skripta:

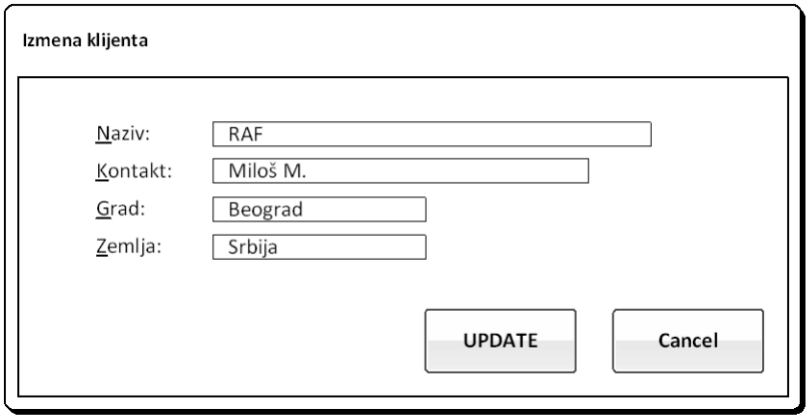
```
USE TSQL;
GO
IF (OBJECT_ID('dbo.Klijenti') IS NOT NULL) DROP TABLE dbo.Klijenti;
GO
CREATE TABLE dbo.Klijenti (
      KlijentId int IDENTITY PRIMARY KEY NOT
      NULL, Naziv nvarchar(40) NOT NULL, Kontakt
      nvarchar(30) NOT NULL,
      Grad nvarchar (15) NOT NULL,
      Zemlja nvarchar(15) NOT NULL);
GO
INSERT INTO dbo.Klijenti
(Naziv, Kontakt, Grad, Zemlja)
SELECT companyname, contactname, city,
country FROM Sales.Customers;
```

Za tabelu **dbo.Klijenti** kreirati Windows aplikaciju sa odgovarajućim korisničkim interfejsom koji omogućava **CRUD** (**Create Read Update Delete**) operacije.

String konekcije koji koristi SqlConnection klasa čuvati u **App.config** fajlu. Početna forma aplikacije treba da obezbedi pregled podataka u kontroli **dataGridView** i da ima sledeći okvirni izgled:

Za akcije INSERT i UPDATE obezbediti korisnički interfejs na novim formama gde se omogućava unos, odnosno izmena podataka. Na sledeće dve slike su prikazane forme za unos novog i izmenu postojećeg sloga:





Na strani korisničkog interfejsa uraditi sve potrebne validacije pre pokretanja INSERT ili UPDATE akcije. DELETE akciju uraditi direktno na DELETE dugmetu početne forme uz predhodno kontrolno pitanje - "Da li ste sigurni da želite da obrišete klijenta", pomoću MessageBox klase.

Ko želi, može INSERT i UPDATE akcije implementirati na jednoj istoj formi gde će se dinamički, u vreme izvršenja aplikacije, kontrolisati ponašanje i interfejs.

Sve potrebne akcije na strani baze podataka implementirati **kroz stored procedure** koje se pozivaju sa strane klijentske aplikacije. Stored procedure trebaju imati obradu grešaka. Povratna vrednost stored procedura treba da bude indikator njihovog uspešnog (povratna vrednost=0) ili neuspešnog izvršavanja i obavezno je kontrolisati na strani korisničkog interfejsa.

Pozive ovih procedura implementirati kao funkcije u klasi **clsDataAccess** koja se nalazi u okviru aplikacije - kreiramo dvoslojnu arhitekturu. Dakle, treba imati četiri stored procedure i četiri metode u klasi **clsDataAccess**. U kodu metoda klase uraditi obradu grešaka.