Zadatak 4

Proračuni na osnovu zahteva

Aplikacija se sastoji od servisne i klijentske aplikacije. Servis i klijentska aplikacije komuniciraju putem WCF-a. Klijentska aplikacija je konzolna aplikacija koja ima opciju unosa naredbe koja se prosleđuje servisu.

1. Scenario rada aplikacije

Scenario korišćenja aplikacije je sledeći (preduslov je da se pokrenu servisna i klijentska aplikacija):

1. Korisnik sačuva CSV datoteku na definisanoj lokaciji na računaru na kom se nalazi klijentska aplikacija. Ova CSV datoteka sadrži podatke o ostvarenim vrednostima potrošnje električne energije, po satima, za današnji dan.
2. Korisnik upiše u konzolu klijentske aplikacije naredbu „Send“. Ovim se datoteka poslala servisnoj aplikaciji, i izbrisala se sa lokacije gde je prethodno sačuvana.
3. Koraci 3 i 4 se ponavljaju kao stižu sveži podaci o potrošnji električne energije.
4. Korisnik upiše u konzolu naredbu „Get“, sa parametrima. Broj parametara može da bude od 1 do 3. Parametri mogu da imaju vrednosti: „min“, „max“, „stand“. Parametar „min“ označava da se očekuje prijem podatka o minimalnoj potrošnji tekućeg dana. Parametar „max“ označava da se očekuje prijem podatka o maksimalnoj potrošnji tekućeg dana. Parametar „stand“ označava da se očekuje prijem podatka o standardnoj devijaciji potrošnje u tekućem danu.

2. Poslovna logika

Servisna aplikacija čuva pristigle podatke u bazi podataka koja je implementirana kao datoteka TBL_LOAD.xml. Ova datoteka se nalazi na lokaciji koja je definisana u konfiguracionoj datoteci servisne aplikacije App.config.
Kada servisnoj aplikaciji stigne CSV datoteka („send“ naredba), servis izvrši parsiranje datoteke i kreiranje kolekcije objekata klase ***Load***. Podaci za svaki sat iz CSV datoteke reprezentovani su jednim ***Load*** objektom. Zatim se ovim podacima ažurira baza podataka.
Ako se za neki sat uoči greška, kreira se novi *Audit* objekat. Audit objekti se upisuju u bazu podataka i prosleđuju se klijentskoj aplikaciji u okviru rezultata. Tekstovi grešaka iz Audit objekata ispisuju se na konzoli klijentske aplikacije. Greška se može desiti ako je podatak nevalidan (na primer nevalidno float ili DateTime polje), ako nedostaju neki podaci ili ako postoji greška u strukturi CSV datoteke.
Kada servisnoj aplikaciju pristigne zahtev za proračunatim podacima („get“ naredba), pristupa se izvršavanju događaja proračuna (Event). Koje metode će događaj proračuna izvršiti zavisi od pristiglih opcija (parametar „min“ za minimalnu potrošnju, „max“ za maksimalnu potrošnju i „stand“ za standardnu devijaciju).
Metod koji izračunava minimalnu potrošnju čita podatke iz baze podataka za trenutni dan i vraća minimalnu potrošnju.
Metod koji izračunava maksimalnu potrošnju čita podatke iz baze podataka za trenutni dan i vraća maksimalnu potrošnju.
Metod koji izračunava standardnu devijaciju čita podatke iz baze podataka za trenutni dan i vraća vrednost standardne devijacije. Za formulu standardne devijacije konsultovati Internet.
Kada su izvršeni događaji proračuna, kreira se tekstualna datoteka (.txt) koja se prosleđuje klijentskoj aplikaciji.

Kada je klijentska aplikacija primila tekstualnu datoteku, datoteka se čuva na lokaciji koja je definisana u konfiguracionoj datoteci klijentske aplikacije App.config. Naziv datoteke se kreira na osnovu trenutnog datuma i vremena. Takođe se na konzoli ispisuje poruka o imenu i putanji kreirane tekstualne datoteke.

3. Model podataka

Model podataka obuhvata sledeće klase:

- **Load** (polja: Id, Timestamp, MeasuredValue)
- **Audit** (Id, Timestamp, MessageType, Message)
  MessageType može da ima vrednosti Info, Warning i Error

4. Implementacija baze podataka

Baza podataka treba da bude implementirana kao XML.
XML baza podataka sadrži XML datoteke u koje se upisuju podaci. Svaka tabela je implementirana kroz jednu XML datoteku. Ukoliko XML datoteka ne postoji, potrebno je da bude kreirana automatski. Primeri XML tabela nalaze se u prilogu.

5. Tehnički i implementacioni zahtevi

1. Aplikacija treba da bude u višeslojnoj arhitekturi. Aplikacija treba da sadrži najmanje sledeće komponente:
   - baza podataka (XML baza podataka)
   - servisni sloj
   - korisnički interfejs – konzolna aplikacija
   - Common – projekat koji je zajednički za sve slojeve

2. Komunikacija između klijentske aplikacije i servisa obavlja se putem WCF-a

3. Rad sa datotekama treba da bude implementiran tako da se vodi računa o održavanju memorije, korišćenjem Dispose paterna
4. Događaj proračuna izvršava se korićenjem Event-a i Delegate-a. Delegati treba da pokazuju na odgovarajuće metode proračuna (proračuni minimalne potrošnje, maksimalne potrošnje i standardne devijacije).
5. Za aplikaciju treba da postoje sledeći dokumenti:
   - User manual
   - Dokument u kom je opisana arhitektura aplikacije