Web aplikacija za reprodukciju glazbe

Matijaš, Nancy

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

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 03.09.2023.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada na preddiplomskom sveučilišnom studiju

Ime i prezime Pristupnika:	Nancy Matijaš
Studij, smjer:	Programsko inženjerstvo
Mat. br. Pristupnika, godina upisa:	R4538, 27.07.2020.
OIB Pristupnika:	71824054800
Mentor:	izv. prof. dr. sc. Josip Balen
Sumentor:	,
Sumentor iz tvrtke:	
Naslov završnog rada:	Web aplikacija za reprodukciju glazbe
Znanstvena grana rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Zadatak završnog rad:	U Završnom radu potrebno je napraviti web aplikaciju reprodukciju glazbe u kojoj korisnik nakon logiranja ima raznolike mogućnosti upravljanja glazbenim sadržajima (poput slušanja i spremanja svojih najdražih pjesma, izvođača, albuma, ima mogućnost pauziranja i preskakanja pjesama, itd.). Potrebno je napraviti i administratorski panel gdje treba biti omogućen pristup pristup i upravljanje svim korisnicima,
Prijedlog ocjene završnog 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: 2 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene od strane mentora:	03.09.2023.
Datum potvrde ocjene od strane Odbora:	08.09.2023.

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, 08.09.2023.

Ime i prezime studenta:	Nancy Matijaš
Studij:	Programsko inženjerstvo
Mat. br. studenta, godina upisa:	R4538, 27.07.2020.
Turnitin podudaranje [%]:	9

Ovom izjavom izjavljujem da je rad pod nazivom**: Web aplikacija za reprodukciju glazbe**

izrađen pod vodstvom mentora izv. prof. dr. sc. Josip Balen

i sumentora ,

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:

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I

INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni studij

WEB APLIKACIJA ZA REPRODUKCIJU GLAZBE

Završni rad

Nancy Matijaš

Osijek, 2023.

SADRŽAJ

1. UVOD

Glazba je neizostavni dio života koji nas okružuje. U ljudima budi razne osjećaje poput sreće, tuge ili pak bola. Kroz prošlost, svjedočili smo iznimnom tehnološkom napretku i raznim inovacijama koje su donijele brojne uređaje za reprodukciju glazbe. Počevši od preteče gramofona, fonografa, pa sve do pametnih telefona i računala. Uređaji za reprodukciju glazbe prošli su dug put i mijenjali su načine na koje pristupamo glazbi.

Davne 1887. godine Thomas Edison predstavlja preteču gramofona, fonograf, na kojem je puštena prva snimka. RCA Victor 1934. godine pušta u prodaju prvi gramofon priključen na zvučnike. 1954. godine, na tržištu se pojavljuje prvi prijenosni radio uređaj TR-1. Ovaj uređaj je označio početak slobode u slušanju glazbe jer je omogućio ljudima da sami odaberu radio stanice koje žele slušati, neovisno o mjestu na kojem se nalaze. Prijenosni radio uređaj je otvorio vrata razvoju glazbene industrije i radio stanica, što je rezultiralo rastom popularnosti pop-rock kulture tog vremena. Desetak godina nakon, na tržištu dolazi do pojave audio-kazeta koje omogućuju pohranu zvuka, a primarni uređaji za snimanje i reprodukciju zvuka postaju kazetofoni. U to vrijeme također dolazi i do pojave boombox uređaja, prijenosnog uređaja za reprodukciju glazbe s mogućnošću zamjene baterije i umetanja čak dvije kazete. 1979. godine, tvrtka Sony na tržište pušta prvi walkman TPS-L2. Walkman je osvojio veliku popularnost zahvaljujući svojoj kompaktnoj veličini i prenosivosti. Nekoliko godina nakon, na tržištu se pojavljuje Discman. Discman je poboljšana verzija Walkmana koja omogućava reprodukciju glazbe pomoću CD-ova umjesto kazeta. 2000.-ih godina, dolazi do pojave MP3 uređaja koji su osvojili široku popularnost zbog svoje kompaktne veličine, dugotrajne baterije, te mogućnosti pohrane i upravljanja određenim pjesmama uključujući brisanje i preskakanje [1].

U današnje vrijeme pametni telefoni postaju najpopularniji način za uživanje u glazbi. Pružaju nam mogućnost snimanja i reprodukcije glazbe koja se nalazi na uređaju, kao i pristup raznim aplikacijama za reprodukciju glazbe koje se mogu instalirati direktno na uređaj ili pronaći na web stranicama. Upravo iz tog razloga, svrha ovog završnog rada jest razviti web aplikaciju koja će omogućiti korisnicima lak pristup njihovim najdražim pjesmama neovisno o tome koriste li mobilni uređaj ili računalo.

Struktura rada sastoji se od 5 poglavlja. Prvo poglavlje opisuje zadatak i ciljeve koji se žele postići ovim radom. Drugo poglavlje prikazuje pregled sličnih rješenje, dok treće poglavlje detaljno opisuje razvojna okruženja i tehnologije koje su bile korištene prilikom izrade aplikacije.

U četvrtom poglavlju prikazani su dizajn i rad aplikacije, te su opisane njezine mogućnosti. Zadnje, odnosno peto poglavlje ovog rada predstavlja zaključak u kojem je sažet rad.

1.1. Zadatak završnog rada

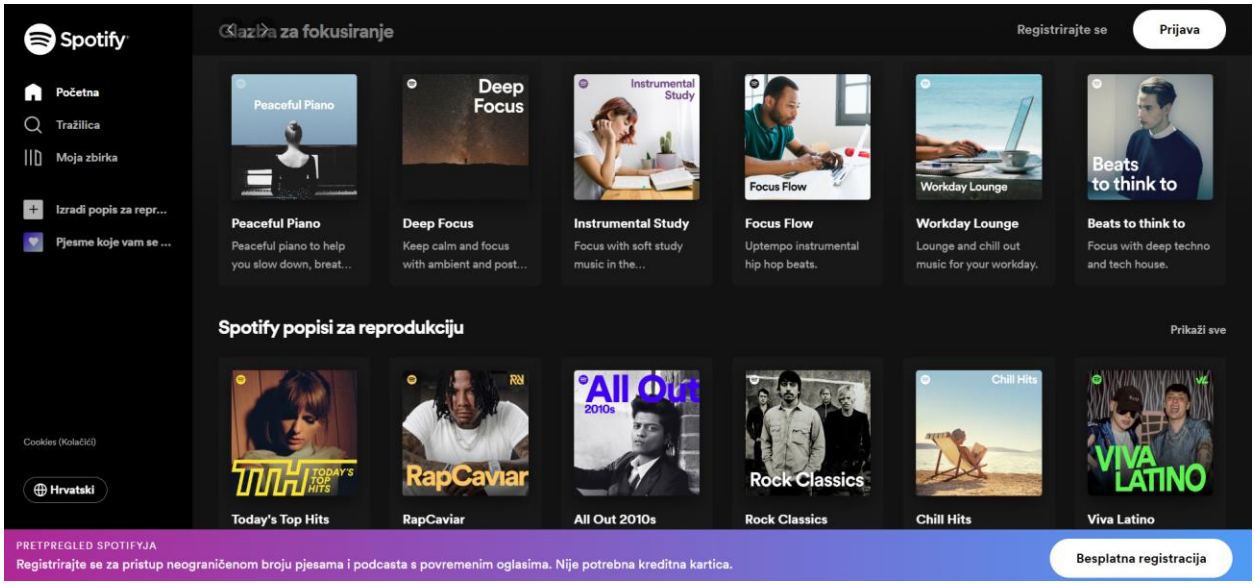
Zadatak ovog završnog rada jest realizacija aplikacije za reprodukciju glazbe. Korisnik nakon prijave ima raznolike mogućnosti upravljanja glazbenim sadržajima poput slušanja i spremanja svojih najdražih pjesma i albuma. Također ima mogućnost pauziranja, preskakanja, te nasumične reprodukcije pjesama. Potrebno je napraviti i administratorski panel gdje treba biti omogućen pristup i upravljanje svim korisnicima što uključuje pregled njihovih podataka i spremljenih lista za reprodukciju. Od razvojnih tehnologija potrebno je koristiti HTML, CSS, Bootstrap, JavaScript i MySQL.

2. PREGLED POSTOJEĆIH RJEŠENJA

S obzirom na iznimnu popularnost web aplikacija za reprodukciju glazbe, u nastavku teksta, navedene su neke od najpopularnijih aplikacija poput Spotify, YouTube Music, Apple Music i Deezer platformi. Ove aplikacije omogućavaju pristup velikom katalogu glazbenog sadržaja, pri čemu omogućavaju korisnicima da uživaju u glazbi na zahtjev, stvarajući personalizirane liste reprodukcije i otkrivajući nove glazbene izvođače. Uz mogućnost streaminga glazbe na mobilnim uređajima ili računalima, ove aplikacije pružaju korisnicima priliku da glazbu slušaju bilo gdje i bilo kada.

2.1. Spotify

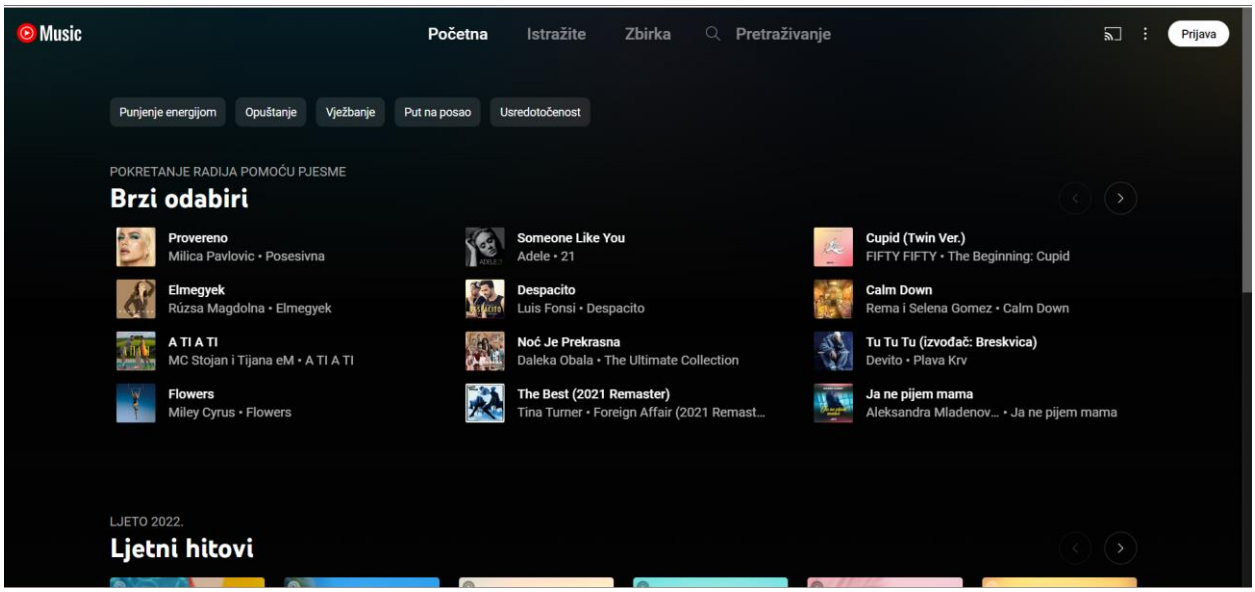
Spotify je nedvojbeno vodeće ime u industriji streaminga glazbe s impresivnih 165 milijuna pretplatnika. Na Spotify platformi gotovo je nemoguće ne pronaći željenu pjesmu. Također, platforma nudi originalni sadržaj poput videozapisa, podcast-ova i prijenosa uživo koji su ekskluzivni za Spotify. Platforma nudi raznolike glazbene žanrove, mogućnost filtriranja sadržaja prema osobnim preferencijama, streaming glazbe visoke kvalitete s tekstom pjesme, pregledavanje već postojećih lista za reprodukciju i razne druge mogućnosti koje se mogu vidjeti na slici 2.1. Spotify dolazi u besplatnoj i Premium verziji. Premium verzija nudi mogućnost slušanja glazbe bez oglasa, neograničenog preskakanja pjesama, te reprodukciju sadržaja izvan mreže [2].



Slika 2.1. *Početna stranica Spotify platforme [3].*

2.2. YouTube Music

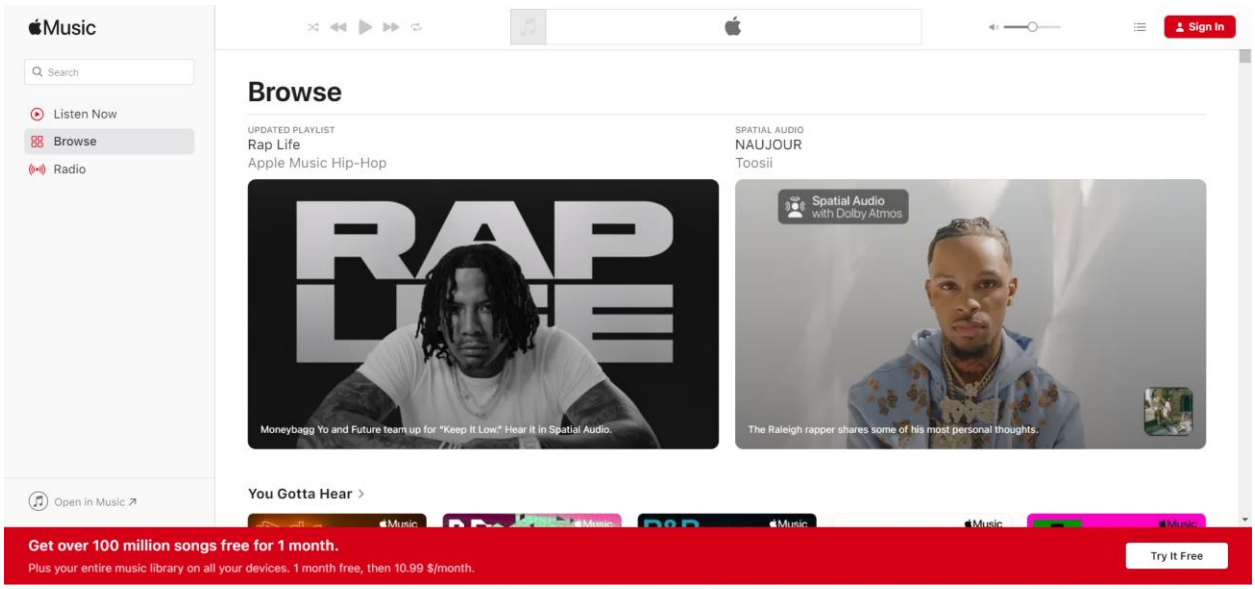
YouTube Music je platforma za streaming glazbe razvijena od strane YouTube mreže. Platforma olakšava otkrivanje novih pjesama, omogućava pretraživanje glazbenog sadržaja prema naslovu pjesme, imena izvođača ili imena albuma. Također moguće je pronaći pjesme upisivanjem teksta ili opisivanjem pjesme. Postoje dvije verzije YouTube Music platforme, a to su besplatna i Premium verzija. Na slici 2.2. prikazana je početna stranica YouTube Music platforme kada korisnik nije prijavljen. Premium verzija pruža dodatne mogućnosti poput reprodukcije bez oglasa i neograničenog preskakanja. Za razliku od YouTube mreže, YouTube Music Premium omogućava reprodukciju glazbe s isključenim zaslonom [2].



Slika 2.2. *Početna stranica YouTube Music platforme [4].*

2.3. Apple Music

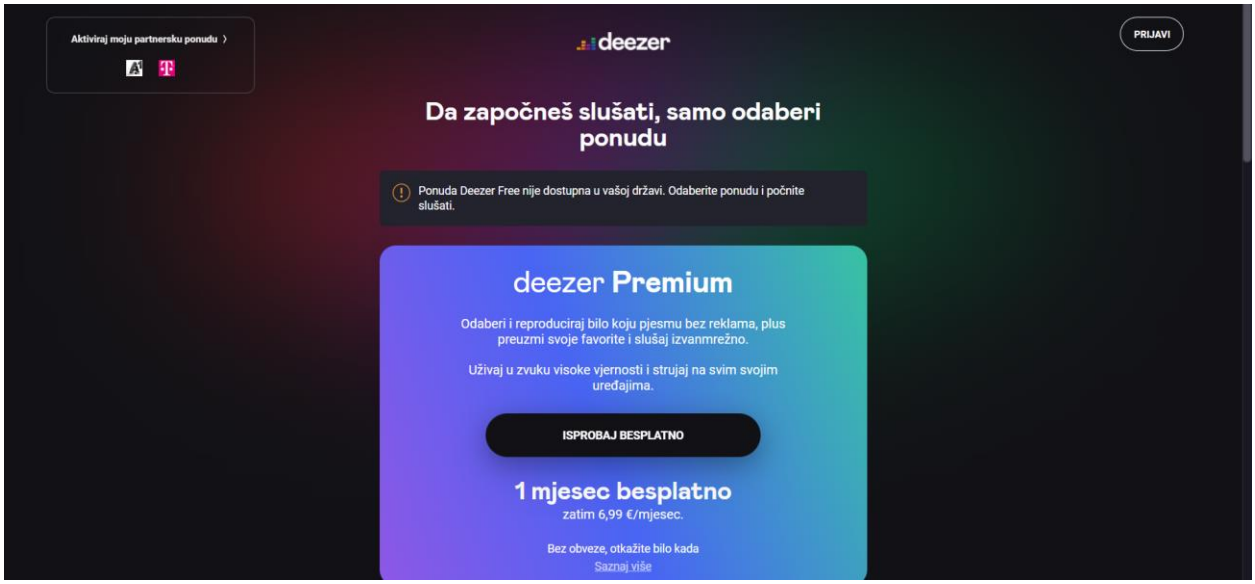
Apple Music je platforma za reprodukciju glazbenog sadržaja stvorena od strane Apple tvrtke. Ima biblioteku od preko 70 milijuna pjesama koje se mogu reproducirati na iOS i Android uređajima. Uz mogućnost pretraživanja željenog sadržaja, Apple Music nudi i tri radio stanice. Apple Music 1 emitira razne glazbene emisije i intervjue, Apple Music Hits reproducira popularne hitove, dok je Apple Music Country stanica posvećena country glazbi. Za razliku od Spotify platforme, Apple Music ne nudi besplatnu verziju za reprodukciju glazbenog sadržaja, već korisnici moraju imati pretplatu. Neprijavljeni korisnik može poslušati samo isječak izabrane pjesme u trajanju od 30 sekundi, a početna stranica neprijavljenog korisnika vidljiva je na slici 2.3. Mjesečna pretplata iznosi 10.99 dolara, a novim korisnicima je prvi mjesec besplatan [2].



Slika 2.3. *Početna stranica Apple Music platforme [5].*

2.4. Deezer

Deezer je glazbena platforma koja sadrži preko 73 milijuna audio zapisa iz cijelog svijeta, ali također nudi i vlastiti sadržaj poput glazbe, videozapisa i audio podcast-a. Na temelju reproduciranih pjesama, Deezer sastavlja liste pjesama za reprodukciju za pojedinačnog korisnika, ali nudi i mogućnost kreiranja vlastitih listi. Također moguće je pretraživanje određenih izvođača, albuma ili pjesama. Deezer platforma nije dostupna za besplatno korištenje u Republici Hrvatskoj što se može vidjeti na slici 2.4. Premium verzija Deezer aplikacije iznosi 6.99 eura mjesečno [3].



Slika 2.4. *Početna stranica Deezer platforme [6].*

3. TEHNOLOGIJE KORIŠTENE U IZRADI APLIKACIJE

Pri izradi ove aplikacije korištene su različite tehnologije kako bi se stvorilo funkcionalno i vizualno privlačno korisničko sučelje. Kod aplikacije pisan je u Visual Studio Code uređivaču izvornog koda. Za razvoj pristupnog sučelja (engl. *front-end*), te jest dijela aplikacije koji je vidljiv korisniku, korišteni su prezentacijski jezik HTML, stilski jezik CSS i Boostrap. Za razvoj pozadinskog sustava aplikacije (engl. *back-end*), korištena je Node.js platforma koja koristi JavaScript programski jezik kao interpretacijski jezik za obradu zahtjeva. Za bazu podataka korišten je MySQL koji je integriran putem XAMPP paketa. XAMPP je paket softvera koji uključuje Apache web server, MySQL bazu podataka, PHP i druge komponente potrebne za razvoj web aplikacija.

3.1. Visual Studio Code uređivač

Visual Studio Code je uređivač izvornog koda koji se izvodi na radnoj površini, a proizveden je od strane Microsofta. Dolazi s ugrađenom podrškom za Node.js, JavaScript i TypeScript. Omogućuje proširenja za razne jezike kao što su JavaScript, Python, C#, C++ i slično, kao i vrijeme izvođenja poput .NET i Unity [7]. Visual Studio Code dostupan je na raznim operacijskim sustavima poput macOS, Windows i Linux operacijskog sustava, što ga čini vrlo pristupačnim velikom broju korisnika. Još jedna od zanimljivih stavki, koja Visual Studio Code uređivač čini privlačnim mnogim korisnicima, jest njegova mogućnost prilagodbe. Korisnici mogu samostalno postaviti izgled platforme prema svojim preferencijama. Mogu mijenjati teme, raspored prozora, te razne druge aspekte sučelja.

3.2. HTML opisni jezik

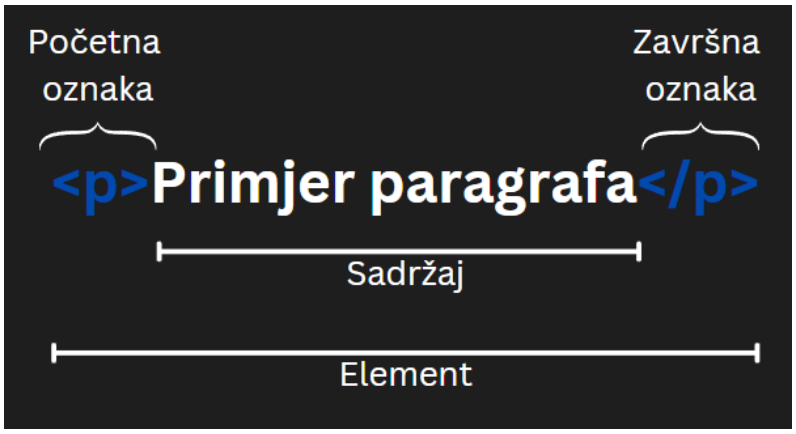
HTML (engl. *HyperText Markup Language*) je opisni jezik (engl. *markup language*) koji se koristi za izradu web stranica. HTML nije programski jezik jer ne stvara dinamičku funkcionalnost stranice. HTML datoteka može biti kreirana korištenjem bilo kojeg tekstnog editora. Kako bi se prikazala unutar Internet preglednika (npr. Google Chrome, Mozilla Firefox, Microsoft Edge) treba biti spremljena s ekstenzijom .html [8]. HTML dokument sastoji se dva dijela, a to su zaglavlje (engl. *head*) i tijelo (engl. *body*) dokumenta. U zaglavlju se definiraju elementi koji se ne prikazuju na stranici. Npr. naslov *(*engl. *title)* koji se ne prikazuje izravno na HTML stranici, već u imenu kartice unutar web preglednika. Tijelo dokumenta predstavlja sadržaj

koji se prikazuje unutar web preglednika. Primjer osnovne strukture HTML dokumenta prikazan je programskim kodom 3.1.

```
<!DOCTYPE html>
    <head>
        <title>Naslov stranice</title>
    </head>
    <body>
        Sadržaj stranice
    </body>
</html>
```

Programski kod 3.1. *Osnovna HTML struktura.*

Na slici 3.1. prikazana je struktura HTML elementa. HTML element sastoji se od oznake za početak, sadržaja i oznake za kraj (koja nije nužna za sve elemente). Početna oznaka sadrži ime elementa koje se nalazi unutar znakova <*i*>. Sadržaj elementa predstavlja tekst koji se nalazi između početne i završne oznake. Završna oznaka ima sličnu strukturu kao i početna oznaka, a jedina je razlika znak / prije imena elementa [9].

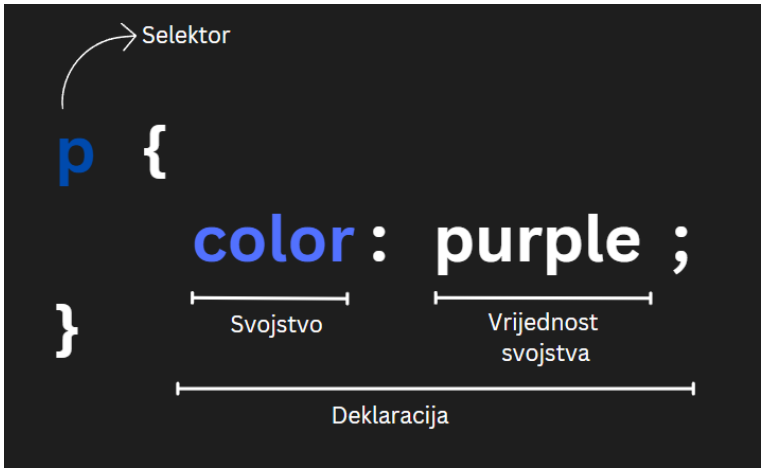


Slika 3.1. *Struktura HTML elementa.*

3.3 CSS stilski jezik

CSS (engl. *Cascading Style Sheets*) stilski je jezik koji se koristi za uređivanje i stiliziranje elemenata unutar HTML dokumenta. Pomoću njega definira se način prikazivanja HTML elemenata. Moguće ga je definirati unutar samog elementa (engl. *inline*), unutar <*style*> oznake (unutrašnji stil) ili kreiranjem posebne datoteke s ekstenzijom .css (vanjski stil). CSS sintaksa, prikazana je na slici 3.2., a sastoji se od selektora koji predstavlja naziv HTML elementa koji se stilizira, te deklaracije. Deklaracija se sastoji od svojstva i vrijednosti svojstva, a predstavlja

pravilo koje određuje koje se svojstvo elementa želi urediti. Svojstvima se opisuju pojedini atributi, a pomoću vrijednosti svojstava odabire se jedan od mogućih izgleda za određeno svojstvo [10].



Slika 3.2. *CSS sintaksa.*

Postoje razne vrste selektora, a neke od najpoznatijih su univerzalni selektor koji odabire sve elemente u dokumentu, *id* selektor koji specificira stil za jedan jedinstveni element, te klasni selektor koji specificira stil za grupu elemenata. U programskom kodu 3.2. prikazano je unutrašnje definiranje CSS programskog koda. Prvo, uveden je univerzalni selektor (*) koji primjenjuje stilove na sve elemente u cijelom HTML dokumentu. Konkretno, ovaj selektor postavlja *padding* za svaki element na 20 piksela. Zatim je definiran klasni selektor koji za sve elemente koji imaju klasu *container* postavlja gornju marginu na 100 piksela. Na kraju, definiran je *id* selektor koji se primjenjuje na HTML element s atributom *id* postavljenim na *sidebar*. Ovaj selektor mijenja boju teksta elementa na crnu.

```
<style>
    *{
        padding: 20px;
    }
    .container{
        margin-top: 100px;
    }
    #sidebar{
        color: black;
    }
</style>
```

Programski kod 3.2. *Primjer programskog koda za CSS.*

3.4. Node.js i JavaScript

Node.js je izvršno okruženje (engl. *runtime*) za JavaScript koje se koristi na strani servera i omogućava izvršavanje JavaScript koda izvan web preglednika. Temelji se na V8 tehnologiji koja je napisana u C++ programskom jeziku, te predstavlja Google JavaScript i WebAssembly mehanizam otvorenog koda. Glavna zadaća Node.js platforme je pružiti siguran i lak način izgradnje web aplikacija visokih performansi u JavaScript programskom jeziku. Jedna od glavnih prednosti je mogućnost obrade velikog broja zahtjeva, istovremeno, bez blokiranja izvršavanja drugih operacija. Node.js izvršno okruženje, osim pri izradi web stranica, koristi se i pri izradi raznih alata, poput PostCSS alata, pri testiranju web korisničkog sučelja, pri izradi desktop i mobilnih aplikacija [11]. U razvoju ove aplikacije korišten je i Express.js, standardni dio Node.js platforme. Express.js je popularni okvir (engl. *framework*) koji se koristi za izgradnju pozadinskog sustava web aplikacija i temelji se na Node.js platformi.

JavaScript je dinamički, objektno-orijentirani programski jezik. Koristi se za razvoj računalnih igara, web stranica i slično. Omogućava implementaciju dinamičkih značajki koje se ne mogu postići isključivo korištenjem HTML i CSS jezika, poput padajućih izbornika i animacija. Programi napisani u ovom jeziku nazivaju se skriptama, te ih je moguće pisati izravno unutar HTML dokumenta ili u zasebnim datotekama s ekstenzijom .js. Programski kod se automatski se izvršava prilikom učitavanja web stranice. JavaScript moguće je uključiti u HTML dokument na nekoliko načina. Prvi i najčešći način jest uključivanje programskog koda iz druge datoteke pomoću *src* atributa. Primjer za ovaj način povezivanja prikazan je programskim kodom 3.3.

```
<script src="layout.js"></script>
```

Programski kod 3.3. *Primjer povezivanja sa vanjskom skriptom.*

Drugi način umetanja programskog koda jest unutar HTML datoteke. U HTML datoteci, JavaScript kod umeće se unutar <*script*> i </*script*> oznaka. JavaScript kod može biti smješten bilo gdje na stranici, a najčešće se nalazi unutar <*head*> dijela ili na kraju <*body*> dijela. Unutar HTML dokumenta moguće je umetnuti više od jedne skripte, koje se izvršavaju redom kojim su napisane. Treći način uključivanja JavaScript koda jest umetanje kao URL korištenjem <*a href*> oznake, umjesto uobičajenog *https://* u ovom slučaju navodi se *javascript:*. Četvrti je način umetanje programskog koda unutar atributa na način prikazan programskim kodom 3.4. [12].

```
<!DOCTYPE html>
<body>

    <h2>Primjer</h2>

    <button onclick="document.getElementById('myImage').src='pic_bulbon.gif'">
        Upali
    </button>

    <img id="myImage" src="pic_bulboff.gif" style="width:100px">

    <button
onclick="document.getElementById('myImage').src='pic_bulboff.gif'">
        Ugasi
    </button>

    </body>
</html>
```

Programski kod 3.4. *Primjer pokretanja JavaScript koda unutar HTML atributa [13].*

Programski kod prikazuje paljenje i gašenje žarulje pritiskom na gumbove *Upali* i *Ugasi*. Zadana vrijednost žarulje, prilikom prvog učitavanja stranice, jest ugašeno. Pritiskom na gumb *Upali*, svijetlo na žarulji se pali, te ona mijenja boju u žuto. Pritiskom na gumb *Ugasi* svijetlo na žarulji se gasi, te jest, žarulja ponovno postaje siva.

3.5. Bootstrap okvir

Bootstrap je jedan od najpoznatijih okvira za izradu korisničkih sučelja putem HTML, CSS i JavaScript jezika. Ovaj okvir ima za cilj olakšati dizajniranje responzivnih web stranica, posebno naglašavajući prilagodbu mobilnim uređajima. Osigurava da se svi elementi unutar web stranice pravilno prikazuju neovisno o veličini zaslona. Bootstrap nudi bogatu kolekciju predložaka temeljenih na HTML i CSS jezicima, što uključuje stiliziranje tipografije, obrazaca, tablica, gumba, navigacije te raznih drugih kontrola. Ovaj okvir uključuje i raznovrsne JavaScript dodatke kako bi se povećala funkcionalnost korisničkog sučelja. Jedna od ključnih prednosti Bootstrap okvira leži u njegovoj kompatibilnosti s raznim preglednicima kao što su Mozilla Firefox, Google Chrome, Opera, Safari i drugi.

Da bi Bootstrap okvir pravilno funkcionirao, koristite se HTML5 elementi i CSS svojstva koja su kompatibilna s HTML5 doctype. Atributom *lang*, koji se koristi za pravilno stiliziranje elemenata na već stranici, potrebno je postaviti korištenje engleskog jezika. Za ispravnu

funkcionalnost prikaza (engl. *rendering*) i zumiranja, potrebno je dodati i oznaku <*meta*> unutar zaglavlja dokumenta. Programski kod 3.5. prikazuje pravilno dodavanje atributa i oznake.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
</html>
```

Programski kod 3.5. *Osnovni HTML predložak koji koristi Bootstrap okvir.*

Prilikom implementacije, Bootstrap se sastoji od tri glavne mape: *css*, *js* i *img*. Ove je mape preporučljivo smjestiti u korijenski direktorij projekta. U sklopu Bootstrap okvira dolaze i komprimirane verzije CSS i JavaScript jezika. Iako nije obavezno koristiti obje verzije, komprimiranu i nekomprimiranu, preporučuje se upotreba nekomprimirane verzije tijekom razvoja, dok se za produkciju preporučuje prelazak na komprimiranu verziju radi optimizacije performansi i brzine učitavanja [13].

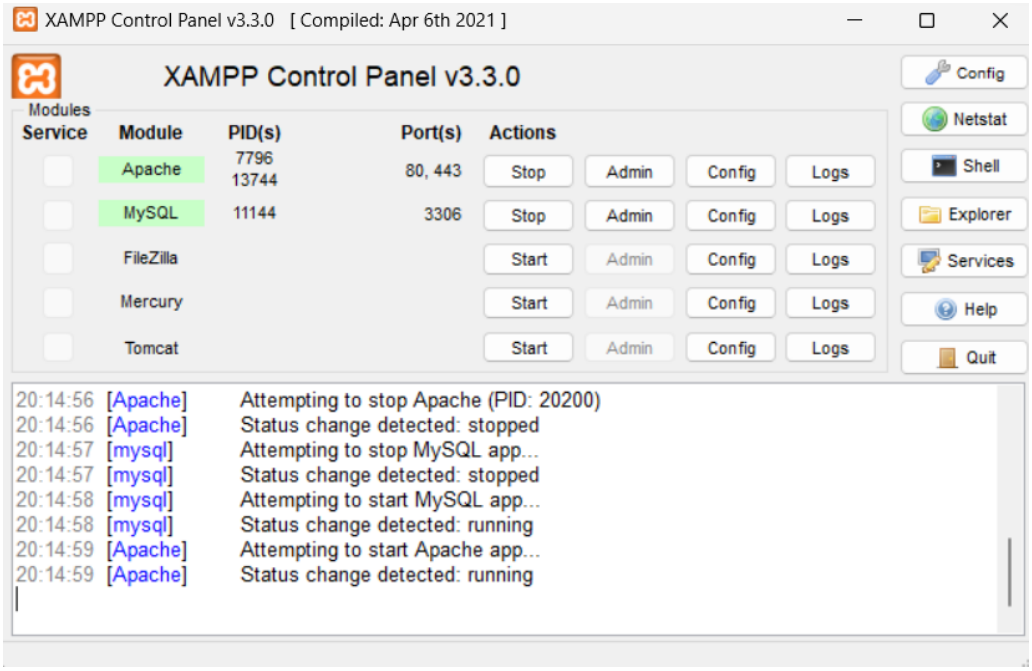
3.6. MySQL baza podataka

SQL (engl. *Structured query language*) je strukturni jezik za pretraživanje, izradu i obradu podataka relacijskih baza podataka. Pruža naredbe za izvođenje različitih zadataka, uključujući postavljanje upita za podatke, uklanjanje, dodavanje i mijenjanje redaka u tablicama, uklanjanje, stvaranje i mijenjanje objekata sheme i kontrolu pristupa bazama podataka i objektima sheme. Također osigurava dosljednost baze podataka. SQL je neproceduralni jezik, otvorenog koda. Jednostavan je za učenje jer za naredbe koristi riječi koje su uobičajene u engleskom jeziku. Neke od naredba su *SELECT, INSERT, DELETE* i slično. Podatci u SQL jeziku mogu biti različitog tipa, a neki od najčešćih su CHAR, VARCHAR, INT i NUMERIC.

MySQL je otvoreni relacijski SQL sustav za upravljanje relacijskim bazama. MySQL sustav postiže iznimnu popularnost zbog svoje brzine, pouzdanosti i lakoće upotrebe [14]. Pojavljuje se prvi put 1995. godine i od tog trenutka bilježi veliki broj novih nadogradnji u raznim područjima.

3.7. XAMPP

XAMPP (engl. *X – Cross platform, A – Apache, M – MySQL, P – PHP, P – Pearl*) besplatni je paket softvera (engl. *software stack*) otvorenog koda. Razvijen je od strane grupe nazvane Apache Friends. XAMPP pruža Apache server koji omogućava korisnicima jednostavno testiranje baza podataka i aplikacija koje koriste te baze podataka. Koristi se za kreiranje lokalne baze podataka na računalu, pri čemu je baza dostupna samo korisniku računala. Za izradu ove aplikacije potrebno je uključiti Apache i MySQL kao što je prikazano na slici 3.3. Nakon instaliranja XAMPP paketa, moguće je pristupiti phpMyAdmin sučelju putem web preglednika i koristiti ga za upravljanje MySQL bazama podataka koje su povezane s lokalnim web serverom.



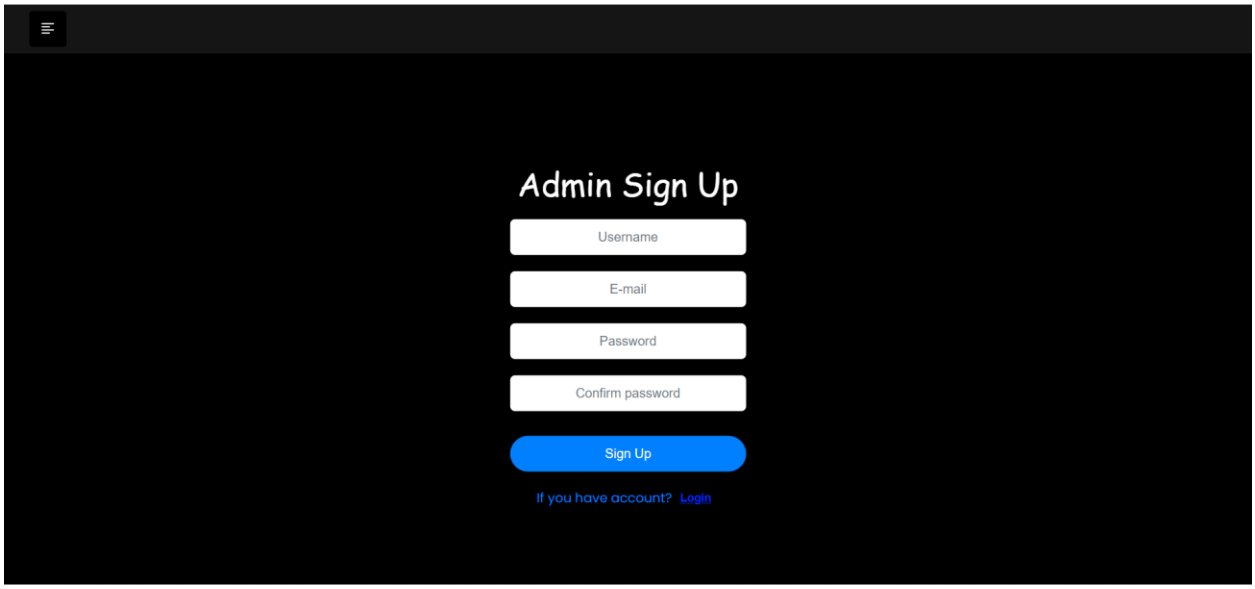
Slika 3.3. *Prikaz XAMPP kontrolnog panela.*

4. RAZVOJ APLIKACIJE

U ovom poglavlju detaljno je opisan razvoj web aplikacije. Poglavlje se sastoji od 2 dijela. U prvom dijelu prikazan je opis administratorskog dijela aplikacije koji uključuje registraciju i prijavu administratora, te prikaz administratorskog panela. U drugom dijelu prikazan je pogled na aplikaciju od strane korisnika.

4.1. Administratorski dio aplikacije

Kako bi se pristupilo administratorskom panelu potrebno je otvoriti stranicu *http://localhost:4567/admin/signup*. Nakon unošenja adrese otvara se sustav za registraciju administratora što je prikazano slikom 4.1. Ukoliko je administrator već prijavljen u sustavu tada je potrebno pritisnuti *Login* na dnu stranice kako bi se korisnik preusmjerio na formu za prijavu.



Slika 4.1. *Registracija administratora.*

Nakon uspješne registracije, administrator se sprema u tablicu *admins* koja je prikazana na slici 4.2. Tablica se sastoji od stupaca *id, username, email, password* (koji je zaštićen pomoću hash funkcije), te stupca *is_verify* koji prikazuje je li registrirani administrator verificiran.

	id	username	email	password	is_verify
Edit Copy Delete	2	admin2	admin2@gmail.com	$2b$10$5qTWX.Tle6mWb7kcKxz.3ufdulEnKChXzR0hi2.TgKD...	1
Edit Copy Delete	5	admin3	admin3@gmail.com	$2b$10$sjQ2DVZlF6c6YKfyW6YNAO04TTPpvfH1gWfsYSa.9p6...	1
Edit Copy Delete	6	admin4	admin4@gmail.com	$2b$10$4FQXPO1OJOBqSC7cC1zVN.XbzCJG6fB5BdDXHYFPaC4...	1
Edit Copy Delete	7	admin7	admin7@gmail.com	$2b$10$v8JPNB4RYfbzdonYyKjoPOOpTYHLMBIAMGN4VVCugXR...	1
Edit Copy Delete	8	admin5	admin5@gmail.com	$2b$10$qdP1nvbh2BEG8giae5K9c./hH2PZA6tvY5AIK7ZrKW/...	0
Edit Copy Delete	9	admin78	admin78@gmail.com	$2b$10$8q85lbgJMFi3IOnua24z7.40tNmnnUcQGlOyf9TsRL4...	0

Slika 4.2. *Tablica admins unutar MySQL baze podataka.*

Administrator pri registraciji nije verificiran, stupac *is_verify = 0*, te ga verificirani administrator potvrđuje pritiskom na gumb *VERFIY*. Programskim kodom 4.1. prikazana je ruta za verifikaciju administratora. U bazi podataka provjerava se postoji administrator s određenom *id* vrijednosti i stupcem *is_verify = 1*. Ako ne postoji ili nije verificiran, vraća korisniku poruku i preusmjerava ga na prijavu administratorskog računa.

```
router.post('/verify_admin', function (req, res) {
    const adminId = req.body.adminId;

    db.query("SELECT * FROM admins WHERE id = ? AND is_verify = 1",
[req.session.admin], (error, results) => {
        if (error) throw error;

        if (results.length === 0) {
            return res.send('<script>alert("Administrator nije verificiran!");
window.location.href="/admin/login";</script>');
        }

        db.query("UPDATE admins SET is_verify = 1 WHERE id = ?", [adminId],
(error, result) => {
            if (error) throw error;
            res.redirect('/admin/dashboard_admin');
        });
    });
});
```

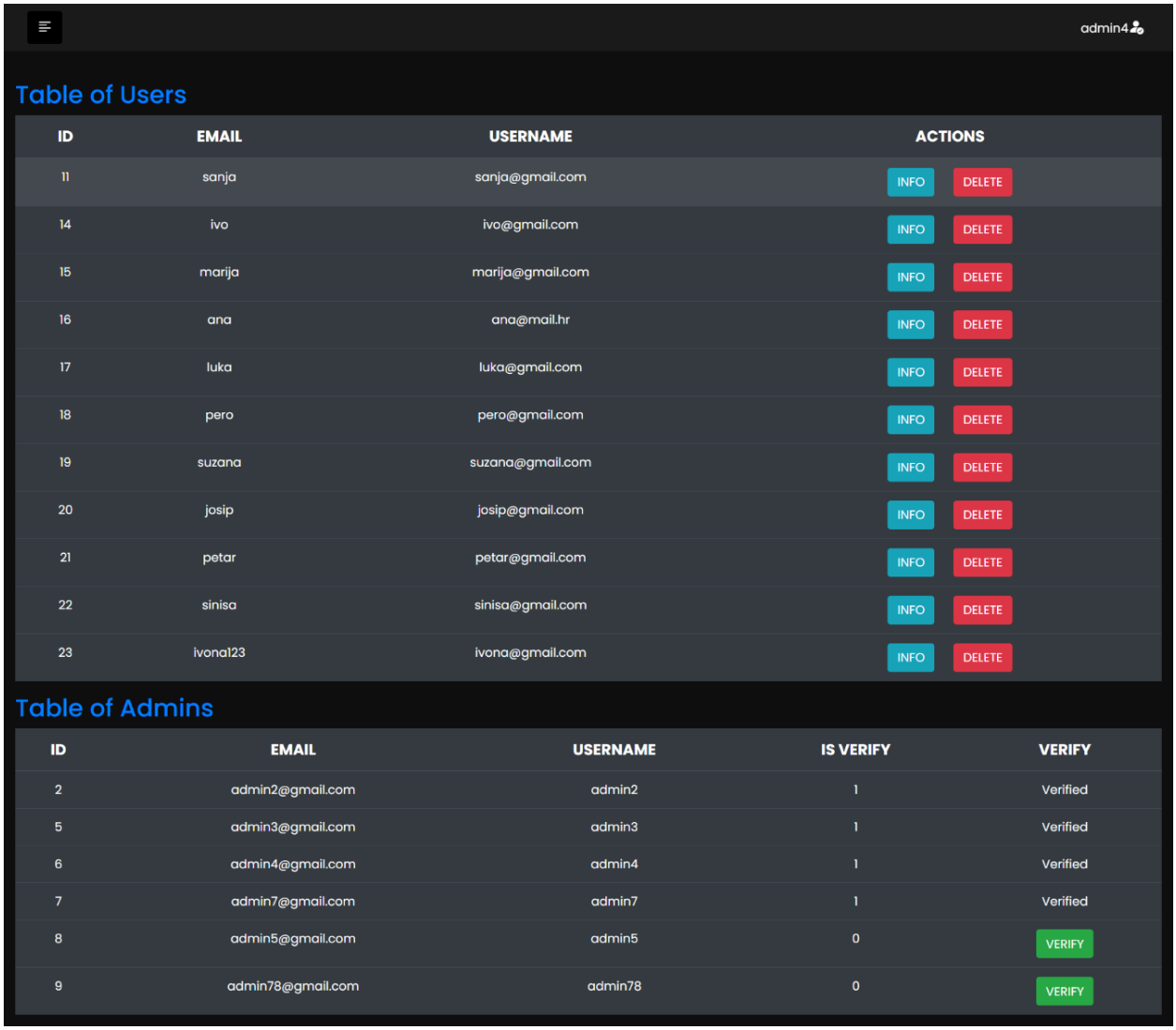
Programski kod 4.1. *Prikaz rute za verifikaciju administratora.*

Također verificirani administrator ima mogućnost pregleda svih korisnika koji se nalaze u tablici *users*. Klikom na gumb *DELETE* administrator briše korisnika iz baze zajedno sa svim sadržajem vezanim uz njega. Programski kod 4.2. prikazuje rutu za brisanje korisnika iz baze.

```
router.post('/delete_user', function (req, res) {
    const userId = req.body.userId;

    db.query("DELETE FROM users WHERE id = ?", [userId], (error, result) => {
        if (error) throw error;

        console.log(`User with ID ${userId} deleted successfully`);

        res.redirect('/admin/dashboard_admin');
    });
});
```

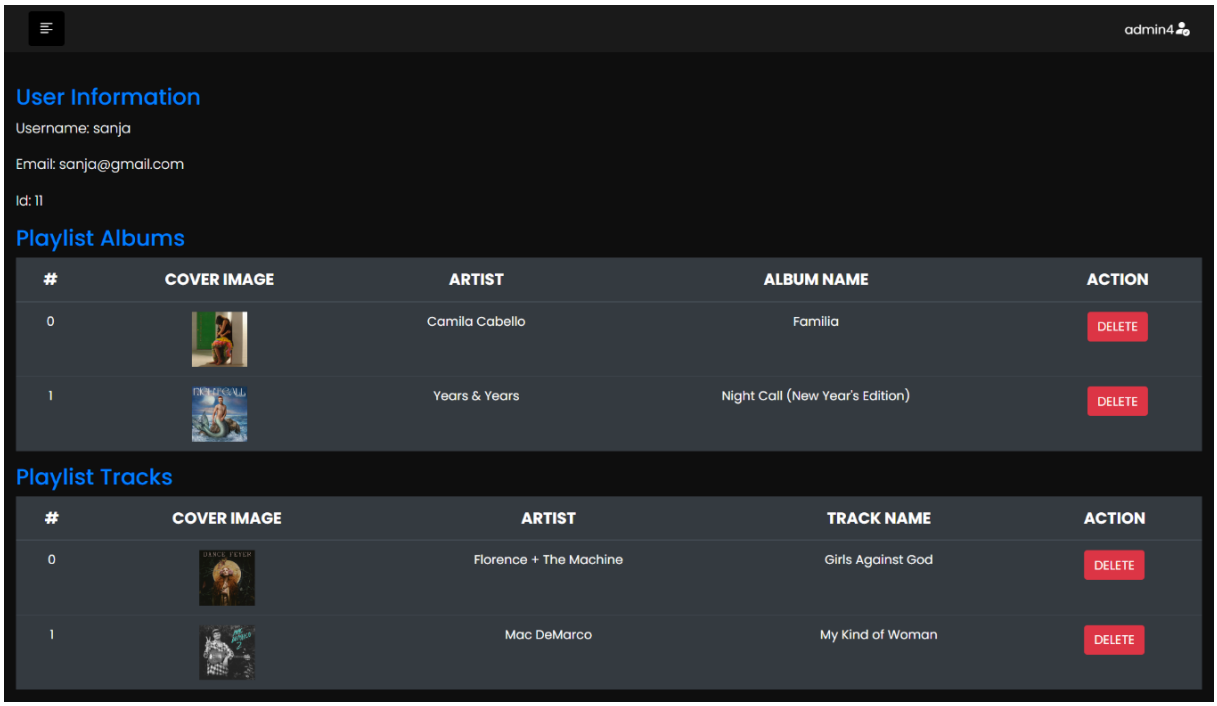
Programski kod 4.2. *Prikaz rute za brisanje korisnika iz tablice users.*

Prethodno navedene funkcionalnosti za verifikaciju administratora i brisanje korisnika iz baze podataka mogu se ostvariti preko administratorskog panela koji je prikazan slikom 4.3.



Slika 4.3. *Prikaz administratorskog panela.*

Još jedna od funkcionalnosti, koja nije prethodno spomenuta, ali se može vidjeti iz slike 4.3. jest prikaz informacija o određenom korisniku. Ova funkcionalnost može se postići klikom na gumb *INFO.* Pritiskom na taj gumb, administratoru se otvara nova stranica koja sadrži podatke o izabranom korisniku. Na toj stranici prikazani su podatci poput korisničkog imena, adrese elektroničke pošte, identifikacijske oznake korisnika te lista za reprodukciju glazbe koje uključuju liste albuma i pjesama. Administrator ima mogućnost brisanja svake pjesme i albuma koji je spremljen u korisnikovu listu za reprodukciju. Slikom 4.4. prikazani su ranije spomenuti podatci o izabranom korisniku.



Slika 4.4. *Prikaz podataka izabranog korisnika koji se dobiva pritiskom na gumb INFO.*

Pritiskom na gumb *DELETE* administrator ima mogućnost brisanja izabranog albuma ili pjesme s korisnikove liste za reprodukciju, a samim time i iz tablica *playlist_albums* i *playlist_tracks*. Programskim kodom 4.3., koji se nalazi u nastavku, prikazana je ruta za brisanje pjesme. Pomoću naredbe *DELETE FROM* izabire se tablica iz koje se brišu podatci. Naredbom *WHERE* točno se odabiru podatci iz tablice koji će biti izbrisani. Ako je pjesma uspješno obrisana iz tablice u terminalu se prikazuje poruka o uspješnom brisanju pjesme, a u suprotnom se prikazuje pogreška.

```
router.post('/dashboard_admin/user/:userId/delete_track/:trackId', function
(req, res) {
    const trackId = req.params.trackId;
    const userId = req.params.userId;

    db.query("DELETE FROM playlist_tracks WHERE id = ?", [trackId], (error,
result) => {
        if (error) throw error;

        console.log(`Track with ID ${trackId} deleted successfully`);

        res.redirect('/admin/dashboard_admin/user/' + userId);
    });
});
```

Programski kod 4.3. *Ruta za brisanje pjesme iz tablice playlist_tracks.*

Osim pregleda podataka o korisnicima i popisa administratora, verificirani administrator ima mogućnost pregleda izvođača koji se nalaze na Spotify platformi. Prethodno navedeno prikazano je slikom 4.5.

Table of Artists

ID	Name	Popularity
3kYFawNQVZ00FQbgs4rVBe	Artist Vs Poet	42
2hRgsNuAo7lhUqmrl25WD5	The Artist Ren	35
0KlSW3j5fUEfej47FOrBMr	Artists Of Then, Now & Forever	37
1JOdUvDAzNy3L37rZ4Nigr	pewdiepie	42
54nlhVtTKUjoVNSK8cjEzk	Arti Party	40
4QXKOrTZGGRM5qaDBU4ZVD	Artistic Samurai	35
6e2ZpZDumV9AabK9rxplUg	Arta Porting	49
4cS0lGolwn4vOj3uzkiOrK	Artists for Puerto Rico	29
4gvjmrtzydbMpyJaXUtwvP	Addison Rae	34
26kmQCgGLCe2XEeUhycGhD	ACTORMUSIC	34
1ZwdS5xdxEREPySFridCfh	2Pac	78
66CXWjxzNUsdJxJ2JdwvnR	Ariana Grande	87
3Pwl7aWPHoK3Enm59wt7M2	ARTAN	50
4y5yS9t228r4iyTqHaagmc	Artist	0
1cyPW4frKbhmtMQ8XVLsMN	Artonym	40
44NX2ffIYHr6D4n7RaZF7A	Van Morrison	70
6eUKZXaKkcviH0Ku9w2n3V	Ed Sheeran	88
0WjcdSKSTiNtfM4uQDlPDm	Artists for Haiti	28
6HhnhnxLsowYuuejvku0Bz	Lartiste	61
2abFwLvm5vnUGT7eNtLEXs	Know ART	30
6gjAbs229sNHhSskJpQ2mG	Artist Unknown	20
3NChzMpu9exTINPiqUQ2DE	Thrice	54
6gPKjPlXbBBnuLyLEq79Sz	Arta	46

Slika 4.5. *Tablica s popisom izvođača.*

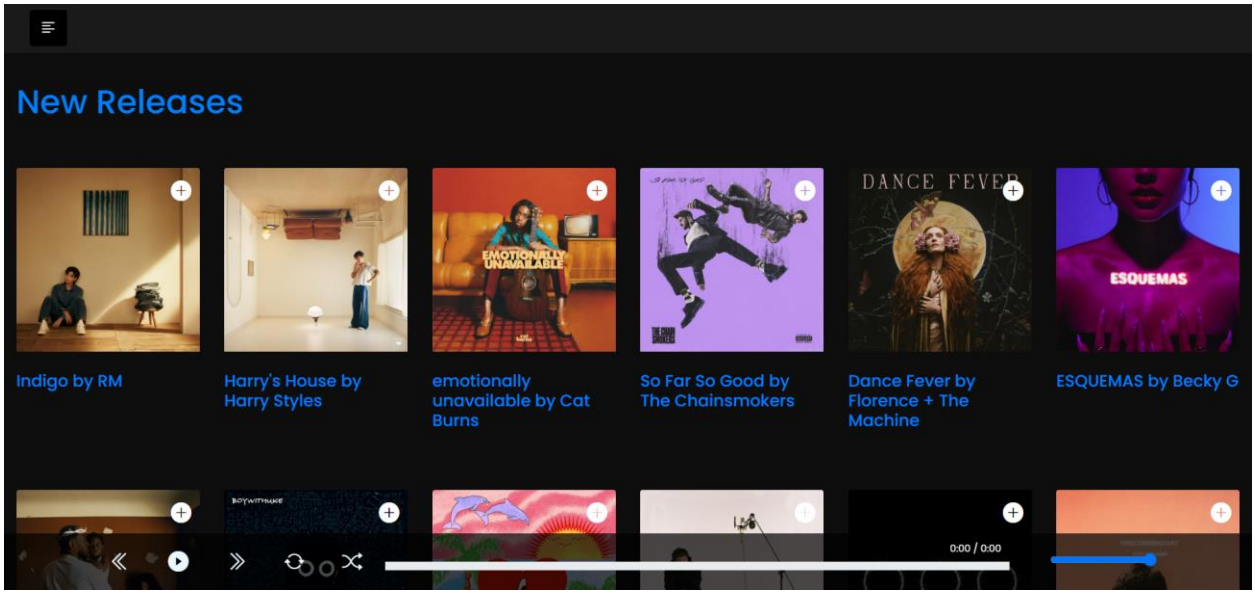
Programskim kodom 4.4. prikazano je korištenje Spotify API sučelja za pretragu izvođača. Ovaj programski kod omogućuje verificiranom administratoru mogućnost pregleda dobivenih izvođača. Prva linija programskog koda šalje zahtjev prema Spotify API sučelju za pretragu izvođača na temelju zadane ključne riječi *artist*. Zatim se rezultat pretrage izvođača sprema u varijablu *artists*. Na taj način, administrator može dobiti popis izvođača.

```
const artistsResponse = await spotifyApi.searchArtists('artist', { limit: 50
});
const artists = artistsResponse.body.artists.items;
```

Programski kod 4.4. *Dohvaćanje popisa izvođača uz pomoć Spotify API sučelja.*

4.2. Korisnički dio aplikacije

Početna stranica predstavlja korisničko sučelje koje prikazuje najnovije albume sa Spotify platforme. Na slici 4.6. prikazan je izgled početne stranice, na kojoj su prikazani najnoviji dostupni albumi. Pritiskom na određeni album, ikonu albuma ili njegovo ime, korisnik ulazi u sam album, te ima pregled na sve pjesme koje se nalaze unutar njega i dostupne su za slušanje.



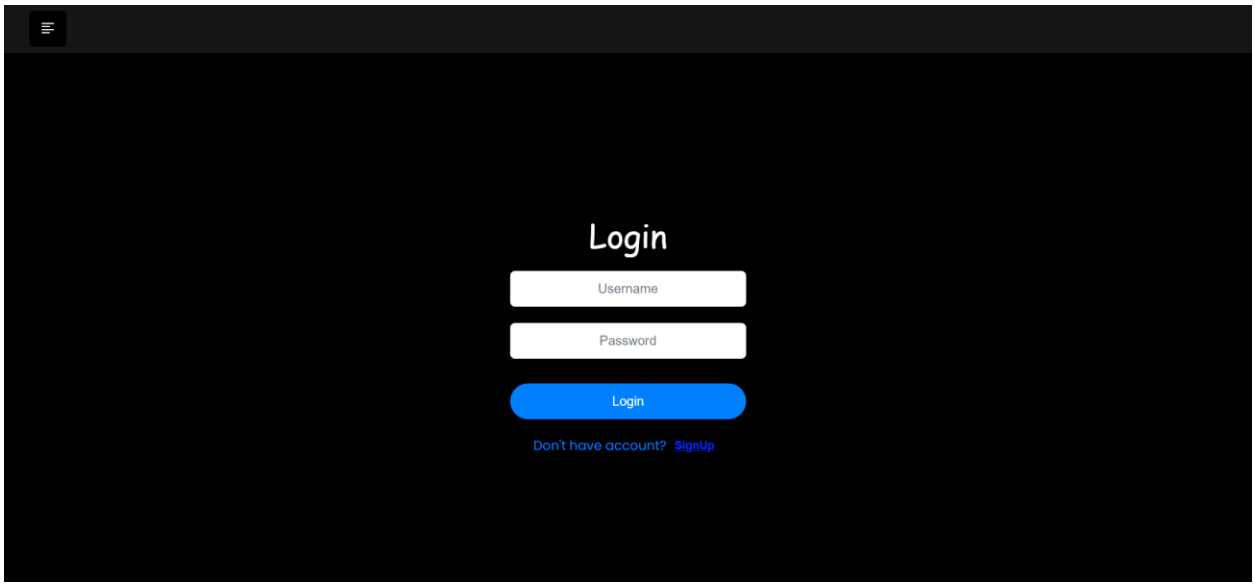
Slika 4.6. *Izgled početne stranice [15].*

Spotify aplikacijsko programsko sučelje (engl. *Spotify Application Programming Interface – API*) koristi se za dohvaćanje albuma sa Spotify platforme. Prikuplja najnovije informacije o albumima izravno s platforme. Spotify API usluga je koja omogućuje pristup različitim glazbenim sadržajima, uključujući pjesme, albume i izvođače. Programski kod 4.5. prikazuje dohvaćanje najnovijih albuma i pjesama pomoću Spotify API sučelja. Prvo se koristi metoda *clientCredentialsGrant()* za autentifikaciju i dobivanje pristupnog tokena. Zatim se dobiveni pristupni token postavlja na Spotify API objekt pomoću metode *setAccessToken()*. Nakon postavljanja pristupnog tokena, poziva se metoda *getNewReleases()* koja omogućuje dohvaćanje novih albuma. Nakon uspješnog dohvaćanja novih izdanja, rezultat se obrađuje u *.then()* bloku. Lista albuma se sprema u varijablu *albums*, a zatim se prolazi kroz svaki album pomoću metode *map()*. Korištenjem metode *map()*, za svaki album, stvara se nova verzija albuma s dodanom varijablom *imageUrl* koja sadrži URL slike albuma.

```
  spotifyApi.clientCredentialsGrant().then(
    (data) => {
      spotifyApi.setAccessToken(data.body['access_token']);

      spotifyApi.getNewReleases({ limit: 50, offset: 0, country: 'HR',
include_groups: 'album,single' })
        .then((data) => {
          const albums = data.body.albums.items;

          const albumsWithImages = albums.map(album => {
            const imageUrl = album.images.length > 0 ? album.images[0].url :
null;
            return {
              ...album,
              imageUrl: imageUrl
            };
          });

          res.render('index', { albums: albumsWithImages, loggedIn:
req.session.loggedIn, username: username });
        })
        .catch((err) => {
          console.log('Error occurred while getting new releases:', err);
          res.render('error');
        });
    },
    (err) => {
      console.log('Error retrieving access token:', err);
      res.render('error');
    }
  );
```

Programski kod 4.5. *Programski kod za dohvaćanje najnovijih albuma uz pomoć Spotify API sučelja.*

Na navigacijskoj traci aplikacije, u gornjem lijevom kutu, nalazi se ikona koja ima funkcionalnost otvaranja bočne navigacijske trake. Ova traka korisniku pruža različite opcije među kojima su i registracija i prijava u sustav, ukoliko korisnik već nije prijavljen. Na slici 4.7. prikazana je forma za prijavu korisnika. Ova forma omogućuje korisniku da unese svoje podatke i prijavi se u aplikaciju kako bi dobio pristup svim njezinim funkcionalnostima. U slučaju da korisnik još uvijek nije prijavljen u sustav, aplikacija mu pruža mogućnost registracije putem opcije *SignUp* koja se nalazi ispod forme za prijavu. Klikom na tu opciju korisnika se preusmjerava na stranicu za registraciju gdje može unijeti svoje podatke i stvoriti korisnički račun.



Slika 4.7. *Izgled forme za prijavu korisnika.*

Kada se korisnik uspješno prijavi u aplikaciju, omogućen mu je pristup dodatnim funkcionalnostima. Jedna od dodatnih mogućnosti je pretraživanje različitih glazbenih naslova. Ova funkcionalnost omogućava korisniku da pronađe svoje omiljene pjesme ili istraži nove glazbene naslove. U programskom kodu 4.6. prikazan je kod koji se koristi za rutu */search* prilikom primanja HTTP GET zahtjeva. Kada korisnik posjeti rutu, aplikacija dohvaća podatke za pretragu iz zahtjeva koji je upućen. Ti se podaci koriste za pretraživanje pjesama pomoću Spotify API sučelja. Nakon što se izvrši pretraga, rezultati se prikazuju korisniku. Rezultati sadrže informacije o pronađenim pjesmama, kao što su naziv pjesme i izvođač. Informacije se šalju u predložak koji je dizajniran za prikazivanje rezultata pretrage na korisničkom sučelju aplikacije.

```
router.get('/search', function (req, res) {
  var searchData = req.query.data;
  var username = req.session.username;
  spotifyApi.searchTracks(searchData)
      .then(function (data) {
          res.render('search', { results: data.body.tracks.items, loggedIn:
req.session.loggedIn, username: username });
      })
      .catch(function (err) {
          console.error(err);
          res.render('search', { results: [], loggedIn: req.session.loggedIn,
username: username  });
      });
});
```

Programski kod 4.6. *Prikaz rute za pretraživanje pjesama.*

Jedna od funkcionalnosti koja je dostupna prijavljenim korisnicima ove aplikacije je mogućnost stvaranja personaliziranih lista za reprodukciju omiljenih pjesama. Ova funkcionalnost korisnicima omogućuje organiziranje njihovih omiljenih glazbenih zapisa na način koji olakšava pronalazak i reprodukciju željenih pjesama. Korisnici mogu iskoristiti ugrađenu tražilicu za pretraživanje i pronalaženje željenih naslova ili direktno dodavati pjesme na svoju listu za reprodukciju iz albuma. Korisniku je omogućeno dodavanje pjesama na listu za reprodukciju pritiskom na ikonu znaka plus koja se nalazi u gornjem desnom kutu naslovne slike albuma kao što je to prikazano na slici 4.8.

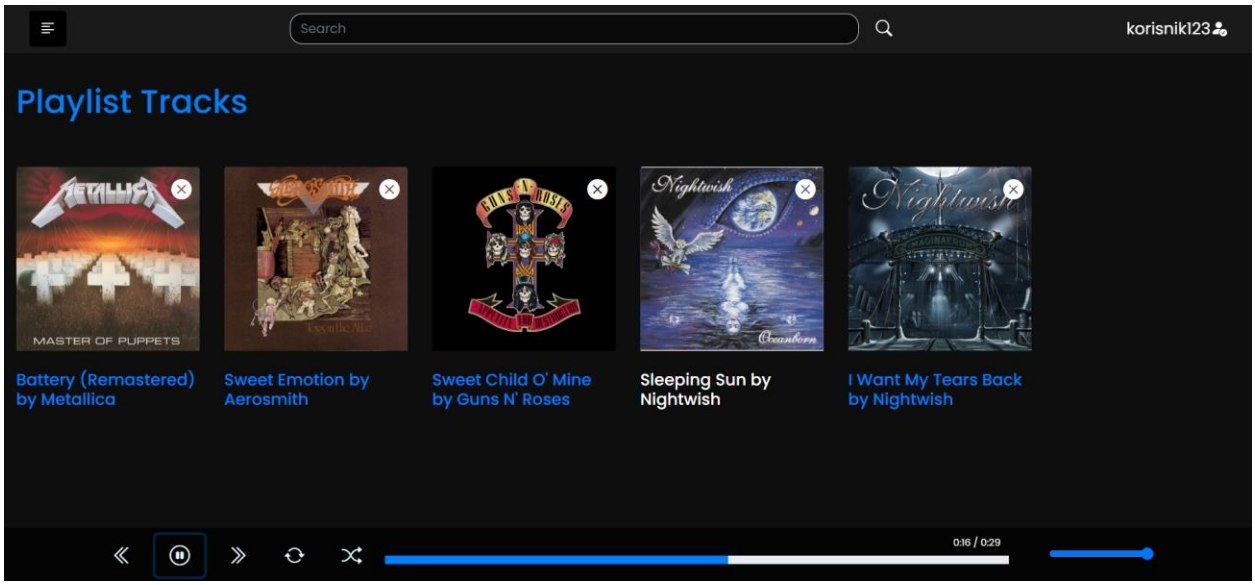
Slika 4.8. *Prikaz naslovne slike, imena pjesme i izvođača [15].*

Kada korisnik odabere pjesmu koju želi dodati na svoju listu za reprodukciju, aplikacija provjerava postoji li već zapis te pjesme na korisnikovoj listi. Ova provjera izvodi se putem SQL upita nad MySQL bazom podataka. Ovaj kod prvo izvršava *SELECT* upit nad tablicom *playlist_tracks* kako bi provjerio postoji li već zapis s istom identifikacijskom oznakom pjesme i identifikacijskom oznakom korisnika. Ako upit pronađe rezultat, to znači da je pjesma već dodana na korisnikovu listu. U tom slučaju, korisnik se preusmjerava na stranicu */playlist_tracks* koja prikazuje njegovu listu za reprodukciju. Ako upit ne pronađe rezultat, to znači da pjesma nije još dodana na listu. Tada se izvršava *INSERT* upit koji dodaje novi zapis u tablicu *playlist_tracks* s informacijama o pjesmi, kao što su identifikacijska oznaka korisnika, identifikacijska oznaka pjesme, naziv pjesme, izvođač, naziv albuma, URL slike naslovnice i URL za reprodukciju. Nakon što se *INSERT* upit uspješno izvrši, korisnik se preusmjerava na stranicu */playlist_tracks* kako bi vidio ažuriranu listu za reprodukciju s dodanom pjesmom. U nastavku je prikazan programski kod 4.7. koji implementira funkcionalnost opisanu u prethodnom tekstu.

```
        db.query(
          "SELECT * FROM playlist_tracks WHERE track_id = ? AND user_id = ?",
          [trackId, userId],
          (error, result) => {
            if (error) {
              throw error;
            }

            if (result.length > 0) {
              res.redirect("/playlist_tracks");
            } else {
              db.query(
                "INSERT INTO playlist_tracks (user_id, track_id, track_name,
artist, album_name, cover_image_url, preview_url) VALUES (?, ?, ?, ?, ?, ?,
?)",
                [userId, trackId, trackName, artist, albumName, imageUrl,
previewUrl],
                (error, result) => {
                  if (error) {
                    throw error;
                  }
                  res.redirect("/playlist_tracks");
                }
              );
            }
          }
        );
```

Programski kod 4.7. *Pohranjivanje pjesama u tablicu playlist_tracks.*

Na slici 4.9. prikazan je primjer korisničkog sučelja za upravljanje listama za reprodukciju. Na slici je prikazana lista pjesama za reprodukciju koja je stvorena od strane korisnika. Svaka pjesma, ispod naslovnice albuma, sadrži informacije poput naziva pjesme i izvođača. Korisnik ima mogućnost reproducirati pjesme s liste. Na slici je prikazan primjer pjesme koja se trenutno reproducira, a na kontrolnoj traci nalaze se gumbi za pauziranje, preskakanje na sljedeću ili prethodnu pjesmu, kao i mogućnosti ponavljanja pjesme i nasumične reprodukcije. Također, korisnik ima kontrolu nad reprodukcijom pjesme, što uključuje mogućnost premotavanja unaprijed ili unatrag i podešavanje glasnoće. Važno je napomenuti da, s obzirom na ograničenje Spotify API sučelja koje omogućava samo 30 sekundi reprodukcije, korisnik može slušati samo 30 sekundi odabrane pjesme.



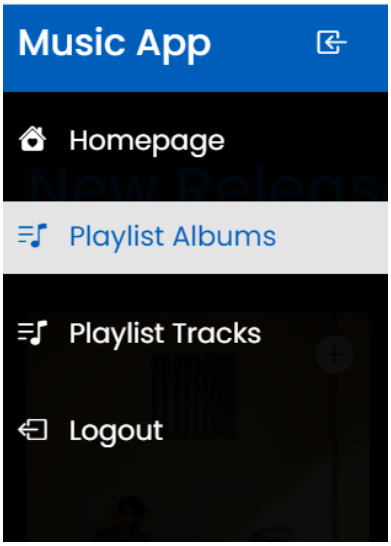
Slika 4.9. *Prikaz liste za reprodukciju pjesama [15].*

U gornjem desnom kutu naslovnice albuma nalazi se ikona znaka x, koja omogućava korisniku uklanjanje pjesme s liste za reprodukciju. Kada korisnik pritisne na ikonu za uklanjanje pjesme, aplikacija automatski osvježava web stranicu kako bi se prikazala lista bez uklonjene pjesme. Na taj način korisnik ima uvid da je pjesma uklonjena bez potrebe za ručnim osvježavanjem ili ponovnim učitavanjem stranice. Osim uklanjanja s web stranice, pjesma će biti trajno izbrisana iz baze podataka. Aplikacija će izvršiti odgovarajući SQL upit nad MySQL bazom podataka kako bi uklonila zapis o toj pjesmi iz tablice *playlist_tracks*. U programskom kodu 4.8. prikazuje se izvršavanje SQL upita *DELETE* nad tablicom *playlist_tracks* u bazi podataka. Upit briše zapis iz tablice koji ima *track_id* jednak *trackId* i *user_id* jednak *userId*. Ako se upit uspješno izvrši, korisnik će biti preusmjeren na stranicu */playlist_tracks*.

```
db.query(
  "DELETE FROM playlist_tracks WHERE track_id = ? AND user_id = ?",
  [trackId, userId],
  (error, result) => {
    if (error)
      throw error;
    res.redirect("/playlist_tracks");
  }
);
```

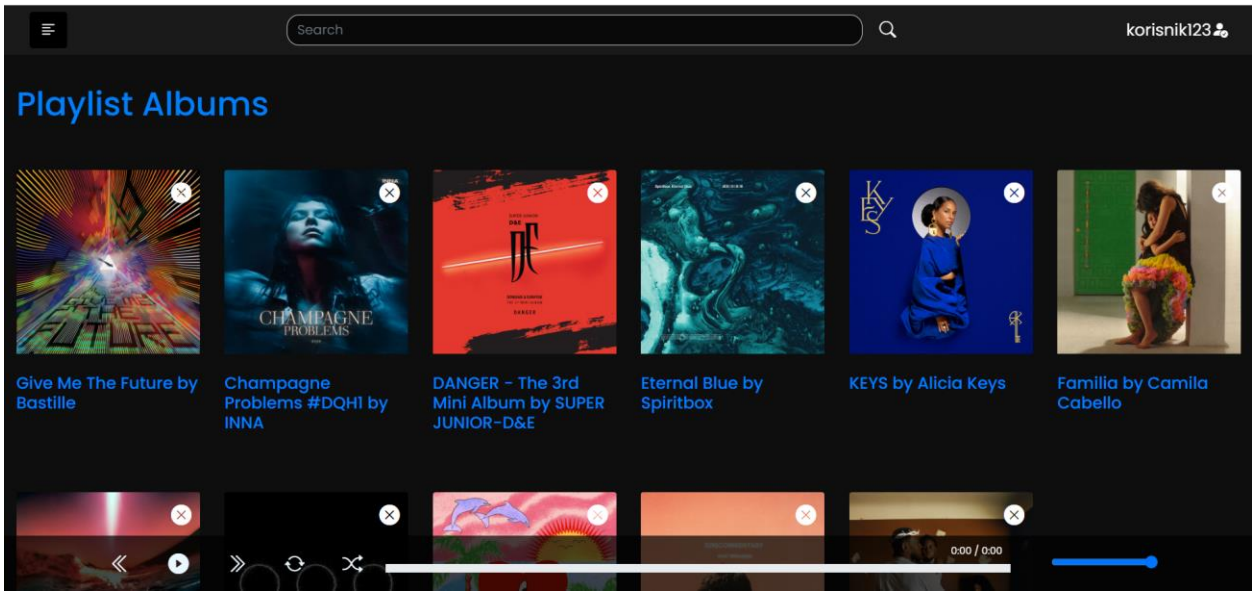
Programski kod 4.8. *Uklanjanje pjesme iz tablice playlist_tracks.*

Korisnicima se, osim spremanja omiljenih pjesama, omogućuje i spremanje njihovih najdražih albuma na jednom mjestu. Do spremljenih albuma moguće je doći putem bočne navigacijske trake pritiskom na *Playlist Albums* kao što se može vidjeti iz slike 4.10.



Slika 4.10. *Prikaz bočne navigacijske trake kada je korisnik prijavljen u aplikaciju.*

Kada korisnik pristupi svojoj listi za reprodukciju albuma, može vidjeti prikaz svih albuma koje je pohranio. Korisnik ima mogućnost ulaska u određeni album, i slušanja svih pjesama koje se nalaze na tom albumu. Također, ima mogućnost uklanjanja određenog albuma iz liste za reprodukciju tako da pritisne ikonu *x* koja se nalazi u gornjem desnom kutu naslovne fotografije albuma što se može vidjeti sa slike 4.11.



Slika 4.11. *Prikaz liste za reprodukciju albuma [15].*

Pohrana albuma na listu za reprodukciju odvija se na isti način kao i pohrana pjesama. Programskim kodom 4.9. prikazano je pohranjivanje albuma u tablicu *playlist_albums*.

```
        db.query(
          "SELECT * FROM playlist_albums WHERE album_id = ? AND user_id = ?",
          [albumId, userId],
          (error, result) => {
            if (error)
              throw error;
            if (result.length > 0) {
              res.redirect("/playlist_albums");
            } else {
              db.query(
                "INSERT INTO playlist_albums (album_id, user_id, artist,
album_name, cover_image_url) VALUES (?, ?, ?, ?, ?)",
                [albumId, userId, artist, albumName, imageUrl],
                (error, result) => {
                  if (error) throw error;
                  res.redirect("/playlist_albums");
                }
              );
            }
          }
        );
```

Programski kod 4.9. *Pohrana albuma u tablicu playlist_albums.*

Prvo se izvršava upit *SELECT* nad tablicom *playlist_albums* s ciljem provjere postoje li već unosi za određenu identifikaciju oznaku albuma i korisničku identifikacijsku oznaku. Parametri *albumId* i *userId* koriste se za filtriranje rezultata upita. Ako album već postoji u tablici, korisnik se preusmjerava na rutu */playlist_albums* kako bi mogao vidjeti svoje spremljene albume. Ukoliko album još ne postoji u tablici izvršava se *INSERT INTO* koji dodaje novi unos u tablicu. Parametri *albumId*, *userId*, *artist*, *albumName* i *imageUrl* koriste se za unos odgovarajućih vrijednosti u tablicu.

5. ZAKLJUČAK

U okviru ovog završnog rada realizirana je web aplikacija za reprodukciju glazbe koja korisniku aplikacije pruža lak pristup raznim glazbenim sadržajima. U radu je dan prikaz već postojećih web aplikacija koje nude slična rješenja.

Aplikacija je stvarana u integriranom razvojnom okruženju pod nazivom Visual Studio Code. Za izradu ove aplikacije bilo je potrebno prisjetiti se nekolicine, već poznatih, tehnologija naučenih na fakultetu, poput HTML i CSS jezika, Bootstrap okvira, MySQL jezika. Također je bilo potrebno i upoznati se s nekim novim tehnologijama poput XAMPP paketa, Node.js platforme i Express.js okvira. Sve korištene tehnologije opisane su u radu. Aplikacija je podijeljena na dva osnovna segmenta, a to su administratorski dio aplikacije i korisnički dio aplikacije. Administratoru je omogućen pregled svih korisnika unutar baze i može njima upravljati, dok korisnici imaju uvid u sučelje koje im omogućuje pretragu glazbenog sadržaja koji se povlači sa Spotify API-ja.

LITERATURA

[1] Mixmag Adria: Kako smo sve slušali glazbu [online], dostupno na: https://mixmagadria.com/feature/kako-smo-sve-slusali-glazbu [28.5.2023.]

[2] Software: 10 Best Music Streaming Services [online], dostupno na: https://www.softwaretestinghelp.com/best-music-streaming-services/#5_YouTube_Music [30.5.2023.]

[3] Spotify: Spotify, dostupno na https://open.spotify.com/ [30.5.2023.]

[4] Youtube Music: Youtube Music, dostupno na https://music.youtube.com/ [30.5.2023.]

[5] Apple Music: Apple Music, dostupno na https://music.apple.com/us/browse [30.5.2023.]

[6] Deezer: Deezer, dostupno na https://www.deezer.com/hr/offers [30.5.2023.]

[7] Microsoft: Upotreba proširenja Visual Studio Code [online], dostupno na: https://learn.microsoft.com/hr-hr/power-apps/maker/portals/vs-code-extension [30.5.2023.]

[8] W3Schools: HTML Introduction [online], dostupno na: https://www.w3schools.com/html/html_intro.asp [30.5.2023.]

[9] MDN Web Docs: HTML basic [online], dostupno na: https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/HTML_basics [30.5.2023.]

[10] MDN Web Docs: CSS basic [online], dostupno na: https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/CSS_basics [30.5.2023.]

[11] D. Herron, Node.js web razvoj, Kompjuter biblioteka, Beograd, 2020.

[12] W3Schools: JavaScript Intro [online], dostupno na: https://www.w3schools.com/js/tryit.asp?filename=tryjs_intro_lightbulb [30.5.2023.]

[13] J.Spurlock, Bootstrap, O'Reilly Media, Sebastopol, 2013.

[14] L.Welling, L.Thomson, Priručnik za MySQL, Mikro knjiga, Beograd, 2005.

[15] Spotify for Developers: Web API [online], dostupno na: https://developer.spotify.com/documentation/web-api/reference/get-an-album [18.8.2023.]

SAŽETAK

Cilj ovog završnog rada bio je stvoriti internet aplikaciju za reprodukciju glazbe. Aplikacija omogućava pregled glazbenog sadržaja koji se dobiva sa Spotify API-ja. Teorijski dio rada daje pregled sličnih rješenja, opis tehnologija korištenih u izradi aplikacije, te opisuje slijed razvoja aplikacije. Praktični dio rada sastoji se od izrade same aplikacije. Izgled i struktura aplikacije rađeni su pomoću opisnih jezika HTML i CSS, dok je funkcionalni dio aplikacije napravljen u JavaScript skriptnom jeziku. Za izradu baze podataka korišten je MySQL programski jezik.

Ključne riječi: baza podataka, CSS, glazba, HTML, internet aplikacija, JavaScript.

ABSTRACT

The aim of this final paper was to create an internet application for music playback. The application allows users to browse music content obtained from the Spotify API. The theoretical part of the project provides an overview of similar solutions, describes the technologies used in the development of the application, and outlines the development process. The practical part of the project consists of building the actual application. The design and structure of the application were created using descriptive languages HTML and CSS, while the functional part of the application was developed using the JavaScript scripting language. MySQL programming language was used for database creation.

Keywords: CSS, database, HTML, internet application, JavaScript, music.

ŽIVOTOPIS

Nancy Matijaš rođena je 25.09.2001. u Osijeku. Pohađala Osnovnu školu Vladimira Nazora Čepin. Nakon završetka osnovne škole, 2016. godine, upisuje III. Gimnaziju Osijek. Svoje obrazovanje nastavlja na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku gdje 2020. godine upisuje preddiplomski sveučilišni studij Računarstva, smjer Programsko inženjerstvo.