Alat za grafičku prezentaciju CDL datoteka

Jukić, Magdalena

Master's thesis / Diplomski rad

2023

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:625477

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-05-14**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni studij

ALAT ZA GRAFIČKU PREZENTACIJU CDL DATOTEKA

Diplomski rad

Magdalena Jukić

Osijek, 2023.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac D1: Obrazac za imenovanje Povjerenstva za diplomski ispit

Osijek, 19.09.2023.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za diplomski ispit

Ime i prezime Pristupnika:	Magdalena Jukić
Studij, smjer:	Diplomski sveučilišni studij Računarstvo
Mat. br. Pristupnika, godina upisa:	D-1207R, 07.10.2021.
OIB studenta:	41505611896
Mentor:	prof. dr. sc. Dominika Crnjac Milić
Sumentor:	izv. prof. dr. sc. Alfonzo Baumgartner
Sumentor iz tvrtke:	Damir Birovljević
Predsjednik Povjerenstva:	prof. dr. sc. Krešimir Nenadić
Član Povjerenstva 1:	prof. dr. sc. Dominika Crnjac-Milić
Član Povjerenstva 2:	izv. prof. dr. sc. Zdravko Krpić
Naslov diplomskog rada:	Alat za grafičku prezentaciju CDL datoteka
Znanstvena grana diplomskog rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Zadatak diplomskog rada:	Potrebno je napraviti program za konverziju CDL (Command Desription Language) datoteka u reprezentacijski format XML ili HTML koji će u skladu sa zadanom stilizacijom prikazivati podatke u obliku odgovarajućih tablica. Primjeri ulaznih podataka i odgovarajućih rezultata bit će dobiveni od strane poduzeća u suradnji. U radu je potrebno opisati tehnologije kojima će zadatak biti ispunjen, ali i primjenu rješenja u poslovnoj praksi. Mentorica (FERIT): Prof.dr.sc. Dominika Crnjac Milić Sumentor (FERIT): Izv.prof.dr.sc. Alfonzo Baumgartner Sumentor (Enea Software d.o.o.): Damir Birovljević
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 2 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene od strane mentora:	19.09.2023.

Potvrda mentora o predaji konačne verzije rada:	*Mentor elektronički potpisao predaju konačne verzije.*
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 13.10.2023.

Ime i prezime studenta:	Magdalena Jukić
Studij:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D-1207R, 07.10.2021.
Turnitin podudaranje [%]:	2

Ovom izjavom izjavljujem da je rad pod nazivom**: Alat za grafičku prezentaciju CDL datoteka**

izrađen pod vodstvom mentora prof. dr. sc. Dominika Crnjac Milić

i sumentora izv. prof. dr. sc. Alfonzo Baumgartner

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD 1

2. PRIKAZ DOSADAŠNJIH RJEŠENJA 2

2.1. Alati za vizualizaciju podataka 2

2.1.1. Besplatni alati za vizualizaciju podataka 2

2.1.2. Komercijalni alati za vizualizaciju podataka 2

2.2. Alati za pretvorbu podataka 3

3. KORIŠTENI ALATI I TEHNOLOGIJE 5

3.1. HTML 5

3.2. CSS 6

3.3. JavaScript 8

3.4. PHP 9

3.5. XAMPP 10

3.6. JSON 10

3.7. XML 11

3.8. Python 12

3.9. Visual Studio Code 13

4. PRIPREMA PODATAKA 14

4.1. CDL datoteke 14

4.2. Učitavanje CDL datoteka 14

4.3. Rječnik 18

4.4. Generiranje HTML, JSON i XML datoteka 20

4.5. Prikaz početne stranice i tablica 22

5. KORIŠTENJE I PRIKAZ 24

5.1. Raspored datoteka i pokretanje 24

5.2. Funkcionalnosti 25

5.3. Primjer CDL datoteke 28

6. ZAKLJUČAK 31

LITERATURA **32**

SAŽETAK **35**

ABSTRACT **36**

ŽIVOTOPIS **37**

PRLOZI **38**

1. UVOD

U današnje vrijeme veliki broj informacija kruži velikom brzinom. To predstavlja izazov za tvrtke koje svoje odluke donose na temelju raznih podataka. Budući da se radi o neobrađenim podacima, potrebno ih je prikupiti i pravilno obraditi. Najvažnije je uspješno odrediti ključne informacije iz podataka. Međutim, dolazi do problema jer se radi o velikim podacima koje je teško analizirati, ali i razumjeti koje informacije ti podaci nose. Da bi se došlo do ispravnog tumačenja tih podataka, postoje razni načini njihovog prikupljanja i analiziranja. U svrhu toga koriste se različite vizualizacije. Vizualizacije uvelike pomažu upravo razumijevanju i shvaćanju koje su to važne informacije koje podaci donose te na temelju toga tvrtke kasnije mogu donijeti ispravne odluke.

Prema [1], vizualizacija podataka je predstavljanje podataka i informacija kao grafički prikaz. Svrha vizualizacije podataka je da grafičkim prikazima te kreativnim i dinamičkim odnosom prema korisnicima privuče njihovu pažnju kako bi oni jasnije i lakše shvatili prenesene informacije.

Postoje različite vizualizacije podataka. Ovisno o tipu podataka potrebno je odabrati ispravne metode i alate kojima se služi za vizualizaciju kako ne bi došlo do gubljenja ključnih informacija, ali i da se izbjegne opterećivanje nepotrebnim informacijama.

Danas se razvijaju brojni alati za što bolju i jednostavniju prezentaciju podataka s ciljem da se olakša tvrtkama analiza i sinteza podataka. Svrha ovog diplomskog rada je opis kreiranja i opis funkcionalnosti alata koji je napravljen kako bi olakšao analizu poslovnih procesa. Na temelju CDL (*Command Description Language)* datoteke bit će generirana HTML (*HyperText Markup Language)* datoteka u kojoj su informacije prikazane u tablici. Taj alat krajnjim korisnicima olakšava snalaženje i pregledavanje vrijednosti parametara.

U drugom poglavlju je pregled dosadašnjih alata koji služe za vizualizaciju i pretvorbu podataka. U trećem poglavlju su opisane korištene tehnologije i alati za izradu rješenja. U četvrtom će se prikazati postupak izrade alata. Peto poglavlje će sadržavati uvid u alat i opis načina korištenja. Naposljetku, u šestom poglavlju iznijeti su zaključci diplomskog rada i dane su preporuke za daljnje istraživanje i poboljšanje rješenja.

2. PRIKAZ DOSADAŠNJIH RJEŠENJA

Unutar ovog poglavlja predstavit će se alati koji se danas koriste za vizualizaciju podataka. Također, bit će navedeni i alati koji se koriste za pretvorbu podataka iz jednog tipa u drugi.

2.1. Alati za vizualizaciju podataka

Velika važnost se pridaje vizualizaciji podataka, posebno u poslovnom svijetu. Na temelju primljenih podataka potrebno je uočiti najvažnije informacije i predstaviti ih tako da svima, kojima su te informacije bitne, bude jasno što one predstavljaju. Vizualizacija podataka se može olakšati korištenjem različitih alata. Neki su alati dostupni svima na korištenje i besplatni su, dok ima i alata koji su komercijalni.

2.1.1. Besplatni alati za vizualizaciju podataka

Prema [2], besplatni alati su kvalitetni i dobro razvijeni bez obzira na to što su lako dostupni. Oni se najčešće koriste za obrazovanje, ali i za osobne potrebe. Oni imaju raznovrsnu primjenu, od alata za prezentaciju, matematičkih softvera pa do različitih programskih jezika, ali svi imaju svrhu vizualizacije. Tehnologije koje se često znaju koristiti za vizualizaciju podataka su HTML, CSS (*Cascading Style Sheets)*, JavaScript, PHP (*Hypertext Preprocessor*) i programski jezik kao što je Python. Uz navedene tehnologije moguće je koristiti i biblioteke kao što su *Chart.js* ili *d3.js* za prikaz različitih vizualizacija.

Referirajući se na [3], jedan od poslovnih alata koji je dobar za male organizacije je *GnuCash*. On se koristi za vođenje financija unutar organizacije, ali prikladan je i za osobne potrebe. Neke od glavnih značajki ovog alata su: usklađivanje transakcija, stvaranje rasporeda za transakcije i usklađivanje transakcija.

2.1.2. Komercijalni alati za vizualizaciju podataka

Prema [4], na današnjem tržištu ima mnogo komercijalnih alata koji olakšavaju rad s podacima. Tvrtkama je bitno da odaberu najbolje alate za obradu i vizualizaciju podataka kako bi mogli donijeti ispravne poslovne odluke.

Osvrćući se na [5], Microsoft Excel je najpoznatiji alat za vizualizaciju i obradu velikog skupa podataka. On omogućuje stvaranje grafikona i dijagrama na temelju podataka. Uz to, pruža opcije kao što su: oblikovanje, vizualno isticanje podataka i stvaranje tablica za lakše razumijevanje podataka.

Prema [6], Zoho Analytics je također jedan od takvih alata. On omogućuje: stvaranje grafova i tablica na temelju predanih podataka koji mogu biti s lokalnog diska, internetske stranice ili čak podaci spremljeni u *cloud drive*.

Prema [7], alat koji pomaže u poslovnoj inteligenciji je Tableau. On olakšava analizu podataka velikim tvrtkama koje rade sa složenim podacima i omogućuje stvaranje tablica, grafova, karti, kontrolne ploče i infografika. Također, Tableau pruža i naprednije vizualizacije podataka, a one su: oblak mjehurići, histogrami, vremenske crte, gantogram i slično.

Referirajući se na [8], Sisense je još jedan od alata za vizualizaciju podataka koji se koristi u poslovnoj inteligenciji. Ovaj alat pruža mogućnost pametnog izračuna i stvaranje grafikona na temelju specifičnog skupa podataka. Korisnici tako mogu stvoriti jedinstvene vizualizacije koje su prikladne za analizu njihovih podataka.

Ostvareno rješenje u ovom diplomskom radu se razlikuje od drugih alata na način da za ulazne podatke prihvaća samo CDL datoteke koje su napisane na svojstven način. Specifično je po tome što pretvara navedene datoteke u HTML oblik koji je čitljiviji i jasniji i omogućuje korisniku lakše razumijevanje podataka iz CDL datoteke.

2.2. Alati za pretvorbu podataka

Prema [9], pretvorba podataka je važan proces u analizi podataka. To je proces u kojemu se podaci pretvaraju iz jednog oblika u drugi zbog lakše analize. Pretvorba se može ostvariti korištenjem već gotovih alata. Hevo Data je alat koji omogućuje različite manipulacije s podacima. Pretvorbe podataka koje korisnici mogu koristiti su: sanitizacija i sažimanje tablica i stupaca te raščlanjivanje podataka zapisanih u JSON formatu. Data Building Tool (DBT) je alata koji omogućuje korisnicima pretvorbu i pripremu podataka. Pomoću ovog alata mogu raditi s neobrađenim podacima, mogu ih razvijati, rasporediti ili testirati. Za korištenje ovog alata potrebno je poznavati SQL i Python. Alteryx platforma također omogućuje rad s neobrađenim podacima. Osim toga, nudi pojedinačne alate za postavljanje vrste podataka, čišćenje vrijednosti, kodiranje podataka i slično.

Osvrćući se na [10], PDFmyURL je alat koji omogućuje pretvorbu internetskih stranica u PDF. Prednosti korištenja ovog alata su: pretvorba cijele internetske stranice odjednom, slanje obavijesti nakon što se stvori PDF odabrane internetske stranice i odabiri različitih opcija kao što su veličina stranice, orijentacija, zaglavlje, podnožje i slično.

Prema [11], jedan od poznatih alata za pretvorbu podataka je i VideoSolo. On omogućuje pretvorbu video datoteka u audio datoteke, ali i videozapise iz jednog formata u neki drugi. U slučaju da korisnik nije siguran koji mu format videozapisa treba, umjesto formata može odabrati uređaj i program će sam odrediti potrebni format. Također, ovaj alat je besplatan i jednostavan za korištenje.

3. KORIŠTENI ALATI I TEHNOLOGIJE

Ovo poglavlje sadrži opisane alate i tehnologije koje su korištene za izradu rješenja. Detaljno se opisuju označni jezici HTML te CSS koji se koriste za uređivanje prikaza HTML dokumenta. Nakon toga se definiraju skriptni jezik JavaScript te programski jezik PHP koji služe za ostvarivanje funkcionalnosti unutar HTML dokumenta. Zatim se predstavlja platforma XAMPP (*Cross-platform, Apache Server, MariaDB, PHP, Perl*) koji služi za pokretanje PHP dokumenta. Nakon toga, slijedi poglavlje gdje se opisuje JSON (*JavaScript Object Notation*) te programski jezik Python koji su poslužili za čitanje i zapisivanje datoteka u željenom obliku. U zadnjem poglavlju se opisuje integracijska okolina za uređivanje koda Visual Studio Code koji je služio za pisanje PHP datoteke i analizu generiranih HTML datoteka.

3.1. HTML

Prema [12], HTML je opisni jezik za izradu internetskih stranica. U nazivu HTML-a, *hypertext* označava poveznice koje povezuju internetske stranice, bilo to veze unutar jedne ili između dvije odvojene internetske stranice. Poveznice su osnove interneta. HTML koristi *markup* kako bi označio tekst, slike ili druge sadržaje za prikaz na pregledniku. Također, postoje i posebni elementi kao što su <*head*>, <*title*>, <*body*>, <*p*> itd. Ti elementi se zovu oznake. Ime elementa se nalazi unutar šiljastih zagrada, „<“ i „>“. Svaki element koji je otvoren treba imati svoj par koji zatvara sadržaj unutar tog elementa.

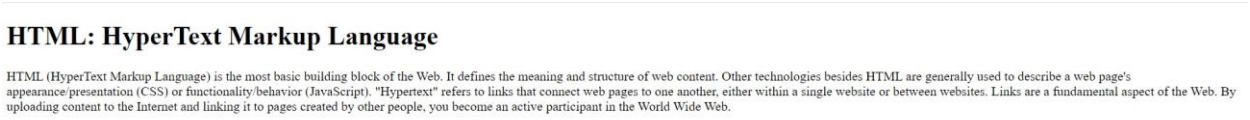
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Home</title>
</head>
<body>
    <h1>HTML: HyperText Markup Language</h1>
    <p>
        HTML (HyperText Markup Language) is the most basic
        building block of the Web. It defines the meaning and
        structure of web content. Other technologies besides HTML
        are generally used to describe a web page's appearance/presentation
        (CSS) or functionality/behavior (JavaScript).

        "Hypertext" refers to links that connect web pages to one
        another, either within a single website or between websites.
        Links are a fundamental aspect of the Web. By uploading content to the
        Internet and linking it to pages created by other people, you become an
        active participant in the World Wide Web.
    </p>
</body>
</html>
```

Sl. 3.1. Primjer HTML koda

Na slici 3.1. prikazan je jednostavni HTML kod. On se sastoji od tri važna dijela: <*html*>, <*head*> i <*body*>. Referirajući se na [13], HTML element <*head*> sadrži strojno čitljive informacije, metapodatke i informacije o dokumentu kao što su: njegov naslov, skripte i stil (linije 4 – 6).

Prema [14], element <*body*> predstavlja sadržaj HTML dokumenta i može postojati samo jedan takav element unutar <*html*> elementa. Na internetskom pregledniku se prikazuje samo onaj sadržaj koji se nalazi između oznaka <*body*> i <*/body*>. Slika 3.2. predstavlja stranicu koja se stvara na temelju koda prikazanog na slici 3.1.



Sl. 3.2. Primjer HTML stranice

Cilj korištenja HTML-a je da se omogući prikaz tablica i podataka unutar tablica. Nadalje, HTML je potreban kako bi se prikazao sav tekst povezan uz tablice te za izradu poveznica između stranica. U poglavlju 4.3. je prikazana svrha korištenja HTML-a.

3.2. CSS

Prema [15], CSS je stilski jezik koji se koristi za opis dokumenta pisanih u HTML-u ili XML-u (*eXtensible Markup Language*). CSS opisuje način prikaza elemenata odnosno kako trebaju izgledati. U ovom diplomskom radu, CSS se koristio kako bi se tablice s podacima prikazale što pristupačnije korisnicima. Koristile su se prilagođene boje kako bi se istaknule pojedine ćelije u

tablicama. Također, tekst u tablicama je stiliziran i prilagođena je veličina slova. CSS služi i za dinamički prikaz tablica što je prikazano u poglavlju 4.3.

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Home</title>
7      <style>
8          h1{
9              background-color: rgb(119, 255, 0);
10             text-align: center;
11             font-family: 'Courier New', Courier, monospace;
12         }
13     </style>
14
15     <link rel="stylesheet" href="styles.css">
16 </head>
17 <body>
18     <h1>HTML: HyperText Markup Language</h1>
19     <p style="background-color: rgb(253, 250, 183);">
20         HTML (HyperText Markup Language) is the most basic
21         building block of the Web. It defines the meaning and
22         structure of web content. Other technologies besides HTML
23         are generally used to describe a web page's appearance/presentation
24         (CSS) or functionality/behavior (JavaScript).
25
26         "Hypertext" refers to links that connect web pages to one
27         another, either within a single website or between websites.
28         Links are a fundamental aspect of the Web. By uploading content to the
29         Internet and linking it to pages created by other people, you become an
30         active participant in the World Wide Web.
31     </p>
32
33     <div>This is for external CSS code</div>
34 </body>
35 </html>
```

Sl. 3.3. Primjer CSS koda koji je pisan direktno u HTML dokument

Prema [16], CSS se može pisati na tri načina: u redu, interno i eksterno. Na slici 3.3. je primjer pisanja stila u redu, odnosno unutar oznake (linija 19). Unutar oznake <*p*> naznačeno je stiliziranje u redu ključnom riječi *style*, te je naveden atribut koji se želi stilizirati. U ovom slučaju je to boja pozadine elementa <*p*>. Za interno ubacivanje dizajna na stranicu potrebno je unutar oznake <*head*> dodati element <*style*> te napisati elemente koje se želi preoblikovati. Primjer za takav način ubacivanja CSS koda je prikazan na slici 3.3. (linije 7 – 13). Treći način je eksterno ubacivanje CSS dokumenta koji se radi tako da se unutar elementa <*head*> doda element <*link*> koji će povezati HTML dokument sa CSS dokumentom, prikazano na slici 3.3. (linija 15). Eksterni CSS dokument je prikazan na slici 3.4.

```
1  div {
2      background-color: powderblue;
3    }
4    p {
5      color: red;
6    }
```

Sl. 3.4. Primjer CSS koda koji je pisan u poseban dokument, *styles.css*

Slika 3.5. prikazuje kako izgleda HTML dokument sa slike 3.2. nakon što se primjeni CSS stilizacija. Ovo je primjer stilizacije elemenata i teksta unutar njih, ali CSS može poslužiti i za neke veće izmjene. To je prikazano u poglavlju 4.3.

Sl. 3.5. Primjer HTML dokument nakon primjenje CSS koda

3.3. JavaScript

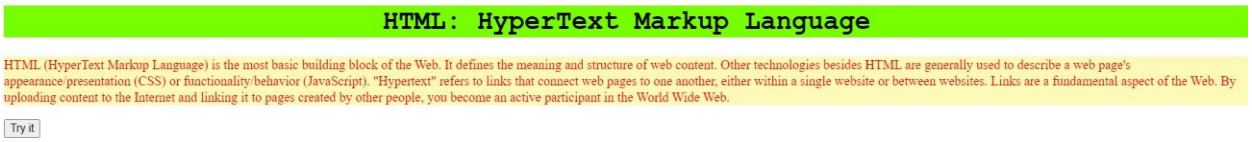
Prema [17], JavaScript je programski jezik koji se koristi i na strani klijenta i na strani poslužitelja te omogućuje da internetske stranice budu interaktivne. To uvelike privlači korisnike te poboljšava njihovo korisničko iskustvo. Može se reći da JavaScript daje ponašanje internetskim stanicama.

Osvrćući se na [18], najčešće se koristi kada je potrebno prikazati interaktivne karte, animacije koje su 2D ili 3D grafike, videozapise ili druge multimedijske sadržaje. Uz HTML i CSS to je jedan od standardnih *web* tehnologija.

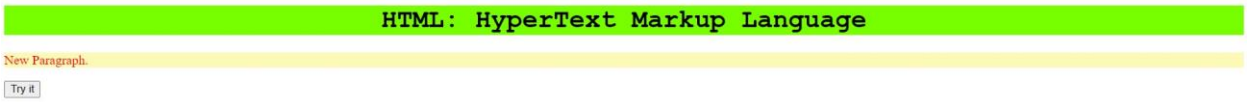
```
33    <button type="button" onclick="myFunction()">Try it</button>
34    <script>
35        function myFunction() {
36          document.getElementById("demo").innerHTML = "New Paragraph.";
37        }
38    </script>
```

Sl. 3.6. Primjer JavaScript koda

Kod se piše unutar elementa <*body*> u kojem je potrebno otvoriti <*script*> element gdje se mogu pisati dinamičke funkcionalnosti stranice. Slika 3.6. prikazuje primjer JavaScript koda. Definiran je gumb koji sadrži funkcionalnost (linija 33). Klikom na njega aktivira se funkcija *myFunction* (linije 35 – 36) koja sav tekst unutar određenog paragrafa zamjenjuje s novim tekstom.



Sl. 3.7. Primjer funkcionalnosti JavaScripta prije klika na gumb „Try it“



Sl. 3.8. Primjer funkcionalnosti JavaScripta poslije klika na gumb „Try it“

Na slikama 3.7. i 3.8. se može zamijetiti kako klikom na gumb „*Try it*“ tekst napisan unutar narančasto obojanog paragrafa se zamjenjuje novim tekstom.

U ovom diplomskom radu je potrebno koristiti JavaScript za implementiranje dinamičke funkcionalnosti generiranoj tablici, odnosno za funkcionalnost proširivanja i smanjivanja tablice.

3.4. PHP

Prema [19], PHP je skriptni programski jezik koji je otvorenog koda (engl. *open-source code*). PHP dokument, čija je ekstenzija *.php*, može sadržavati tekst, HTML, CSS, JavaScript i PHP kod. PHP se koristi uglavnom za: generiranje dinamičkog sadržaja internetske stranice, za otvaranje, čitanje, pisanje, brisanje i zatvaranje datoteka na poslužitelju. Može se koristiti i za promjenu podataka unutar baze i slično. Isto tako može poslužiti na različitim platformama kao što su: Windows, Linux, Unix i Mac OS. Također, podržava širok raspon bazi podataka.

Referirajući se na [20], PHP je najkorišteniji poslužiteljski skriptni jezik za razvoj internetskih stranica. Kako bi se pokrenuo PHP dokument potreban je internetski poslužitelj. Neki od poznatih poslužitelja su XAMPP koji će se opisati u poglavlju 3.5.

Za potrebe ovog diplomskog rada koristi se PHP za čitanje dokumenata unutar mape te ispis na početnu stranicu. U poglavlju 4.4. je opis primjene PHP-a.

3.5. XAMPP

Referirajući se na [20], XAMPP (X - *Cross-platform,* A - Apache Server, M - MariaDB, P – PHP, P - Perl) je paket rješenja *web* poslužitelja otvorenog koda. Koristi se za testiranje internetskih aplikacija na lokalnom poslužitelju. XAMPP je korišten za pokretanje početna stranica napisana u PHP-u.

Prema [21] neke od prednosti korištenja XAMPP-a su: mogućnost korištenja na različitim operacijskim sustavima (Windows i Linux) i vrlo je jednostavan za korištenje. Sadrži kontrolni panel na kojem se lako može pokrenuti i zaustaviti pojedini mehanizmi, kao što je *Apache*. Također, uključuje i dodatne module: OpenSSL, phpMyAdmin, MediaWiki, Joomla i WordPress.

3.6. JSON

Osvrćući se na [22], JSON je najpopularniji format za prijenos podataka. Podatke može prenositi između dva računala, baze podataka ili programa. Nastao je kao potreba za komunikacijom između poslužitelja i preglednika u stvarnom vremenu. Douglas Crockford ga je specificirao i popularizirao početkom 2000. godine. Osnovna svojstva JSON-a su: lagani format prijenosa podataka zasnovan na tekstu, ekstenzija mu je *.json*, lako čitljiv i upotrebljiv, neovisan je o programskom jeziku. Prilikom stvaranja JSON objekta nužno se treba držati osnovnih pravila. Svaki JSON objekt započinje i završava vitičastim zagradama. Sastoje se od parova (ključ, vrijednost), a svaki par je odvojen zarezom. Ključ i vrijednost pisani su u navodnicima i odvojeni su dvotočkom. Vrijednosti mogu biti različiti tipovi podataka, na primjer: *string, integer, boolean*

i slično. Uređena zbirka podataka predstavlja JSON niz. On se definira unutar uglatih zagrada gdje su vrijednosti međusobno odvojene zarezom.

Sl. 3.9. Primjer JSON niza

Na slici 3.9. prikazan je JSON zapis gdje se mogu vidjeti svi spomenuti dijelovi JSON objekta. Za izradu diplomskog rada nužno je poznavanje strukture JSON objekta i JSON niza za olakšanu kontrolu ispisa podataka koje je potrebno zapisati u tablice.

3.7. XML

Prema [23], XML (eXtensible Markup Language) je označni jezik koji služi za spremanje i transport podataka. Podaci su zapisani između oznaka. Oznake nisu definirane unaprijed nego autor XML dokumenta sam određuje nazive oznaka. Osim oznaka, autor određuje strukturu dokumenta. XML pohranjuje podatke u obliku teksta. Također, XML olakšava proširivanje i nadogradnju na nove operacijske sustave, aplikacije ili preglednike bez gubitaka podataka. Čitanje podataka zapisanih u XML datoteke je jednostavno, jasno je i ljudima i računalima.

Slika 3.10. prikazuje podatke o osobama zapisane u XML oblik. Podatke se lako može pročitati i razumjeti koje su informacije zapisane. Također, ovaj zapis se može usporediti sa slikom 3.9. gdje su isti podaci zapisani u JSON. XML za razliku od JSON-a ima kompleksniji zapis podataka, ali je i dalje čitljiv i jasan.

```xml
<?xml version="1.0" ?>
<root>
    <person>
        <ime>Marko</ime>
        <prezime>Lukić</prezime>
        <id>2988394</id>
        <programskiJezici>
            <item>Java</item>
            <item>C#</item>
        </programskiJezici>
    </person>
    <person>
        <ime>Tiho</ime>
        <prezime>Majić</prezime>
        <id>2982444</id>
        <programskiJezici>
            <item>C</item>
            <item>C#</item>
            <item>Python</item>
        </programskiJezici>
    </person>
</root>
```

Sl. 3.10. Primjer XML datoteke

3.8. Python

Prema [24], Python je programski jezik koji se koristi za razvoj internetskih stranica, programske podrške, automatizaciju zadataka te analizu i vizualizaciju podataka.

Referirajući se na [25], Python je interpretirani, objektno orijentirani programski jezik visoke razine uz dinamičku semantiku. Razvio ga je Guido van Rossum te je objavljen 1991. godine. Python je jednostavan te prilagođen početnicima. Sintaksa sliči engleskom jeziku uz utjecaj matematike stoga je jednostavan za čitanje. Umjesto oznaka točka-zarez i zagrada koje se koriste kod izvršenja naredbi kod drugih programskih jezika, Python jednostavno koristi novu liniju. Omogućuje programerima pisanje koda u manje linija nego što bi inače bilo potrebno prilikom korištenja drugih programskih jezika. Uz to, korisnici mogu pokrenuti program sve do problematičnog dijela jer Python koristi *run-time* tip provjere. Također, podržan je od strane više platformi kao što su Windows, Mac, Linux, Raspberry Pi i dr.

Budući da Python ima dosta ugrađenih metoda kojima se lako može upravljati datotekama različitog tipa, u ovom diplomskom radu se koristi za čitanje datoteka, stvaranje njihovog novog zapisa te na kraju stvaranje HTML datoteka za prikaz željenih tablica.

Osvrćući se na [26], IDLE (*Integrated Development and Learning Environment*) je integrirano razvojno okruženje za Python. IDLE modul se već nalazi unutar Python instalacijskog programa za Windows. Kao i Python Shall, IDLE se može koristiti za izvršavanje jedne naredbe te za

stvaranje, modificiranje i izvršavanje Python skripti. Također, pruža uređivač teksta za stvaranje Python skripti uz značajke kao što su: isticanje sintakse, automatsko dovršavanje i pametno uvlačenje linija.

3.9. Visual Studio Code

Prema [27], Visual Studio Code je moćan uređivač koda. Mnogi razvojni inženjeri ga koriste zbog njegove dostupnosti i pristupačnosti. Dostupan je za Windows, macOS, Linux i Raspberry Pi OS. Također, ima ugrađenu podršku za JavaScript, TypeScript i Node.js. Visual Studio Code ima mogućnost proširenja za korištenje drugih programskih jezika (npr. C++, C#, Java, Python i PHP), okruženja (npr. Docker i Kubernetes) i oblaka (npr. Amazon Web Services, Microsoft Azure i Google Cloud Platform). Prednost korištenja Visual Studio Code-a je njegovo lako i brzo pokretanje. Također, ima ugrađen IntelliSense za dovršavanje koda, funkcija i uvezenih modula. Treba istaknuti i grafičko uklanjanje grešaka, savjeti za nazive parametara i navigacija koda te refaktoriranje kao neke od dodatnih prednosti.

Visual Studio Code, zbog svoje raznovrsne primjene, služi pri pisanju PHP datoteka i analiza generiranih tablica u obliku HTML datoteka.

4. PRIPREMA PODATAKA

U ovom poglavlju objašnjava se što je CDL datoteka te se detaljno objašnjava kod koji služi za pripremu podataka. Prikazan je primjer jedne CDL datoteke te osnovne funkcije koje se koriste za stvaranje novog zapisa datoteke. Nakon toga, objašnjeno je što je rječnik (engl. *dictionary*) i kako se podaci zapisuju u rječnik. Zatim, prikazan je i objašnjen dio koda koji služi za stvaranje HTML dokumenta. Prikazan je i kod koji služi za stvaranje početne stranice.

4.1. CDL datoteke

CDL datoteke sadrže podatke kojima se definira konfiguracija sustava. Podaci su zapisani kao parovi ključ-vrijednost te time CDL datoteke podsjećaju na druge slične formate datoteka, na primjer JSON datoteke. Ti podaci su važni za ljude koji se bave izgradnjom sustava telekomunikacije, ali i za njihove klijente. Produkti koji koriste CDL datoteke su: Enea Access Manager, Enea Policy Manager, ENEA Subscription Manager i Enea ENUM. Pojavila se potreba za stvaranjem alata koji bi podatke iz CDL datoteka prikazao na što jasniji način te omogućilo svima lakše snalaženje u brojnim podacima i analiziranje vrijednosti parametara. Budući da su podaci u CDL datotekama nepregledni te sadrže mnogo znakova koji su suvišni, potrebno je ukloniti sve suvišne znakove te pretvoriti CDL datoteke u drugi, lakše čitljiv format. Tablični prikaz podataka je najprikladniji s obzirom na raspored podataka u CDL datotekama. Kako bi se to ostvarilo korišten je HTML te se podaci zapisuju unutar HTML tablice. Tako umjesto čitanja iz datoteka klijenti imaju pred sobom tablice koje efektnije opisuju konfiguracije.

4.2. Učitavanje CDL datoteka

Kako bi se proces generiranja tablica, odnosno stvaranja HTML datoteka ubrzao, koriste se funkcije koje su omogućile da program samostalno obradi sve zadane procese na svim CDL datotekama. Potrebno je pročitati sve CDL datoteke iz mape. Korištena je funkcija *os.listdir(path)* kako bi se došlo do odabrane mape gdje su spremljene sve datoteke. Zatim, postojao je uvjet da datoteka mora imati ekstenziju *.cdl* kako bi se osiguralo da su odabrane samo željene datoteke. Datoteke se spremaju u listu koja je poslužila za automatizaciju procesa. Korištenjem *for*-petlje u novonastaloj listi CDL datoteka provodi se proces obrade svake datoteke zapisane u tu listu. Slijede objašnjenja funkcija koje su korištene za obradu CDL datoteka.

Za učitavanje datoteke korištena je ugrađena funkcija *open(file, mode)*. Prema [28], prvi argument predstavlja putanju do datoteke koja može biti apsolutna ili relativna, a drugi označava način rada (najčešće se koriste navedeni načini rada: „r“ – otvaranje datoteke za čitanje, „a“ – otvaranje za

pisanje na kraj već postojeće datoteke , „w“ – otvaranje za pisanje, „x“ – otvaranje za ekskluzivno stvaranje datoteke, neuspješno je ako datoteka već postoji). Za čitanje sadržaja datoteke potrebno je koristiti ugrađenu funkciju *read()*. Ona sprema sadržaj datoteke unutar varijable čiji je tip podatka *string*.

```
AuthClients {
    Isa = Clients
    Type = KeyValuePair
    Keys {
        Type = String
        ShortHelp = "Name of the prov chunk"
    }
    Values {
        Type = KeyValuePair
        ShortHelp = "clients allowed to access the 5G Authenticator"
        Keys {
            Type = String
            ShortHelp = "The global name of the client"
        }
        Values {
            Type = Hash
            ShortHelp = "The clients"
            Items {
                MaxSessionIdleTime {
                    Type = Duration
                    Default = 15s
                    ShortHelp = "Specifies the max. amount of time an authentication session may be idle before its state information is dismissed."
                }
                Authentication {
                    ShortHelp = 'The list of authentication profiles enabled for this client, in descending order of precedence'
                    Type = List
                    Keys {
                        Type = String
                    }
                }
                EAPKeyTableSource {
                    Type = Enumeration
                    Mandatory = false
                    ShortHelp = "Specifies the source of EAP profile data {NDS, STORE} "
                    Default = 'NDS'
                    Values {
                        NDS, STORE
                    }
                }
                KeyTableView {
                    Type = String
                    Mandatory = false
                    ShortHelp = "Name of the LDAPView to use when accessing the EAP key table via LDAPTD"
                }
            }
        }
    }
}
```

Sl. 4.1. Primjer CDL dokumenta

Na slici 4.1. prikazana je CDL datoteka *authclients.cdl* koju vidimo nakon korištenja prethodno navedenih funkcija.

Rad je otežan s takvim zapisom jer se pojavljuje puno različitih znakova. Za uklanjanje svih neželjenih znakova, na primjer: „\t“, „\n“„ = “, „ , “, korištena je funkcija *replace(string, string)* koja za prvi argument traži riječ koju je potrebno zamijeniti, a drugi argument je nova riječ kojom se mijenja stara. Svi neželjeni znakovi su zamijenjeni s prazninom, „ “.

Iduća funkcija koja se poziva je *tokenize(string)*. Ona za argument prihvaća varijablu koja je tip podatka *string*. Funkciji *tokenize* predana je varijabla koja predstavlja sadržaj datoteke. Nakon funkcije *replace(string, string)*, od svih znakova iz CDL datoteke u varijablu su se spremila slova,

vitičaste zagrade te dvostruki i jednostruki navodnici. Dobivena varijabla služi za stvaranje liste koja sadrži podatke koji su *string* tip podatka.

Na slici 4.2. prikazana je funkcija *tokenize*. Zastavica *token_start* služi za označavanje od kojeg indeksa počinje neka cjelina. Cjeline mogu biti: opis zapisan unutar navodnika, vitičaste zagrade i ostale riječi unutar dokumenta. Sve cjeline se spremaju u listu. Znakovi koje je potrebno promatrati su: navodnici, dvostruki i jednostruki jer opis unutar navodnika čini jednu cjelinu (linije 20 - 34). Oni se koriste za različite upute opisane unutar CDL datoteke. Zastavica *quotation_mode* označava koji se navodnik pojavio. Za dvostruki se koristi vrijednost 2 (linija 22), a za jednostruki 3 (linija 24). Zastavica *quotation_mode* ima vrijednost nula ako navodnika nema (linija 14) ili ako je došlo do kraja opisa (linija 36). Za zapisivanje riječi i vitičastih zagrada kao cjelina u listu, potrebno je odrediti početak i kraj njihovog pojavljivanja (linije 37 - 45). Rezultat „poziva“

funkcije *tokenize* je prikazan na slici 4.3. Funkcija „vraća“ listu koja se sastoji od riječi, složenih opisa i vitičastih zagrada.

```
def tokenize(string):
    tokens = []

    # 0 - none, 2 - ", 3 - '
    quotation_mode = 0
    token_start = -1

    for i in range(len(string)):
        # include spaces inside quotes
        if (quotation_mode != 0) and string[i] == " ":
            continue

        # start a quote
        if (quotation_mode == 0) and token_start == -1 and (string[i] == '"' or string[i] == "'"):
            if string[i] == '"':
                quotation_mode = 2
            else:
                quotation_mode = 3
            token_start = i
            continue

        # end a quote
        if (quotation_mode != 0) and token_start != -1 and (string[i] == '"' or string[i] == "'"):
            if (string[i] == '"' and quotation_mode == 2) or (string[i] == "'" and quotation_mode == 3):
                tokens.append(string[token_start + 1:i:])
                token_start = -1
                quotation_mode = 0
                continue

        # start a word
        if token_start == -1 and string[i] != " ":
            token_start = i
            continue

        # end a word
        if token_start != -1 and (quotation_mode == 0) and string[i] == " ":
            tokens.append(string[token_start:i:])
            token_start = -1
            continue

    return tokens
```

Sl. 4.2. Funkcija *tokenize*

```
['AuthClients', '{', 'Isa', 'Clients', 'Type', 'KeyValuePair', 'Keys', '{', 'Type', 'String', 'ShortHelp', 'Name
of the prov chunk', '}', 'Values', '{', 'Type', 'KeyValuePair', 'ShortHelp', 'clients allowed to access the 5G A
uthenticator', 'Keys', '{', 'Type', 'String', 'ShortHelp', 'The global name of the client', '}', 'Values', '{',
'Type', 'Hash', 'ShortHelp', 'The clients', 'Items', '{', 'MaxSessionIdleTime', '{', 'Type', 'Duration', 'Defaul
t', '15s', 'ShortHelp', 'Specifies the max. amount of time an authentication session may be idle before its stat
e information is dismissed.', '}', 'Authentication', '{', 'ShortHelp', 'The list of authentication profiles enab
led for this client  in descending order of precedence', 'Type', 'List', 'Keys', '{', 'Type', 'String', '}', '}'
, 'EAPKeyTableSource', '{', 'Type', 'Enumeration', 'Mandatory', 'false', 'ShortHelp', 'Specifies the source of E
AP profile data  { NDS  STORE }  ', 'Default', 'NDS', 'Values', '{', 'NDS', 'STORE', '}', '}', 'KeyTableView', '
{', 'Type', 'String', 'Mandatory', 'false', 'ShortHelp', 'Name of the LDAPView to use when accessing the EAP key
table via LDAPTD', '}', '}', '}', '}', '}']
```

Sl. 4.3. Ispis funkcije *tokenize*

4.3. Rječnik

Listu koja sadrži podatke iz CDL datoteka je potrebno zapisati u drugi tip podatka za lakše generiranje tablice. Za to je korištena funkcija *create_definition.*

```
def create_definition(tokens, name, index, isEnumeration):
    this_object = {}

    while(index < (len(tokens) - 1)):
        if tokens[index] == "}":
            return (index + 1, None, this_object)

        values = []
        if name == "Values" and isEnumeration:
            while(tokens[index] != "}"):
                values.append(tokens[index])
                index += 1
            index += 1
            return (index, values, None)

        defaults = []
        if tokens[index] == "Default" and tokens[index + 1] == "{":
            index = index + 2
            while(tokens[index] != "}"):
                defaults.append(tokens[index])
                index += 1

            this_object["Default"] = defaults
            index += 1
            continue

        long_help = ""
        if tokens[index] == "LongHelp" and tokens[index + 1] == "{":
            index = index + 2
            while(tokens[index] != "}"):
                long_help += tokens[index] + " "
                index += 1

            this_object["LongHelp"] = long_help
            index += 1
            continue

        short_help = ""
        if tokens[index] == "ShortHelp" and tokens[index + 1] == "{":
            index = index + 2
            while(tokens[index] != "}"):
                short_help += tokens[index] + " "
                index += 1

            this_object["ShortHelp"] = short_help
            index += 1
            continue

        if tokens[index + 1] == "{":
            new_index, values, new_object = create_definition(tokens, tokens[index], index + 2, isEnumeration)

            if values is not None:
                this_object["Values"] = values
            if new_object is not None:
                this_object[tokens[index]] = new_object

            index = new_index
            continue

        this_object[tokens[index]] = tokens[index + 1]
        if tokens[index] == "Type" and tokens[index + 1] == "Enumeration":
            isEnumeration = True
        index += 2

    return (index, None, this_object)
```

Sl. 4.4. Funkcija *create_definition*

Slika 4.4. prikazuje funkciju *create_definition* koja od liste stvara rječnik. Referirajući se na [29], rječnik je ugrađeni tip podatka u Python-u, a definiraju ga parovi engl. *(key, value)*, odnosno ključ i vrijednost. Uvjet je da je ključ jedinstven unutar rječnika. Zarezom se odvajaju parovi jedni od drugih. Također, par vitičastih zagrada predstavlja prazan rječnik, {}.

Funkciji *create_definition* predaju se četiri argumenta: *tokens*, *name*, *index* i *isEnumeration*. *Tokens* predstavlja listu koja se dobije nakon što se izvrši funkcija *tokenize*. Argument *name* je ime koje označava početak novog rječnika. Zatim, argument *index* predstavlja broj koji upućuje na mjesto unutar liste *tokens*. Zadnji argument *isEnumeration* predstavalja zastavicu koja može imati vrijednosti *True* ili *False* ovisno o atributu *Type*. Ako unutar rječnika atribut *Type* ima vrijednost *Enumeration* onda se zastavica postavlja na *True* u suprotnom, zastavica ima vrijednost *False*. Nakon analize datoteka, može se primijetiti kako atribut *Type* određuje kakav će tip podatka biti za atribute *Value*. Ukoliko *Type* ima vrijednost *Enumeration*, tada će atribut *Value* sadržavati listu podataka. Potrebno je paziti na nekoliko stvari za pravilno zapisivanje podataka u obliku (ključ, vrijednost). Uz već naveden jedan od uvjeta, treba obratiti pažnju na znak koji dolazi iza atributa *Default*, *LongHelp* i *ShortHelp*. Ukoliko je taj znak otvorena vitičasta zagrada, tada slijede podaci koje je potrebno spremiti u listu sve dok se ne pojavi zatvorena vitičasta zagrada. Također, funkcija *create_definition* se tada poziva rekurzivno za stvaranje novog rječnika unutar postojećeg rječnika koji će se popunjavati preostalim podacima. Kada *index* unutar liste *tokens* dođe do zatvorene vitičaste zagrade, to je znak za kraj rječnika.

Za ispis rezultata na slici 4.5. koristila se *json* biblioteka. Prema [30], korištenjem funkcije *json.dumps()* podskup Python objekata se pretvara u JSON niz. Ne mogu se svi objekti konvertirati pa je potrebno izraditi rječnik podataka.

Usporedbom slike 4.1. koja prikazuje CDL dokument i slike 4.5. koja prikazuje isti dokument zapisan kao rječnik, primijeti se kako razlika nije velika. Ali, ključno je zapisati dokument u onom obliku u kojem ga program najlakše može „pročitati“ i iskoristiti za krajnji rezultat.

Sl. 4.5. Ispis funkcije *create_definition*

4.4. Generiranje HTML, JSON i XML datoteka

Kako bi se stranica prikazala u željenom izgledu, potrebno je zadati stil koji određuje izgled tablica i tekst.

Sl. 4.6. HTML kod

Na slici 4.6. prikazan je HTML kod napisan u Python skripti te je definiran stil cijele HTML datoteke (linije 202 – 216). Za proširivanje i smanjivanje tablica napisan je CSS (linije 180 i 181) te JavaScript kod (linije 195 - 197). Također, osim tablice, na stranici se prikazuje i gumb koji će biti poveznica između HTML dokumenta i početne stranice *index.php*. Za pravilno generiranje tablica, napisana je funkcija *make_table* koja izvlači informacije iz rječnika i slaže ih na način da stvara jedinstvenu tablicu za svaku CDL datoteku (linija 193).

```
117 def make_table(object_name, object_value):
118     global html
119 
120     for (key, value) in object_value.items():
121         if isinstance(value, str):
122             html += f"<tr><td id='key' colspan='2'>{key}</td><td>{value}</td></tr>"
123         elif isinstance(value, list):
124             html += f"<tr><td id='key' colspan='2'>{key}</td><td>"
125             for i in range(0, len(value)):
126                 html += f"{value[i]}"
127                 if(i < len(value)-1):
128                 #   html+=", <br>" #for new line
129                     html += ", "
130             html += f"</td></tr>"
131         else:
132             html += f"<tr><td id='collapseButton' onclick='collapse(this)'>+</td><th>{key}</th><td id='hidden'><table id='new_table'>"
133             make_table(key, value)
134 
135     html += f"</table></td></tr>"
```

Sl. 4.7. Funckija *make_table*

Slika 4.7. prikazuje funkciju *make_table* koja za argumente uzima naziv tablice i vrijednosti koje se trebaju zapisati u tu tablicu. Za vrijednosti koje su tipa podataka *string* ili *list*, u jedan stupac se zapisuje naziv atributa, a u drugi vrijednost. Dođe li neki drugi tip podatka, unutar funkcije se stvara nova tablica te se događa rekurzija funkcije *make_table* (linija 133).

Za generiranje HTML datoteke koji sadrži informacije iz CDL datoteka potrebno je odrediti ime HTML datoteke i u nju spremiti sadržaj. Slika 4.8. prikazuje kod za stvaranje HTML datoteke. Ime datoteke se određuje prema imenu CDL datoteke i dodaje se ekstenzija *.html* (linija 212). HTML datoteka se sprema na određeno mjesto u mapi. Zatim, funkcija unutar HTML datoteke zapisuje sav prethodno napisan kod koji je zapisan u varijablu *html_content* (linija 213).

```
212     with open("html/"+name+'.html', 'w', encoding='utf-8') as html_file:
213         html_file.write(html_content)
```

Sl. 4.8. Stvaranje HTML datoteke

Zapisivanje JSON datoteke koja nosi iste podatke kao i HTML datoteka ne zahtijeva promjene podataka. Potrebno je rječnik zapisati kao *string*, za to se koristi već spomenuta funkcija

json.dumps(). Nakon toga je potrebno zapisati sadržaj u JSON datoteku. Slika 4.9. prikazuje postupak zapisivanja informacija u JSON oblik.

```
215     json_object = json.dumps(dictionary, indent = 4)
216     with open("json/"+name+".json", 'w') as json_file:
217         json_file.write(json_object)
```

Sl. 4.9. Stvaranje JSON datoteke

Kako bi se stvorila XML datoteka potrebno je koristiti biblioteke *xml.dom.minidom* i *dicttoxml*. Slika 4.10. prikazuje kod za stvaranje XML datoteke. Potrebno je pozvati funkciju *dicttoxml()*. Prema [https://snyk.io/advisor/python/dicttoxml], funkcija *dicttoxml()* pretvara rječnik ili neki drugi tip podatka u važeći XML niz. Kako unutar oznaka ne bi pisali tipovi podatka, atribut *attr_type* je potrebno postaviti na *False* (linija 219). Nakon toga, funkcija *parseString()* iz XML niza se stvara *string* koji sadrži sve podatke iz rječnika (linija 220). Zatim, slijedi dio koda koji sprema XML zapis podataka u XML datoteku.

```
219     xml = dicttoxml(dictionary, attr_type = False)
220     xml_string = parseString(xml).toprettyxml()
221     with open("xml/"+name+".xml", 'w') as xml_file:
222         xml_file.write(xml_string)
```

Sl. 4.10. Stvaranje XML datoteke

4.5. Prikaz početne stranice i tablica

Početna stranica služi da korisnicima olakša traženje HTML datoteka. Napisana je u programskom jeziku PHP pod datotekom *index.php*. Na slici 4.11. prikazana je stranica *index.php*. Unutar <*head*> elementa upotrijebljen je stil koji se želi primijeniti na element <*a*> i elemente koji imaju naziv klase *header*. Unutar elementa <*body*> nalazi se tekst koji se prikazuje na stranici (linije 25 – 27) te se nalazi PHP kod koji ispisuje sve generirane HTML dokumente na početnu stranicu (linije 30 – 42). Kako bi se klikom na odabrani HTML dokument moglo pristupiti željenim prikazima tablica, koristi se element <*a*> koji omogućuje definiranje hiperveze za povezivanje

različitih stranica. Unutar elementa je potrebno definirati vezu tako da se atributu *href* napiše gdje je destinacija druge stranice. Funkcionalnosti su prikazane u poglavlju 5.2.

index.php

```php
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        a{
            text-decoration: none;
            display: inline-block;
            color:black;
            margin: 10px 40px;
            font-size: 20px;
        }
        .header {
            padding: 30px;
            text-align: left;
            background: black;
            color: white;
            font-size: 20px;
        }
    </style>
</head>
<body>
    <div class="header">
        <h1>Configuration Reference</h1>
        <p>Here is the list of files. More information on product's configuration parameters is available in Configuration Reference document.</p>
    </div>

    <?php

        $path = __DIR__ ."/html";
        $files = scandir($path);

        foreach($files as $file){
            if($file != "." && $file != ".." && str_contains($file, ".html"))
            {
                $filename = pathinfo($path."/".$file);
                echo "<a href='html/{$file}'>{$filename['filename']}</a><br>";
            }
        }
    ?>
</body>
</html>
```

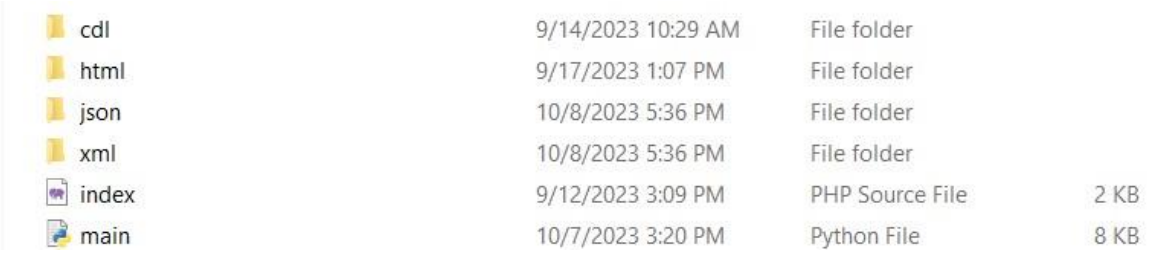
Sl. 4.11. Početna stranica *index.php*

5. KORIŠTENJE I PRIKAZ

U ovom poglavlju predstavljen je raspored datoteka te je objašnjeno korištenje alata. Nakon toga, opisane su funkcionalnosti početne stranice i tablica. Prikazan je i primjer veće CDL datoteke za prikaz funkcionalnosti na novom primjeru te za usporedba dviju datoteka.

5.1. Raspored datoteka i pokretanje

U glavnoj mapi se nalazi mapa za CDL datoteke, mapa za HTML datoteke, mapa za JSON datoteke, mapa za XML datoteke, početna stranica *index.php* i Python skripta *main.py*. Na slici 5.1. prikazane su navedene mape, početna stranica i skripta. Ovakav raspored nužan je kako bi kasnije bilo lakše pratiti gdje se nalazi koji tip datoteke. Također se ovakav raspored ostvaruje zbog preglednosti. Glavnu mapu je potrebno smjestiti na lokaciju *\xampp\htdocs* zbog kasnijeg pokretanja početne stranice.

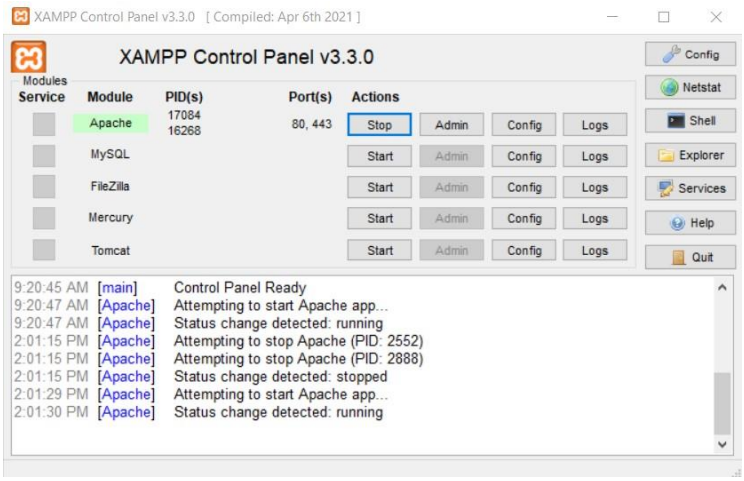


Sl. 5.1. Raspored datoteka

Potrebno je pokrenuti Python skriptu *main.py* koja će iz mape „*cdl*“ obraditi sve CDL datoteke koje se tamo nalaze i u mapu „*html*“ spremiti sve generirane HTML datoteke. Također, sve podatke koji su spremljeni u HTML datoteku zapisat će se i u obliku JSON-a i XML-a i spremit će se u datoteke „*json*“, odnosno datoteku „*xml*“.

Za uvid u rezultat obrade CDL datoteka, potrebno je pokrenuti lokalni poslužitelj, XAMPP. Ono služi za učitavanje PHP datoteka jer internetski preglednici to ne mogu bez poslužitelja.

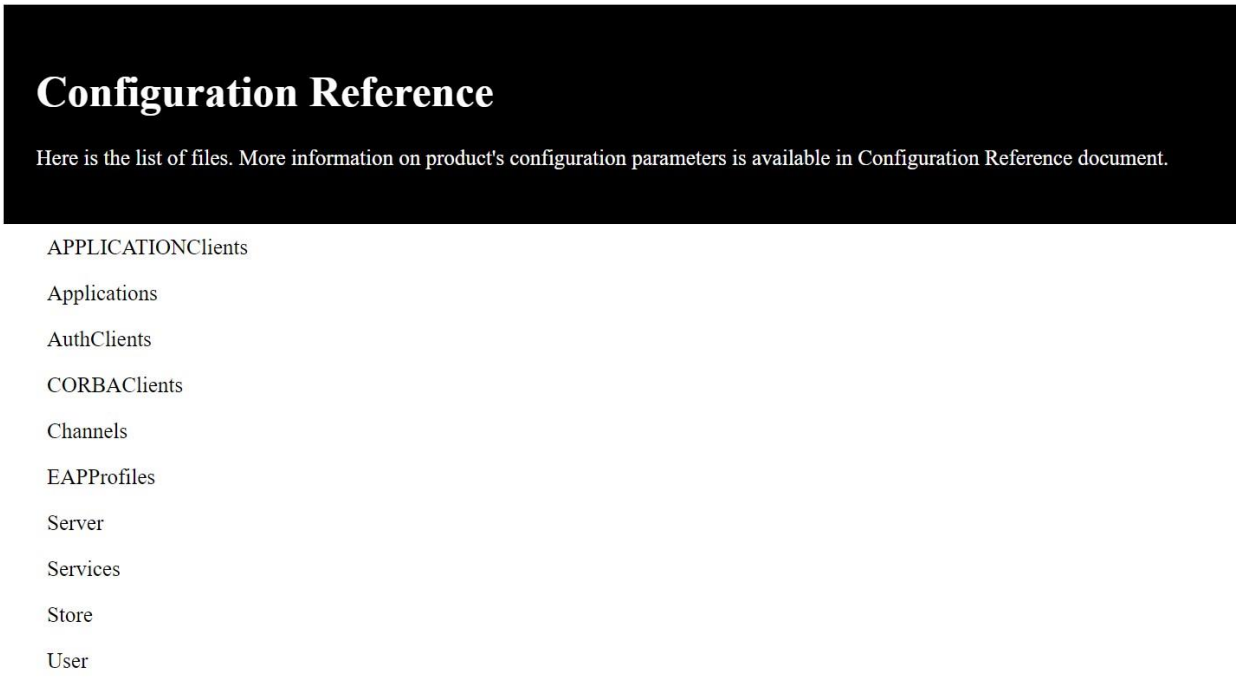
Na slici 5.2. prikazan je XAMPP kontrolni panel gdje se može primijetiti kako je pokrenut *Apache* modul. Nakon toga, na internetskom pregledniku je potrebno napisati *localhost\glavnaMapa* za automatski otvaranje početne stranice *index.php*.



Sl. 5.2. XAMPP kontrolni panel

5.2. Funkcionalnosti

Slika 5.3. prikazuje početnu stranicu. U zaglavlju se nalazi tekst, a ispod su izlistane sve HTML datoteke. Klikom na odabranu datoteku otvara se tablica koja je prethodno generirana obradom CDL datoteke pod tim istim imenom.

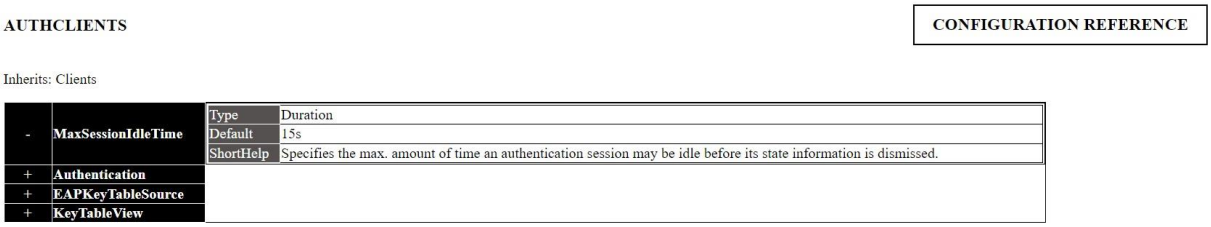


Sl. 5.3. Početna stranica *index.php*

Na slici 5.4. prikazana je generirana HTML datoteka za CDL datoteku *authclients.cdl*. U gornjem lijevom kutu se nalazi naziv datoteke. Ispod je ispisana vrijednost za atribut *Inherits* u slučaju da taj atribut postoji u CDL datoteci. U gornjem desnom kutu nalazi se gumb „*CONFIGURATION REFERENCE*“ koji vraća korisnika na početnu stranicu *index.php*. Na sredini ekrana se nalazi tablica.

Sl. 5.4. Prikaz *authclients.html* datoteke

Klikom na plus pored atributa, tablica se proširi samo za odabrani atribut i pokazuje sve podatke koji su prethodno bili nevidljivi.



Sl. 5.5. Prikaz proširenja tablice

Slika 5.5. prikazuje proširenje tablice za atribut „*MaxSessionIdleTime*“ gdje se mogu pročitati parametri i detalji o tom atributu. Na mjesto plusa se pojavio minus koji služi kako bi se tablica smanjila te bi se prikaz vratio na početni sa slike 5.4.

Također, tablica se može proširiti u cijelosti što je prikazano na slici 5.6. Time su vidljivi svi atributi, njihovi parametri te vrijednosti parametara.

AUTHCLIENTS **CONFIGURATION REFERENCE**

Inherits: Clients

-	**MaxSessionIdleTime**	Type	Duration
		Default	15s
		ShortHelp	Specifies the max. amount of time an authentication session may be idle before its state information is dismissed.
-	**Authentication**	ShortHelp	The list of authentication profiles enabled for this client in descending order of precedence
		Type	List
		- **Keys**	Type String
-	**EAPKeyTableSource**	Type	Enumeration
		Mandatory	false
		ShortHelp	Specifies the source of EAP profile data { NDS STORE }
		Default	NDS
		Values	NDS, STORE
-	**KeyTableView**	Type	String
		Mandatory	false
		ShortHelp	Name of the LDAPView to use when accessing the EAP key table via LDAPTD

Sl. 5.6. Prikaz proširenja cijele tablice

5.3. Primjer CDL datoteke

U ovom poglavlju je prikazano kako alat funkcionira za veće CDL datoteke.

```
User {
    Type = KeyValuePair
    Keys {
        Type = String
        ShortHelp = "Name of the prov chunk"
    }
    Values { Type = KeyValuePair, ShortHelp = 'Maintenance user accounts',
        LongHelp = "User Accounts are maintenance accounts by Name and Password.
                    Each account will get a profile of allowed views and granted permissions. They might be grouped using the 'Isa' machanism."
        Keys { Type = String, ShortHelp = 'A user account name' }
        Values { Type = Hash, ShortHelp = 'The user attributes'
            Items {
                Password    { Type = Password, Mandatory = true, ShortHelp = 'The logon password of this user', Group = 1Authentication }
                Expires     { Type = Date, ShortHelp = 'A limited validity', Group = 1Authentication }
                InactivitySessionTimeout { Type = Duration, ShortHelp = 'The inactivity session timer' }
                Role        { Type = Enumeration,
                Default = Marketing, Mandatory = true,
                Values { Administration, Management, Customer, Marketing },
                Group = 1Authentication }
                Permission        { Type = Enumeration, Default = Allowed, Values {Restricted, Allowed}, Group = 1Authentication }
                DefaultPerspective        { Type = Enumeration, Default = External, Values {Internal, External} }
                Gui         { Type = List, ShortHelp = 'The provided GUI views for this user', Group = 2Access, Keys { Type = String } }
                Read        {
                    Type = List,
                    ShortHelp = 'Sections allowed to be viewed',
                    Group = 2Access
                    Keys {
                        Type = Enumeration,
                        ShortHelp = 'Sections allowed to be viewed',
                        Values { *, APPLICATIONClients, ApplicationConfig, Applications, AuthClients, Authentication, Authenticator,
                        CDALClients, Cells, Certificates, Channels, Clients, ContentBasedConfig, COPSClients,
                        COPSDictionary, CORBAClients, CORBAConnections, DALClients, DHCPClients, DHCPDictionary, DHCPV,
                        DIAMETERClients, DiameterDictionary, DiameterPeers, DiameterRealmRoutingTables,
                        Dictionaries, Values, DNSClients, EAPClients, EAPProfiles, GenericClients, Handler, Hosts,
                        HostTemplates, HTTPClients, Instances, IPPools, Keycloak, KeyTables, LDAPClients, LDAPProfiles, LDAPTDClients,
                        LDAPUsers, LDAPViews, LIDictionary, Limits, LOADBALANCERClients, LoadEvaluationConfig, Logging, Meta, Modules,
                        Monitors, MYSQLClients, MYSQLProfiles, MYSQLUsers, NAFClients, NASSE, Nemo, NemoCustomPages, NemoThemes, NemoUsers,
                        NemoViews, OCSPClients, PCRFClients, PlainSample, PrimaClients, Props, Application, ProxyClients, ProxyDestinations,
                        ProxyGroups, ProxyServer, RADIUSClients, RadiusDictionary, ResourceMap, RESTProfile, RESTUsers, RoleTemplates,
                        SAMCounterTypes, SAMEvents, SAMMonitors, SAMOutputFiles, SAMPartitions, SAMPluginFiles, SAMPlugins, SAMPrometheus,
                        SCRClients, Server, ServerGroups, ServerPools, Servers, Services, SessionClients, Simulators, SnmpDestinations, SOAPClients,
                        SOAPProfiles, SOAPUsers, SPMLClients, Store, TDF, ThreadPools, TraceJobTemplates, TriggerDiameterClients, TriggerSoapClients,
                        UbClients, UMADATAACCESSClients, User, Versions, VNFConfig, WipProvClients, ZHProxyClient }
                        }
                }
                Write
                { Type = List,
                ShortHelp = 'Sections allowed to be modified',
                Group = 2Access
                    Keys {
                        Type = Enumeration,
                        ShortHelp = 'Sections allowed to be modified',
                        Values { *, APPLICATIONClients,
                        ApplicationConfig, Applications,
                        AuthClients, Authentication,
                        Authenticator, CDALClients, Cells,
                        Certificates, Channels, Clients, ContentBasedConfig,
                        COPSClients, COPSDictionary, CORBAClients, CORBAConnections,
                        DALClients, DHCPClients, DHCPDictionary, DHCPV, DIAMETERClients,
                        DiameterDictionary, DiameterPeers, DiameterRealmRoutingTables,
                        Dictionaries, Values, DNSClients, EAPClients, EAPProfiles,
                        GenericClients, Handler, Hosts, HostTemplates, HTTPClients,
                        Instances, IPPools, Keycloak, KeyTables, LDAPClients,
                        LDAPProfiles, LDAPTDClients, LDAPUsers, LDAPViews,
                        LIDictionary, Limits, LOADBALANCERClients, LoadEvaluationConfig,
                        Logging, Meta, Modules, Monitors, MYSQLClients, MYSQLProfiles,
                        MYSQLUsers, NAFClients, NASSE, Nemo, NemoCustomPages, NemoThemes,
                        NemoUsers, NemoViews, OCSPClients, PCRFClients, PlainSample,
                        PrimaClients, Props, Application, ProxyClients, ProxyDestinations,
                        ProxyGroups, ProxyServer, RADIUSClients, RadiusDictionary,
                        ResourceMap, RESTProfile, RESTUsers, RoleTemplates,
                        SAMCounterTypes, SAMEvents, SAMMonitors, SAMOutputFiles,
                        SAMPartitions, SAMPluginFiles, SAMPlugins, SAMPrometheus,
                        SCRClients, Server, ServerGroups, ServerPools, Servers, Services,
                        SessionClients, Simulators, SnmpDestinations, SOAPClients,
                        SOAPProfiles, SOAPUsers, SPMLClients, Store, TDF, ThreadPools,
                        TraceJobTemplates, TriggerDiameterClients, TriggerSoapClients,
                        UbClients, UMADATAACCESSClients, User, Versions, VNFConfig,
                        WipProvClients, ZHProxyClient }
                        }
                }
```

```
            NoRead        {
                Type = List,
                ShortHelp = 'Sections NOT allowed to be viewed',
                Group = 2Access
                Keys { Type = Enumeration,
                ShortHelp = 'Sections NOT allowed to be viewed',
                Values { *, APPLICATIONClients, ApplicationConfig, Applications, AuthClients, Authentication,
                Authenticator, CDALClients, Cells, Certificates, Channels, Clients, ContentBasedConfig, COPSClients,
                COPSDictionary, CORBAClients, CORBAConnections, DALClients, DHCPClients, DHCPDictionary, DHCPV, DIAMETERClients,
                DiameterDictionary, DiameterPeers, DiameterRealmRoutingTables, Dictionaries, Values, DNSClients, EAPClients, EAPProfiles,
                GenericClients, Handler, Hosts, HostTemplates, HTTPClients, Instances, IPPools, Keycloak, KeyTables, LDAPClients, LDAPProfiles,
                LDAPTDClients, LDAPUsers, LDAPViews, LIDictionary, Limits, LOADBALANCERClients, LoadEvaluationConfig, Logging, Meta, Modules,
                Monitors, MYSQLClients, MYSQLProfiles, MYSQLUsers, NAFClients, NASSE, Nemo, NemoCustomPages, NemoThemes, NemoUsers, NemoViews,
                OCSPClients, PCRFClients, PlainSample, PrimaClients, Props, Application, ProxyClients, ProxyDestinations, ProxyGroups, ProxyServer,
                RADIUSClients, RadiusDictionary, ResourceMap, RESTProfile, RESTUsers, RoleTemplates, SAMCounterTypes, SAMEvents, SAMMonitors,
                SAMOutputFiles, SAMPartitions, SAMPluginFiles, SAMPlugins, SAMPrometheus, SCRClients, Server, ServerGroups, ServerPools, Servers,
                Services, SessionClients, Simulators, SnmpDestinations, SOAPClients, SOAPProfiles, SOAPUsers, SPMLClients, Store, TDF,
                ThreadPools, TraceJobTemplates, TriggerDiameterClients, TriggerSoapClients, UbClients, UMADATAACCESSClients, User, Versions,
                VNFConfig, WipProvClients, ZHProxyClient } }
            }
            NoWrite
            { Type = List, ShortHelp = 'Sections NOT allowed to be modified', Group = 2Access
                Keys { Type = Enumeration, ShortHelp = 'Sections NOT allowed to be modified',
                        Values { *, APPLICATIONClients, ApplicationConfig,
                        Applications, AuthClients, Authentication, Authenticator, CDALClients,
                        Cells, Certificates, Channels, Clients, ContentBasedConfig, COPSClients,
                        COPSDictionary, CORBAClients, CORBAConnections, DALClients, DHCPClients, DHCPDictionary,
                        DHCPV, DIAMETERClients, DiameterDictionary, DiameterPeers, DiameterRealmRoutingTables,
                        Dictionaries, Values, DNSClients, EAPClients, EAPProfiles, GenericClients, Handler, Hosts,
                        HostTemplates, HTTPClients, Instances, IPPools, Keycloak, KeyTables, LDAPClients, LDAPProfiles,
                        LDAPTDClients, LDAPUsers, LDAPViews, LIDictionary, Limits, LOADBALANCERClients, LoadEvaluationConfig,
                        Logging, Meta, Modules, Monitors, MYSQLClients, MYSQLProfiles, MYSQLUsers, NAFClients, NASSE, Nemo,
                        NemoCustomPages, NemoThemes, NemoUsers, NemoViews, OCSPClients, PCRFClients, PlainSample, PrimaClients,
                        Props, Application, ProxyClients, ProxyDestinations, ProxyGroups, ProxyServer, RADIUSClients, RadiusDictionary,
                        ResourceMap, RESTProfile, RESTUsers, RoleTemplates, SAMCounterTypes, SAMEvents, SAMMonitors, SAMOutputFiles,
                        SAMPartitions, SAMPluginFiles, SAMPlugins, SAMPrometheus, SCRClients, Server, ServerGroups, ServerPools, Servers,
                        Services, SessionClients, Simulators, SnmpDestinations, SOAPClients, SOAPProfiles, SOAPUsers, SPMLClients, Store,
                        TDF, ThreadPools, TraceJobTemplates, TriggerDiameterClients, TriggerSoapClients, UbClients, UMADATAACCESSClients,
                        User, Versions, VNFConfig, WipProvClients, ZHProxyClient }
                    }
                }
            }
        }
    }
}
```

Sl. 5.7. *user.cdl* datoteka

Na slici 5.7. je prikazana CDL datoteka *user.cdl.* Nakon generiranja HTML datoteke na slici 5.8. se vidi izgled tablice za datoteku *user.cdl.* Prikazana tablica je manja od očekivanog, no ako se proširi svaki atribut, tablica postaje prevelika zbog količine atributa i vrijednosti koji se nalaze u CDL datoteci.

USER **CONFIGURATION REFERENCE**

+	**Password**
+	**Expires**
+	**InactivitySessionTimeout**
+	**Role**
+	**Permission**
+	**DefaultPerspective**
+	**Gui**
+	**Read**
+	**Write**
+	**NoRead**
+	**NoWrite**

Sl. 5.8. Prikaz *user.cdl* datoteke

Slika 5.9. prikazuje proširenu tablicu za *user.cdl* datoteku. Budući da se tablica ne može u cijelosti prikazati na ekranu i da je potrebno listanje kako bi se pronašli svi željeni podaci iz tablice, omogućena je još jedna funkcionalnost. Naslov datoteke i gumb za vraćanje na početnu stranicu su postavljeni tako da se uvijek nalaze na vrhu bez obzira lista li se stranica prema gore ili dolje. Time se osigurava da se korisnik uvijek može lako prebaciti na početnu stranicu, ali i da ima uvid u trenutnu datoteku koju promatra.

USER **CONFIGURATION REFERENCE**

-	**Password**	Type	Password
		Mandatory	true
		ShortHelp	The logon password of this user
		Group	1Authentication
-	**Expires**	Type	Date
		ShortHelp	A limited validity
		Group	1Authentication
-	**InactivitySessionTimeout**	Type	Duration
		ShortHelp	The inactivity session timer
-	**Role**	Type	Enumeration
		Default	Marketing
		Mandatory	true
		Values	Administration, Management, Customer, Marketing
		Group	1Authentication
-	**Permission**	Type	Enumeration
		Default	Allowed
		Values	Restricted, Allowed
		Group	1Authentication
-	**DefaultPerspective**	Type	Enumeration
		Default	External
		Values	Internal, External

Sl. 5.9. Prikaz *user.cdl* datoteke

6. ZAKLJUČAK

U današnje vrijeme velika se važnost pridaje vizualizaciji, posebno u poslovnom svijetu. Pod pojmom vizualizacije podataka podrazumijeva se prikazivanje informacija na kreativan i dosjetljiv način. Također, poželjno je da promatrač navedene vizualizacije može analizirati, odnosno dinamički i interaktivno sudjelovati u analiziranju podataka. Budući da različiti podaci neprestano dospijevaju u tvrtke, one se susreću s izazovima prikupljanja i obrade tih podataka. Vizualizacija olakšava analizu podataka te omogućava zaposlenicima bolje razumijevanje podataka što često dovodi do toga da se važne odluke mogu donijeti brže nego kada vizualizacija nije bila razvijena i pristupačna. Kako bi se prikupljeni podaci što kvalitetnije analizirali, tvrtke koriste različite metode i tehnike vizualizacije tih podataka. Najčešće korištene metode za vizualizaciju su grafovi, stupčasti grafikoni, kružni grafikoni, ali i tablice. Danas postoje brojni alati za kreativan prikaz podataka te za njihovu analizu.

Budući da podaci često budu neobrađeni i ponekad nepravilno zapisani, potrebno je uložiti truda i vidjeti kako se iz gomile podataka mogu izvući one najvažnije informacije. Važno je izbjeći nepotrebno opterećivanje informacijama. No, ponekad može doći do gubljenja podataka uslijed analiziranja što isto nije poželjno. Stoga, prije same vizualizacije podataka treba se odrediti i pronaći način zapisivanja informacija u što jednostavnijem obliku.

U ovom diplomskom radu je opisano stvaranje alata koji služi za prikaz dinamičkih tablica. Za njegovu izradu je bilo potrebno omogućiti korisniku da odredi koji su podaci za njega važni i što je sve potrebno ispisati u tablice. Izrada funkcionalnosti kojima korisnik može ostvariti interakciju s tablicama uvelike olakšava pronalazak željenih informacija.

Alat opisan u ovom radu bio je usmjeren na rješavanje specifičnih zahtjeva. Ukoliko dođe do potrebe za proširenjem zahtjeva ili nastankom novih zahtjeva alat će biti potrebno nadograditi. Alat ima mogućnost unapređivanja dodavanjem novih funkcionalnosti, npr. proširenje cijele tablice odjednom ili postavljanje tražilice koja bi olakšala pronalazak atributa u tablici. Također, korisnicima se može u budućnosti prilagoditi stil alata i količina informacija zapisanih u tablicama.

LITERATURA

[1] V, Otašević, B, Kosanović, "Primena softvera otvorenog koda kod vizualizacije podataka", Treća nacionalna konferencija sa međunarodnim učešćem "Primena slobodnog softvera i otvorenog hardvera PSSOH 2020", str. 34-37, Beograd, Srbija, veljača, 2021., dostupno na: https://zenodo.org/record/4135049/preview/Otasevic-Kosanovic-PSSOH-2020.pdf [Pristup: 27.6.2023.]

[2] P, Predrag, B, Jeren , "Besplatni programski alati", LSS-Labaratorij za signale i sustave, FER, Zagreb, dostupno na: https://www.lss.hr/hr/aktualno/besplatni-programski-alati.html [Pristup: 27.6.2023.]

[3] "Welcome to GnuCash.org", GnuCash, 2023., dostupno na: https://www.gnucash.org/ [Pristup: 27.6.2023.]

[4] Martin6, "The Best Data Visualization Tools", WebSetNet, 2020., dostupno na: https://websetnet.net/the-best-data-visualization-tools/ [Pristup: 27.6.2023.]

[5] "Microsoft Excel:", Microsoft, 2023., dostupno na: https://www.microsoft.com/en-us/microsoft-365/excel [Pristup: 25.8.2023.]

[6] "Modern Self-service BI and Analytics Platform", Zoho, dostupno na: https://www.zoho.com/analytics/ [Pristup: 25.6.2023.]

[7] "Experience the magic of Dreamforce: Tableau helps people see and understand data", Tableau, 2023, dostupno na: https://www.tableau.com/ [Pristup: 25.6.2023.]

[8] " Data Visualization", Sisense, dostupno na: https://www.sisense.com/platform/visualization/ [Pristup: 25.8.2023.]

[9] "10 Best Data Transformation Tools in 2023 (Comparison as per your Need)", Boltic, 2023., dostupno na: https://www.boltic.io/blog/best-data-transformation-tools [Pristup: 25.8.2023.]

[10] "Convert an entire website to PDF Easily", PDFmyURL, 2023., dostupno na: https://pdfmyurl.com/entire-website-to-pdf [Pristup: 25.6.2023.]

[11] T, Fisher,"31 Besplatni programi za pretvorbu videozapisa i online usluge: VideoSolo", eYewanted, dostupno na: https://hr.eyewated.com/31-besplatni-programi-za-pretvorbu-videozapisa-i-online-usluge/ [Pristup: 25.8.2023.]

[12] "HTML: HyperText Markup Language", MDN Web Docs, 2023., dostupno na: https://developer.mozilla.org/en-US/docs/Web/HTML [Pristup: 25.8.2023.]

[13] "<head>: The Document Metadata (Header) element", MDN Web Docs, 2023., dostupno na: https://developer.mozilla.org/en-US/docs/Web/HTML/Element/head [Pristup: 25.6.2023.]

[14] "<body>: The Document Body element", MDN Web Docs, 2023, dostupno na: https://developer.mozilla.org/en-US/docs/Web/HTML/Element/body [Pristup: 25.6.2023.]

[15] "CSS: Cascading Style Sheets", MDN Web Docs, 2023., dostupno na: https://developer.mozilla.org/en-US/docs/Web/CSS [Pristup: 25.6.2023.]

[16] "HTML Styles - CSS ", W3Schools, 2023., dostupno na: https://www.w3schools.com/html/html_css.asp [Pristup: 25.6.2023.]

[17] "What is JavaScript used for?", Galvanize, 2021., dostupno na: https://www.galvanize.com/blog/what-is-javascript-used-for/ [Pristup: 27.6.2023.]

[18] "What is JavaScript?", MDN Web Docs, 2023., dostupno na: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript [Pristup: 27.6.2023.]

[19] " PHP Intoduction ", W3Schools, 2023., dostupno na: https://www.w3schools.com/php/php_intro.asp [Pristup: 27.6.2023.]

[20] A.S. Ravikiran, "What is XAMPP?", Simplilearn, 2023.,dostupno na: https://www.simplilearn.com/tutorials/php-tutorial/php-using-xampp [Pristup: 27.6.2023.]

[21] "What is XAMPP?", Educba, 2023., dostupno na: https://www.educba.com/what-is-xampp/ [Pristup: 27.6.2023.]

[22] "Vodič za JSON: Uvod i cjelovit vodič za početnike", myservername.com, 2023. , dostupno na: https://hr.myservername.com/how-test-insurance-domain-application [Pristup: 27.6.2023.]

[23] " Introduction to XML ", W3Schools, 2023, dostupno na: https://www.w3schools.com/xml/xml_whatis.asp [Pristup: 27.7.2023.]

[24] "What Is Python Used For? A Beginner's Guide", Coursera, 2023, dostupno na: https://www.coursera.org/articles/what-is-python-used-for-a-beginners-guide-to-using-python [Pristup: 27.7.2023.]

[25] "What is Python?", Teradata, dostupno na: https://www.teradata.com/Glossary/What-is-Python [Pristup: 27.7.2023.]

[26] "Phyton - IDLE", TutorialsTeacher, dostupno na: https://www.tutorialsteacher.com/python/python-idle#google_vignette [Pristup: 27.7.2023.]

[27] M,Heller, "What is Visual Studio Code? Microsoft's extensible code editor", InfoWorld, 2022., dostupno na: https://www.infoworld.com/article/3666488/what-is-visual-studio-code-microsofts-extensible-code-editor.html [Pristup: 27.6.2023.]

[28] "Built-in Functions", Python, 2023., dostupno na: https://docs.python.org/3/library/functions.html#open [Pristup: 27.7.2023.]

[29] "Data Structures: Dictionaries", Python, 2023., dostupno na: https://docs.python.org/3/tutorial/datastructures.html#dictionaries [Pristup: 27.7.2023.]

[30] S,Chiramana, "json.dumps() in Python", Geeks for geeks, dostupno na: https://www.geeksforgeeks.org/json-dumps-in-python/ [Pristup: 27.6.2023.]

SAŽETAK

U ovom diplomskom radu opisan je razvoj alata za prikaz CDL datoteka. Alat služi kako bi se CDL datoteke mogle prikazati u tablice što omogućuje lakšu vizualizaciju datoteke te pomoć u analizi podataka. Analiza CDL datoteka nužna je kako bi se pronašao obrazac po kojem se pišu datoteke. Sam razvoj alata, odnosno pisanje koda ostvaren je kroz nekoliko segmenata koji na kraju čine jednu cjelinu. Python skripta je omogućila da se iz datoteka izvuku samo najznačajnije informacije te da se zapišu u novi oblik podatka. Zatim ih je bilo potrebno zapisati u HTML kod u obliku tablice. Kako bi korisnici lakše mogli pristupiti generiranim HTML tablicama stvorena je početna stranica, pisana u PHP-u, gdje se nalazi popis svih HTML dokumenata koji sadrže informacije zapisane u tablicama. Odabirom određenog dokumenta s popisa, prikazuje se željena tablica. Korisnik može sam odrediti koliko informacija iz tablica želi vidjeti što mu omogućuje interakciju s tablicama i bolju kontrolu nad pregledom informacija.

Ključne riječi: alati za vizualizaciju, CSS, HTML, JavaScript, PHP, Python, tablice, vizualizacija podataka

ABSTRACT

Title: Tool for graphic presentation of CDL files

This master thesis describes the development of tools for displaying CDL files. The tool is used so that CDL files can be displayed in a table, which would allow easier visualization of the file and would later help in data analysis. Analysis of the CDL file was necessary to find the pattern in which the files were written. The development of the tool itself, the writing of the code, had several segments that eventually form one whole. The Python script made it possible to extract only the most important information from the file and to write it into a new form of data. It was necessary to write them in the HTML code in the form of a table. In order to make it easier for users to access the generated HTML tables, a home page was created. It was written in PHP. On the home page there is a list of all HTML documents that contain the information written in the tables. By selecting a specific document from the list, the wanted table is displayed. The user can determine how much information from the table he wants to see, which allows him to interact with the tables and have better control over the visibility of the information.

Key words: visualization tools, CSS, HTML, JavaScript, PHP, Python, tables, data visualization

ŽIVOTOPIS

Magdalena Jukić rođena je 16. kolovoza 1999. godine u Slavonskom Brodu. Nakon završenog osnovnoškolskog obrazovanja, u osnovnoj školi Ivana Gorana Kovačića u Slavonskom Brodu, upisuje se u prirodoslovno-matematički smjer u „Gimnaziji Matije Mesića“. Nakon položene državne mature, 2018. godine upisuje „Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek“ preddiplomski studij na smjer Računarstva. Po završetku preddiplomskog studija nastavlja s daljnjim školovanjem te za diplomski studij odabire smjer Programsko inženjerstvo na istom fakultetu.

_____

Potpis autora

PRLOZI

[1] Poveznica za repozitorij na GitHub-u: https://github.com/Magdalena-Jukic/CDL_converter