Predmetni projekat (30 poena)

Opšti zadatak

Potrebno je razviti klijent-server aplikaciju za rad sa zadatim modelom podataka na osnovu *Unified Modeling Language* (UML) dijagrama. Model podataka se sastoji od instanci primarne klase i instanci pomoćnih klasa. Očekuje se primena barem 4 obrasca i pridržavanje **SOLID** principa.

Klijentska aplikacija

Klijentsku aplikaciju razviti u *Windows Presentation Foundation* (WPF) tehnologiji, oslanjajući se na *Model-View ViewModel* (MVVM) obrazac (ne računa se u 4 tražena obrasca). Pri tome, treba omogućiti:

- Prikaz svih instanci primarne klase, sa pomoćnim prikazima instanci ostalih klasa.
- Pretragu instanci primarne klase po vrednostima svih svojstava.
- Dodavanje novih instanci primarne i pomoćnih klasa, koje zahteva prethodno validaciju svih polja preko kojih se unose vrednosti svih atributa, pre čuvanja samih instanci u okviru klijentske i serverske aplikacije.
- Izmenu i uklanjanje (brisanje) instanci primarne klase.
- Poništavanje i ponovno izvršavanje komandi izvršenih nad instancama primarne klase (undo/redo).
- Na osnovu već definisanih stanja u kojima se mogu naći instance primarne klase, za svaku dodatu instancu, simulirati promene kroz sva stanja i ispisivati ih na interfejsu klijentske aplikacije.
- Implementirati da se promene atributa nad instancama primarne klase ažuriraju na prikazu svih instanci.
- Za izdvojenu klasu sa nekompatibilnom strukturom, omogućiti njenu integraciju u sistem.
- Prikazivati, *u realnom vremenu*, koliko instanci primarnih klasa je u kojem od stanja u vidu grafikona. Za grafikone koristiti biblioteku LiveCharts2.

Serverska aplikacija

Serversku aplikaciju razviti u *Windows Communication Foundation* (WCF) tehnologiji, kroz definiciju interfejsa za obradu svih funkcionalnosti koje dodaju/menjaju podatke sistema. Omogućiti čuvanje i učitavanje podataka u *eXtensible Markup Language* (XML), *Comma-Separated Values* (CSV) i *JavaScript Object Notation* (JSON) fomatima datoteka. Pri pokretanju serverske aplikacije, njen korisnik bira format čuvanja podataka.

Logovanje aktivnosti

Omogućiti logovanje svih aktivnosti na strani klijenta i na strani servera. Beležiti aktivnost u tekstualnoj datoteci kroz sledeće informacije: datum i vreme, tip događaja (DEBUG, INFO, WARN, ERROR, FATAL) i poruka koja ga detaljnije opisuje. Za potrebe logovanja, preporučuje se biblioteka log4net.

Napomene:

- Dobijeni *UML* dijagram proširiti podacima koji nedostaju:
  - Dodati sva potrebna polja, svojstva (*property*) i metode,
  - Dopuniti dijagram i dodatnim klasama, ako za to postoji potreba,
  - Ne menjati osnovnu strukturu dobijenog *UML* dijagrama (apstraktne klase, relacije, implementirati metode)
  - Primarna i izdvojena klasa su posebno označene u datum *UML* dijagramu.
  - Stanja u kojima može biti instanca primarne klase i traženi tip grafikona je naveden u komentarima unutar *UML* dijagrama
- Prikaz instanci svih klasa se može realizovati koristeći kontrole koje prikazuju više objekata (*DataGrid*, *ItemsControl*, *ListView*...) i prikazuje sve atribute te instance.
- U podacima serverske aplikacije moraju postojati barem tri instance primarne klase koje se uvek učitavaju na početku, ako je datoteka iz koje se učitava prazna.

- Pored korektno kompletiranog zadatka, na broj osvojenih bodova utiče i nivo kvaliteta dizajna (korišćenje projektnih obrazaca) i obavezno je sve korišćene projektne obrasce, pored implementacije, uključiti i u dati *UML* dijagram.
- U toku implementacije rešenja **pridržavati se SOLID principa**.
- Prilikom pisanja koda, poštovati *C#* konvencije za imenovanje, i pisati što „čistiji“ kod (nazivi kontrola, promenljivih, funkcija, klasa itd) i razdvajati celine rešenja prema ulozi (*Model*, *View*, *ViewModel*, *Helpers*, *Interfaces* i *Services*). Jezik na kome se piše kod mora biti **konzistentan kroz čitavu aplikaciju**. Jezik korisničkog interfejsa takođe mora biti **konzistentan kroz čitavu aplikaciju**.
- Neophodno je znati do najsitnijih detalja šta svaka linija koda u rešenju radi i čemu služi. Za neadekvatna objašnjenja koda i UML dijagrama, nepoznavanje i nerazumevanje sopstvenog koda **će se gubiti bodovi**. Isto važi i za **prepisivanje i plagijarizam**.
- *Pratiće se učestalost i regularnost komitovanja na repozitorijum, sa ciljem da se osigura kontinuirani razvoj i pravovremeno verzionisanje koda*.