

DOKUMENTACIJA PROJEKTOG ZADATKA

KREIRANJE CSV DATOTEKA NA OSNOVU XML BAZE PODATAKA

(Zadatak 2)

VIRTUELIZACIJA PROCESA

Članovi tima:

Jovan Vukosavljević PR 79-2020

Nikola Lazarević PR 82-2020

Vanja Mihajlović PR 74-2020

Dajana Radović PR 75-2020

Profesor:

Bojan Jelačić

Asistenti:

Zoran Janković

Zoran Pajić

Zorana Babić

Sadržaj

Sadržaj.....	2
Uvod	3
Korišćene tehnologije.....	3
Opis interfejsa i osnovnih funkcionalnosti	3
Dijagram	9
Zaključak.....	10

Uvod

Aplikacija treba da se bavi kreiranjem CSV datoteka sa podacima o prognozama i ostvarenjima električne energije, na osnovu prosleđene XML datoteke.

Aplikacija se sastoji od servera i klijentske aplikacije koje komuniciraju putem WCF-a. Klijentska aplikacija ima definisanu putanju direktorijuma koji se osluškuje u konfiguracionoj datoteci. XML datoteka sadrži podatke o prognoziranoj i ostvarenoj potrošnji po satima. Ukoliko se desila promena na definisanoj putanji klijent šalje datoteku serveru.

Korišćene tehnologije

- .NET Framework
- Windows Communication Foundation(WCF)

Opis interfejsa i osnovnih funkcionalnosti

1. Interfejs servera:

- Implementira IData, IAuditingDB za In-Memory bazu podataka

```
public interface IAuditingDB
{
    2 references | 0 changes | 0 authors, 0 changes
    bool InsertLoad(Load ld);
    1 reference | 0 changes | 0 authors, 0 changes
    Dictionary<int, Load> GetLoads();

    8 references | 0 changes | 0 authors, 0 changes
    bool InsertAudit(Audit ad);
    1 reference | 0 changes | 0 authors, 0 changes
    Dictionary<int, Audit> GetAudits();

    2 references | 0 changes | 0 authors, 0 changes
    bool InsertImportedFile(ImportedFile ld);
    1 reference | 0 changes | 0 authors, 0 changes
    Dictionary<int, ImportedFile> GetImportedFiles();
}
```

Slika 1. IAuditing interfejs

```
public interface IData
{
    2 references | jovan12329, 5 days ago | 1 author, 1 change
    bool InsertFile(string fileName, byte[] fileData);
    2 references | jovan12329, 5 days ago | 1 author, 1 change
    Dictionary<string, byte[]> GetFileData(string fileBeginsWith);
}
```

Slika 2. IData interfejs

IAuditingDB interfejs sadrži metode za čuvanje i pribavljanje elemenata za auditing. In-Memory baza podataka je implementirana kao ConcurrentDictionary.

IData interfejs sadrži metode za čuvanje i pribavljanje sačuvanih datoteka unutar In-Memory baze podataka.

- Implementira ISendFile koji se koristi za komunikaciju

```
[ServiceContract]
6 references | jovan12329, 5 days ago | 1 author, 1 change
public interface ISendFile
{
    [OperationContract]
    2 references | 0 changes | 0 authors, 0 changes
    CSVFileResult Send(FileMemOptions fs);
}
```

Slika 3. ISendFile interfejs

ISendFile sadrži jednu metodu prilikom koje se prosleđuje datoteka koju klijent šalje na obradu.

2. Klijentska aplikacija:

- U App.config datoteci je definisana putanja direktorijuma koji se osluškuje

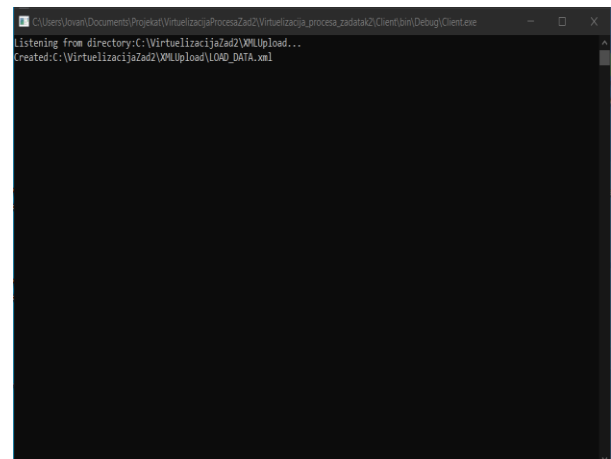
```
App.config X
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
  </startup>
  <appSettings>
    <add key="uploadPath" value="C:\VirtuelizacijaZad2\XMLUpload"/>
    <add key="downloadPath" value="C:\VirtuelizacijaZad2\CSVDownload"/>
  </appSettings>
  <system.serviceModel>
    <client>
      <endpoint name="ServiceName"
        address="net.tcp://localhost:4000/ISendFile"
        binding="netTcpBinding"
        contract="Common.Commands.ISendFile"/>
    </client>
  </system.serviceModel>
</configuration>
```

Slika 4. Klijentska App.config datoteka

- Kada se desila neka promena u direktorijumu, klijent šalje učitanoj datoteku serveru na obradu

```
public class DirListening
{
    private FileSystemWatcher sysDir;
    private ISendFile sender;
    private bool isCreated = false;
    private string pathConfig;
    private string download;
    1 reference | 0 changes | 0 authors, 0 changes
    public DirListening(ISendFile proxy, string path, string download)
    {
        this.download = download;
        this.pathConfig = path;
        this.sender = proxy;
        this.sysDir = new FileSystemWatcher(path);
        this.sysDir.NotifyFilter = NotifyFilters.Attributes
        | NotifyFilters.CreationTime
        | NotifyFilters.DirectoryName
        | NotifyFilters.FileName
        | NotifyFilters.LastAccess
        | NotifyFilters.LastWrite
        | NotifyFilters.Security
        | NotifyFilters.Size;

        this.sysDir.Filter = "*.xml";
        this.sysDir.IncludeSubdirectories = false;
    }
}
```



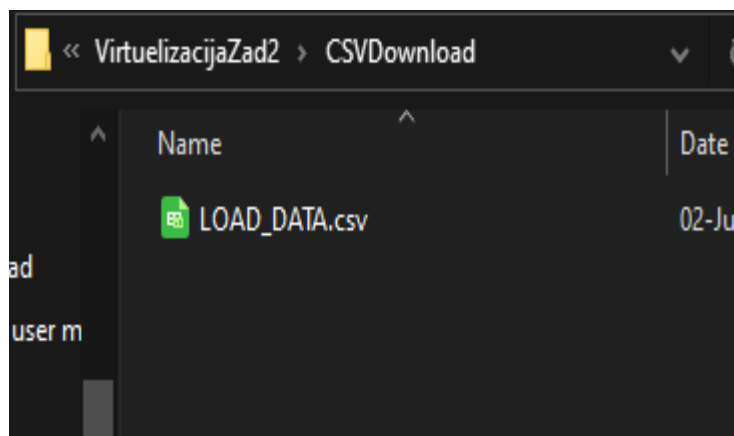
```
C:\Users\jovan\Documents\Projekt\Virtualizacija\Process\Zad2\Virtualizacija_process_zadatak2\Client\bin\Debug\Client.exe
Listening from directory: C:\VirtualizacijaZad2\XMLUpload...
Created: C:\VirtualizacijaZad2\XMLUpload\LOAD_DATA.xml
```

Slika 5. Klasa DirListening u kojoj je implementirano osluškivanje direktorijuma

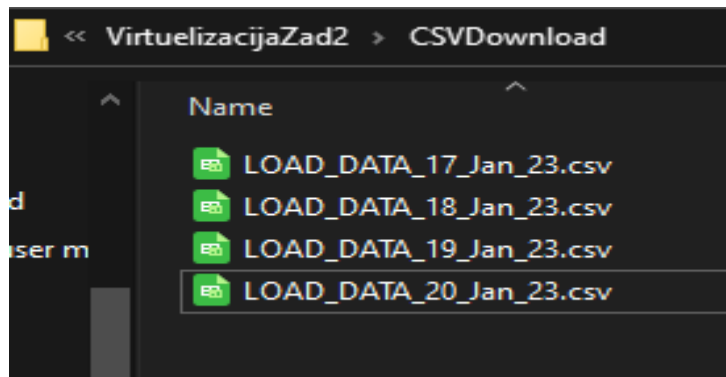
Slika 6. Primer izvršene klase

Osluškivanje je realizovano preko FileSystemWatcher klase kojoj se kao notifier prosleđuju metode OnChanged i OnCreated.

- Klijent će kao povratnu informaciju dobiti CSV datoteke koje je server vratio nakon obrade poslate XML datoteke

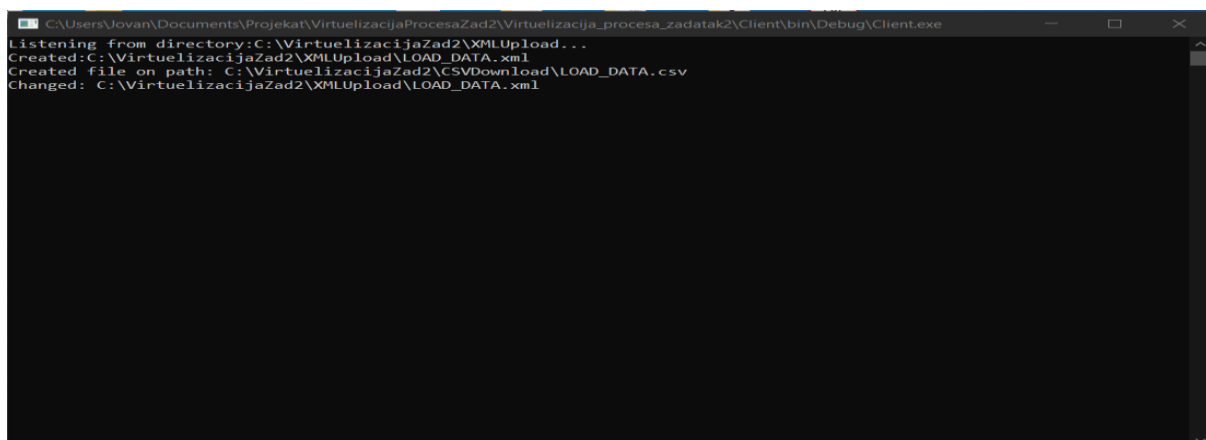


Slika 7. Primer jedne .csv datoteke



Slika 8. Primer više .csv datoteka

- Klijent na konzoli ispisuje putanje datoteka



Slika 9. Ispis putanje na koju je kreirana/e .csv datoteka/e

3. Serverska aplikacija:

- Primanje i smeštanje datoteke u In-Memory bazu podataka

```

public class InMemoryDataBase : IData, IAuditingDB
{
    1 reference | jovan12329, 5 days ago | 1 author, 1 change
    private InMemoryDataBase() { }

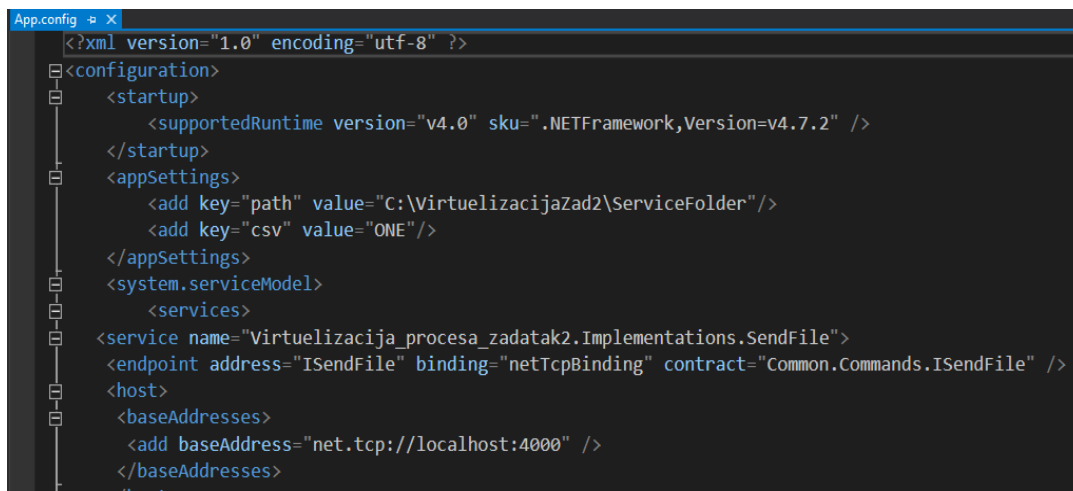
    private static readonly Lazy<InMemoryDataBase> lazyInstance
    {
        return new InMemoryDataBase();
    };

    11 references | jovan12329, 5 days ago | 1 author, 1 change
    public static InMemoryDataBase Instance
    {
        get
        {
            return lazyInstance.Value;
        }
    }

    //Keeps files in DB
    private ConcurrentDictionary<string, byte[]> ddb = new ConcurrentDictionary<string, byte[]>();
    //Keeps file records in DB
    private ConcurrentDictionary<int, Load> lddb = new ConcurrentDictionary<int, Load>();
    private ConcurrentDictionary<int, Audit> audB = new ConcurrentDictionary<int, Audit>();
    private ConcurrentDictionary<int, ImportedFile> imdB = new ConcurrentDictionary<int, ImportedFile>();
}
  
```

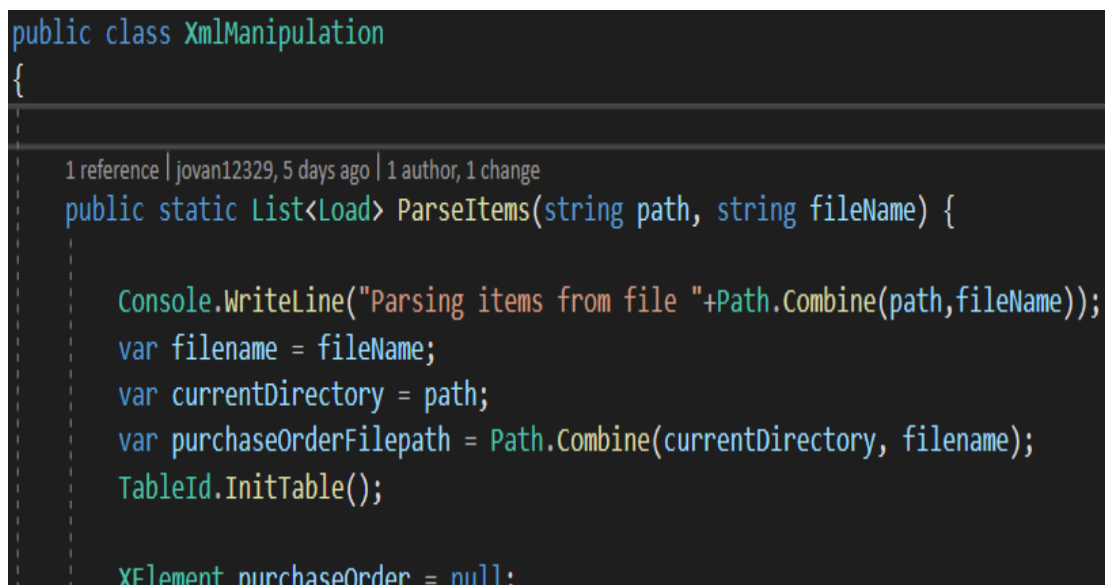
Slika 10. Implementacija In-Memory baze podataka

- Smeštanje primljene datoteke u direktorijum definisan u App.config datoteci



Slika 11. Serverska App.config datoteka

- Čitanje primljene XML datoteke i pravljenje novih objekata(Load, Audit, ImportedFile)



Slika 12. Implementacija čitanja .xml datoteke

Unutar klase XmlManipulation je implementirana statička metoda koja se poziva prilikom parsiranja prispele .xml datoteke. Unutar ove metode se vrši pravljenje objekata(Load, Audit, ImportedFile) kao i njihovo dodavanje u In-Memory bazu podataka.

- Upisivanje objekata u In-Memory bazu podataka

```
//Auditing DBManipulation
2 references | 0 changes | 0 authors, 0 changes
public bool InsertLoad(Load ld)
{
    return lddb.TryAdd(ld.Id, ld);
}
8 references | 0 changes | 0 authors, 0 changes
public bool InsertAudit(Audit ad)
{
    return audB.TryAdd(ad.Id, ad);
}
2 references | 0 changes | 0 authors, 0 changes
public bool InsertImportedFile(ImportedFile ld)
{
    return imdB.TryAdd(ld.Id, ld);
}
```

Slika 13. Metode koje se koriste za upisivanje elemenata u In-Memory bazu podataka

- Nakon poslednjeg Load objekta aktivira se Event koji kreira jednu ili više CSV datoteka u zavisnosti od zadate vrednosti u App.config datoteci(vidi sliku 11.)

```
1 reference | 0 changes | 0 authors, 0 changes
public void LastLoad(object source, PubEventArgs e)
{
    var howMany = ConfigurationManager.AppSettings["csv"];

    if (howMany.Equals("ONE"))
    {
        CreateOneCSV(e);
    }
    else
    {
        CreateMoreCSV(e);
    }
}
```

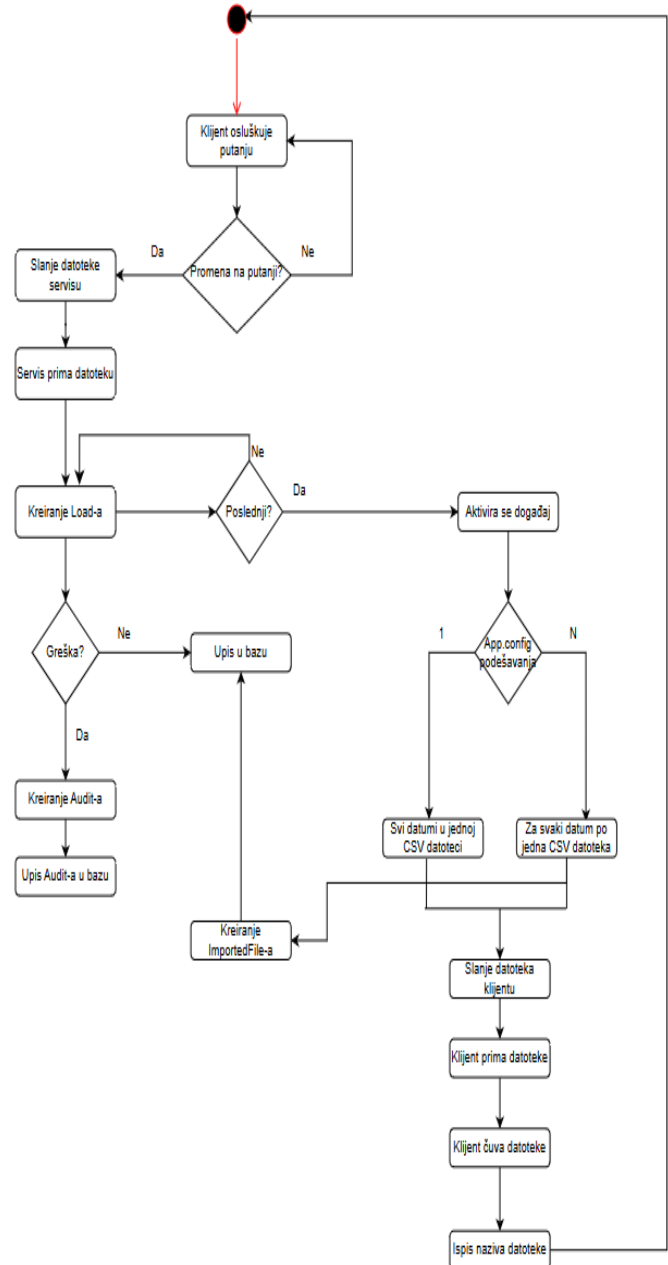
Slika 14. Notifier metoda

Event je zasnovan na observable pattern-u , imamo klasu koja je publisher i klasu koja je subscriber.

- Smeštanje kreiranih CSV datoteka u CSVFileResult
- Prosledjivanje CSVFileResult-a klijentu

Dijagram

Load	ImportedFile	Audit
Id - int	Id - int	Id - int
TimeStamp - DateTime	FileName - string	TimeStamp - DateTime
ForecastValue - double		MessageType - Enum (Info, Warning, Error)
MeasuredValue - double		Message - string



Zaključak

Razvijena aplikacija za generisanje CSV datoteka sa podacima o prognozama i ostvarenjima potrošnje električne energije na osnovu XML datoteke pruža osnovne funkcionalnosti za potrebe kompanije za prenos električne energije.

Mogući pravci budućeg istraživanja i proširenja zadatka mogu uključivati:

Smanjenje opterećenja radne memorije: Umesto upisivanja čitave datoteke u radnu memoriju, uzimati po segment i obraditi ga. Nakon obrade segment se briše iz radne memorije i uzima se naredni. Nastaviti dok se svi segmenti ne obrade.

Implementacija sigurnosnih mehanizama: Dodavanje sigurnosnih mehanizama zaštite podataka, kao što su autentifikacija i autorizacija, kako bi se osigurala bezbednost podataka i sprečile neovlašćene pristupe.

Otpornost na otkaze: Ukoliko želimo da server radi konstantno, neophodno je imati replikator koji će odgovarati klijentu na zahteve ukoliko server otkaže. Takođe, potrebno je imati repliku podataka kako bi se sačuvala konzistentnost i kako bi mogli da vratimo server kakav je bio na poslednjem checkpoint-u.

Proširenje zadatka u ovim pravcima bi unapredilo funkcionalnosti aplikacije i omogućilo bolje iskorišćenje podataka o potrošnji električne energije u svrhe planiranja i optimizacije.