Zbornik radova Fakulteta tehničkih nauka, Novi Sad

**UDK: 004.9**
**DOI: https://doi.org/10.24867/07BE11Jovicic**

# RAZVOJ VEB APLIKACIJE TRIO MJUZIK

# DEVELOPMENT OF A WEB APPLICATION TRIO MUSIX

Vladimir Jovičić, *Fakultet tehničkih nauka, Novi Sad*

**Oblast – RAČUNARSTVO I AUTOMATIKA**

**Kratak sadržaj –** *Zadatak rada predstavlja analizu veb aplikacije koja korisnicima pruža usluge trgovine preko interneta muzičkih instrumenata i opreme, zakazivanje i gledanje kurseva kao i servis instrumenata. Biće posmatrani aspekti programera i korisnika, kako implementacije tako i dizajn za krajnjeg korisnika kako bi on brzo pronašao ono što veb stranica nudi.*

**Ključne reči:** *Prodavnica, servis, kursevi*

**Abstract** – *The task of the paper is to analyze a web application that provides users with online music instruments and equipment trading services, to schedule and watch courses, and to service instruments. Aspects of developers and users will be looked at, both implementation and design for the end user to quickly find what the website has to offer.*

**Keywords:** *Shop,, service, courses*

## 1. UVOD

U današnje vreme, *web* tehnologije i *web* programiranje su zastupljeni u velikoj meri. Moderne *web* aplikacije se najčešće kreiraju tako što se cela aplikacija razdvoji u dva dela : klijentska strana i serverska [1] strana. Klijentska strana predstavlja ono što korisnik vidi i čime upravlja, a serverska strana predstavlja ono što se dešava "u pozadini" tj. obrada zahteva koju korisnik pošalje. Klijentka strana se najčešće naziva i *frontend*, a serverska strana se najčešće naziva i *backend*. Serverska strana uglavnom upravlja sa bazom podataka, izvršava određene operacije nad istom, i šalje klijentskoj strani povratnu informaciju o izvršenim operacijama nad bazom.

Da bi se to olakšalo i da bi se omogudila dinamička web stranica koja bi reflektovala korisničke unose uveden je *CGI* (*Common Gateway Interface*). To je standard koji je uveden za interfejsing spoljnih aplikacija sa web serverima. *CGI* je imao tu manu što je svaki zahtev morao biti pokrenut kao poseban proces.

Velika većina programskih jezika pružaju gotove biblioteke za razne zadatke, ali isto tako, vedina zahteva uključivanje dodatnih biblioteka za rad sa web aplikacijama i npr. pisanje *HTML* koda.

**NAPOMENA:**
**Ovaj rad proistekao je iz master rada čiji mentor je bio dr Dragan Ivetić, red. prof.**

*Web* aplikacije se danas sve više i više dele u po raznim kriterijumima u zavisnosti od njihove namene. Na osnovu toga, neke od vrsta *web* aplikacija mogu biti: *Online* kupovina/prodaja, društvene mreže, sajtovi za edukaciju i gledanje kurseva, sajtovi za zabavu, bankarstvo… Takođe je moguća i kombinacija bilo koje dve predhodno napomenutih vrsta. Na primer sajtovi za *online* kupovinu su usko povezani sa sajtovima za bankarstvo jer se plaćanje vrši uglavnom preko neke centralne banke.

## 2. OPIS REŠAVANOG PROBLEMA

Mnoge modernije i razvijenije *web* aplikacije se kombinuju često i sa više od 2 vrste pri čemu nastaju multi namenske *web* aplikacije. Sajtovi društvenih mreža pored toga što nude kontakt sa određenim "prijateljima", nude i dodatne forume za oglase, video *chat*, *entertainment*, pregled novosti (kako "prijatelja" tako i vesti u globalu). Time je sajt obezbedio sebi širok spetkar korisnika što znači da što više ljudi poseduje sajt na dnevnoj bazi. Ako je sajt napravljen da bude "*user friendly*", što podrazumeva da je jednostavan za upotrebu svim posetiocima, čak i ljudima koji se malo manje razumeju u tehnologiju, tada je sajt sebi osigurao visoku posedenost na dnevnom nivou.

### 2.1. *Web* aplikacije Trio „Trio Music"

Web sajt za *online* prodaju muzičkih instrumenata sadrži u sebi (kao i svaka *online* prodavnica) izbor svih artikala koje korisnik može da kupi. Uz svaki artikal uglavnom se prikazuje i slika, cena, marka (ako je npr. u pitanju gitara onda se navodi i koja je debljina žica koje dolaze uz nju, boja, vrsta drveta od koje je napravljena i slično). Pretraga kod ovakvih sajtova mora da bude veoma detaljna ali i jednostavna.

Svaki instrument (kao i sve ostalo) je sporno kvarovima. Ponekad se kvar može otkloniti i to znatno jeftinije košta nego da se kupuje nov artikal. Za neke sitnije kvarove poželjno je imati kratke opise koje bi korisnik sam mogao da primeni na svom kvaru kako bi ga brzo i efikasno uklonio ali ta opcija je veoma rizična jer iz neiskustva može slediti nanošenje još većeg kvara.

Postoje i ljudi koji žele da nauče da sviraju a ne znaju kako početi, ili oni koji znaju svirati a žele da nauče još neke dodatne tehnike kako bi se usavršili. Nekima je lakše da gledaju lekcije direktno sa svojih uređaja na sajtu, a nekima je lakše interaktivno sa mentorom da nauče nešto novo pri čemu sajt nudi zakazivanje kurseva (individualno ili grupno) jer možda smatra da de svoje veštine savladati na taj način brže i/ili bolje.

## 3. OPIS KORIŠĆENIH TEHNOLOGIJA

Sa developerske strane gledano postoji više načina da se implementira sajt za *online* kupovinu/servis i kurseve sviranja muzičkih instrumenata. Treba voditi računa o nekim osnovnim pravilima kako bi se olakšala ne samo implementacija i razvoj ved i samo održavanje *web* aplikacije što podrazumeva konstantno menjanje izgleda/sadržaja stranice, načina upotrebe i slično. Jer ako se sajt koristi frekventno uvek de se nadi zamerke i određeni *bug*-ovi koji se mogu (i trebaju) popraviti.

### 3.1 *Backend*

Zbog komplikacije ovakve aplikacije, neretko se dešava da više ljudi radi na njoj. Često su poslovi podeljeni na celine *stack*-ova. To znači da jedna ili više osoba rade *frontend*, jedna ili više osoba rade *backend [2]*, jedna ili više osoba se bavi bazom podataka. Takođe je mogude imati i testere koji glume aktivnog korisnika nad gotovim, ali još ne objavljenom aplikacijom, i tako traže mane koji bi se mogle popraviti pre same produkcije

#### 3.1.1 *SOAP* princip

*SOAP* (*Simple Object Access Protocol*) [3] je komunikacioni protokol, nezavisan od platforme, baziran na *XML*-u koji se koristi za razmenu informacije između aplikacije preko *HTTP* protokola. Razvijen je kako bi omogućio jednostavnu komunikaciju tekstualnim sadržajem preko *HTTP* komunikacionog protokola koji je prilagođen upravo razmeni tekstualnih sadržaja. Protokol je nezavisan od programskog jezika, platforme i jednostavno proširiv.

*SOAP* protokol omogućuje komunikaciju između aplikacija koje rade na različitim operativnim sistemima i različitim tehnologijama. Jedini zahtev koji moraju zadovoljiti je da u komunikaciji mogu koristiti *HTTP* protokol. Aplikacije razmenjuju poruke dogovorenog formata.

#### 3.1.2 *REST* princip

*REST* (*REpresentational State Transfer*) [4] arhitekturalni stil je razvijen od strane *Technical Architecture Group (TAG)* paralelno sa *HTTP* 1.1. Ovaj stil je dobio najveću primenu na *World Wide Webu*. *REST* arhitektura se tipično sastoji od klijenta i servera. Klijent inicira zahtev serveru, server procesira zahtev i vrada odgovor klijentu. Glavni ciljevi REST-a su skalabilnost, generalizovanost *interface*-a i nezavisan broj komponenti.

Korišdenje *HTTP* metoda podrazumeva da se pri upotrebi *REST*-a koriste standardni zahtevi *HTTP* metode, a to su: kreiranje i čuvanje (***C**reate*), učitavanje postojećeg (***R**ead*), menjanje postojećeg (***U**pdate*) i brisanje (***D**elete*). Ove 4 metode se zovu (na osnovu početnih slova na engleskom) i *CRUD* metode.

Prenos resursa se najčešće vrši preko *JSON* objekata što predstavlja jedan od lakših tekstualnih otvorenih standarda dizajniran za čitljivu razmenu podataka. Ekstenzija datoteke s podacima u *JSON*-ovom formatu je *.json*, dok je meta oznaka (*MIME* format) application/json. *JSON* u sebi može imati i druge *JSON* objekte.

```
{
    "ime":"Vladimir",
    "prezime":"Jovicic",
    "god":24,
    "struka":"Master inzenjer"
}
```

Slika 3.1.2.1 Prime *JSON* objekta

### 3.2 *Frontend*

*Frontend [5]* se bavi onim delom *web* aplikacije koji korisnik vidi. Ono što je prikazano u samom pretraživaču. On se može podeliti u 3 osnovna dela: Statički i dinamički sadržaj koji se radi uz pomod *HTML*-a (*hyper text makrup language*), upravljanje funkcionalnostima (*JavaScript* ili neka njegova biblioteka ili *framework*) i dizajniranje, opisivanje pozicija i pokreta sadržaja pomodu *CSS*-a (*Cascading Style Sheet*).

#### 3.2.1 *HTML*

*HTML* (*hyper text makrup language*) [6] je opisni jezik specijalno namenjen opisu *web* stranica. Pomoću njega se jednostavno mogu odvojiti elementi kao što su naslovi, paragrafi, citati i slično. Pored toga, u *HTML* standard su ugrađeni elementi koji detaljnije opisuju sam dokument kao što su kratak opis dokumenta, ključne reči, podaci o autoru i slično. Ovi podaci su opštepoznati kao meta podaci i jasno su odvojeni od sadržaja dokumenta. Svi *HTML* dokumenti bi trebalo da počinju sa definicijom tipa dokumenta *DTD* (*Document Type Definition*) koji pregledaču definiše po kom standardu je dokument pisan.

#### 3.2.2 *CSS*

*CSS* (*Cascading Style Sheets*) je jezik formatiranja pomoću kog se definiše izgled elemenata *web* stranice. Prvobitno, *HTML* je služio da definiše kompletan izgled, strukturu i sadržaj *web* stranice, ali je od verzije 4.0 *HTML*-a uveden *CSS* koji bi definisao konkretan izgled, dok je *HTML* ostao u funkciji definisanja strukture i sadržaja. *CSS* je u određenoj formi postojao još u začecima *SGML*-a 1970-ih godina. Kako je *HTML* postajao komplikovaniji, davao je sve više mogućnosti za definiciju izgleda elemenata, ali je istovremeno postajao nečitljiviji i teži za održavanje. Različiti pretraživači su prikazivali dokumente na različite načine, i postojala je potreba za doslednom tehnikom definisanja prikaza elemenata na stranici.

#### 3.2.2 *JavaScript*

*Javascript* [8] je je dinamički, slabo tipiziran i interpretiran programski jezik visokog nivoa. Pored *HTML*-a i *CSS*-a, *JavaScript* je jedna od tri vodeće tehnologije za definisanje sadržaja na *webu*. Većina *web* sajtova koristi *Javascript* a svi moderni pretraživači ga podržavaju bez potrebe za instaliranjem dodataka. Kombinovan sa *HTML* jezikom i *CSS*-om *Javascript* čini *DHTML* (*Dynamic HTML*). *Javascript* je jezik zasnovan na prototipovima sa funkcijama prvog reda, što ga čini jezikom višestruke paradigme koji podržava objektno-orijentisani, imperativni i funkcionalni način programiranja.

## 4. OPIS REŠENJA PROBLEMA

Problem posmatran sam po sebi je pre svega izazovan sa developerske strane posmatrano. Potrebno je da se iskonfigurišu okruženja u kojima de se raditi kao i bazu podataka. Zbog svoje složene prirode i činjenice da de se sve razdvojiti na više pod-aplikacija, potrebno je svaku od njih posebno podesiti.

### 4.1 Glavna arhitektura

Glavna arhitektura aplikacije se sastoji iz 3 zasebna *frontend* dela i 3 zasebna *backend* dela i jednom zajedničkom bazom podataka. Glavna ideja je da svaki frontend deo "komunicira" sa svojim korespodentnim backend delom. Svaki *backend* deo je povezan sa istom bazom podataka što znači da svaki od njih ima zajedničku konfiguraciju pristupu bazi podataka. Time je omogudena podela resursa na sve delove aplikacije i ciljnim pozivanjem upita iz određenih delova dobijaju se određeni entiteti za daljnju obradu. U svakom od *backend* delova se nalazi određeni endpoint koji je potreban da se pozove kako bi se izvršila neka akcija sa obradom podataka (učitavanje iz baze, izvršavanje nekih proračuna…). To je ujedno i jedna od stvari koja konzolne aplikacije razlikuje od proceduralnih (konzolne se izvršavaju u jednom toku,

#### 4.1.1 Baza podataka

Zbog potencijalne velike komplikacije i opširnosti baze podataka, preporučeno je raditi u nekoj od relacionih baza podataka poput *MySQL [9]*. On predstavlja višenitni (*multithread*) je višekorisnički *SQL* sistem za upravljanje bazama podataka. Sistem radi kao server, obezbeđujudi višekorisnički interface za pristup bazi podataka.

### 4.2 Interfejs

*Interface* same *web* aplikacije je prilagođen standardizaciji dizajna modernih *web* aplikacija radi održavanja konzistentnosti. Pri ulasku na sajt, na samoj početnoj strani aplikacije, korisnik ima mogućnost da izabere jednu od 3 ponuđene usluge koje pruža aplikacija. Izbori treba da budu jasno definisani i nedvosmisleni, sve na jednom mestu uredno raspoređeni ili jedna ispod druge ili jedna iznad druge. Pored svega toga ima jedna poruka koja dočekuje korisnika sa dobrodošlicom. Boje treba da budu blage jačine da korisnika „ne udara" u oči kako bi mu lakše pala koncentracija. Izborom bilo koje od 3 opcije korisnik se prosleđuje na određeni deo aplikacije koji je izabrao.

#### 4.2.1 Zajednički interfejs

Da bi aplikacija bila konzistentna, potrebno je da sva 3 dela aplikacije podrže ista glavna svojstva da bi korisnik imao osećaj da je i dalje na istom sajtu. Neke od tih osobina su: Ista pozadina i boja pozadine mora da bude prisutna na svim delovima, isti font mora biti prisutan što podrazumeva istu veličinu, stil, boju i način pisanja, sva obaveštenja i sve poruke koje iskaču, kao i sve animacije moraju biti isto napravljene Pri svakom dolasku na neku od delova aplikacije, pojavljuje se dobrodošlica i u svakom trenutku treba da piše sekcije u kojoj se korisnik trenutno nalazi tako da on u svakom trenutku zna stanje aplikacije i ako se nalazi u onom delu u kome ne želi da bude, da lako može da pređe u sekciju aplikacije koja mu je potrebna.

#### 4.2.2 Prodavnica

Sekciju prodavnice korisnik posećuje kada je odlučio da želi da iz širokog asortimana izabere neki proizvod, vidi recenziju, pogleda da li odgovara cena i zatim putem online kupovine ili lično da kupi i preuzme taj proizvod.

##### 4.2.2.1 Početna stranica

Sama početna strana treba da ima dobrodošlicu za korisnika sa porukom velikim slovima ali ne preterano animirano jer ne treba da korisniku odvuče pažnju. Pošto početna strana treba samo da služi da dočeka korisnika, a smisao i zadatak ovog dela aplikacije je da korisnik kupi nešto, na njoj treba da se nalaze proizvodi koje sama prodavnica izdvaja iz ponude. Ti proizvodi su ili oni koji su na popustu, ili su pri kraju zaliha, ili je veoma popularno u zadnje vreme pa možda bi nešto od toga kupca zanimalo da kupi.

##### 4.2.2.2

Galerija u sebi sadrži skup slika koje predstavljaju izgled prodavnice. Slike su raspoređene po sortirane po datumu objavljivanja. Svakoj slici se može pristupiti u *fullscreen* modu kako se može videti u punoj veličini i moguda je navigacija slika u *fullscreen* modu prelaskom na predhodnu i slededu sliku. Time je omogudeno korisniku da ima uvid o tome kako izgleda sama prodavnica u kojoj želi da kupi stvari. Te slike mogu (ali ne bi trebale) da predstave i reklamiraju konkretne proizvode sa cenama. U bazi podataka se trebaju čuvati samo putanje do slika, ali ne i same slike.

##### 4.2.2.3 Kontakt

Neretka je situacija da korisnik ima neka pitanja vezana za neki konkretan artikl ili bilo šta vezano za prodavnicu koja možda nisu odgovorena na sajtu ili su mu neke stvari nejasne. Na *Contact* stranici ovog dela aplikacije nalaze se sve potrebne informacije kako bi korisnik kontaktirao vlasnike/radnike prodavnice.

##### 4.2.2.4 Recenzije

Svako ko je nov na web aplikaciji može biti skeptičan oko njenog korišćenja. Preporuka od strane drugih korisnika je uvek dobra stvar. Zbog toga je ovde uvedena sekcija *review* za recenzije koje predstavljaju ocene i komentare drugih korisnika koji su koristili sajt i njegove usluge. Svaki ulogovani korisnik može da ostavi komentar i da oceni ocenom od 1 do 5.

##### 4.2.2.5 Pretraga

Pretraga je jedan od najvažnijih delova samo aplikacije jer posredstvom nje, korisnik dolazi do željenog proizvoda. Napravljena je na više načina tako da korisnik može da bira onaj način koji je njemu optimalan: Klasična pretraga koja se sastoji iz jednog tekstualnog polja gde korisnik unosi izraz (jedna ili više reči) na osnovu kog se vrši pretraga, proširena pretraga koja nudi korisniku opciju da sam bira kriterijume pretrage. Nudi mu se iz izbora klasa i grupa opcije i kriterijume koje može da odredi i koje želi da uključi u pretragu.

##### 4.2.2.7 Kupovina putem aplikacije

Svaki korisnik ima svoju "korpu za kupovinu". Kada se odluči da želi da kupi neki proizvod on prvo mora da je

doda u korpu. Ta korpa sama po sebi može biti toliko komplikovana da se može razviti kao posebna aplikacija koja je preko mikroservisa povezana za glavnim *backend*-om. Nakon kupovina korisniku se obavezno prikazuje odgovarajude obaveštenje da je kupovina uspešno izvršena (ili ako je došlo do neke greške da ga obavesti o tome). Obavezno se korisniku šalje mejl sa celim izveštajem i računom o kupovini sa podacima o tome šta i kada je kupio, kako i koliko je platio.

#### 4.2.3 Kursevi

Deo aplikacije koji se bavi kursevima sadrži uputstvo za korisnika kako da dođe do nekog predavača koji bi ga naučio da svira neki instrument ili da savlada neku veštinu sviranja (*arpeggio, blast beat...*). Kao i svaki deo ove aplikacije, sastoji se od početne stranice koja sadrži dobrodošlicu i kratko uputstvo kako da dođe do predavača koji bi bio optimalan za njega.

##### 4.2.3.1 *Online* kursevi

Ukoliko se korisnik odlučio da gleda kurseve preko aplikacije (jer smatra da mu je tako lakše da nauči) poseduje ovaj segment aplikacije. Kursevi prestavljaju video klipove u kojima jedna ili više osoba objašnjavaju korisniku načine i korake koji su potrebni da bi savladao određenu tehniku. Mana ovog načina jeste to što korisnik nema nikakvu interakciju sa predavačem. Zbog toga ovi video klipovi moraju biti jasni i razumljivi.

##### 4.2.3.2 Uživo kursevi

Nekim korisnicima je lakše interaktivno da ih neko nauči nešto novo. Zbog toga, aplikacija nudi kurseve uživo sa predavačima koji se mogu rezervisati. Pretraga za *online* kurs i za rezervaciju predavača funkcionišu na sličan način samo što ovde još korisnik dodatno naglašava dane i vremenske opsege u nedelji kada on želi da ide na časove.

##### 4.2.3.3 Sekcija tutorijala

Aplikacija nudi posebnu sekciju gde korisnici mogu da gledaju besplatne video klipove koji im mogu pomodi oko nekih sitnijih stvari. Ovaj deo je tu da početnicima demonstrira neke proste stvari koje su opšte dostupne i da ih ovde lakše pronađu. Radi konzistentnosti, ovde takođe postoji pretraga na osnovu instrumenata i oblasti samo što rezultujući video klipovi nisu "zaključani" i svako ih može pogledati, oceniti i prokomentarisati.

#### 4.2.4 Servis

Nije retka stvar da se neki instrument može pokvariti, polomiti, vremenom istrošiti ili imati bilo kakav drugačiji kvar. Za takve situacije postoje majstori koji se bave takvim problemima odnosno njihovim rešavanjem. Definisati kvar nije lako objasniti preko bilo koje aplikacije pa je ovaj malo detaljnije napravljen kako bi korisnik najlakše identifikovao kvar. Nakon pretrage korisniku se nudi spisak majstora i osoba koji obavljaju popravku i otklanjanje kvarova.

## 5. ZAKLJUČAK

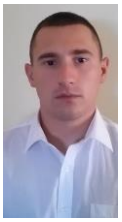
U današnjem modernom vremenu, sve više i više stvari su digitalizovane. Sve više i više stvari se obavljaju posredstvom mobilnih telefona, kompjutera ili drugih elektronskih uređaja. *Web* tehnologije su postale sve više i više zastupljenije jer ne zahtevaju nikakav dodatan napor pri instaliranju nekih programa i dodataka (osim rutinske instalacije u „3 klika"). Jako se lako koriste. Aplikacija „Trio Music" omogućava tri stvari koje ranije nisu mogle biti obavljene na jednom mestu na toliko prost način. Korisnik samim prijavljivanjem na sajt dobija mogućnost da pregleda artikle u samoj prodavnici što mu znatno skraćuje vreme da luta po raznim prodavnicama koje možda i nemaju ono što on traži ili mu cena ne odgovara ili je najbliža muzička prodavnica previše udaljena pa mu online kupovina omogudava i lakše i brže pladanje.

Ne mora da „kopa" po internetu da bi pronašao kurseve koji mu možda odgovaraju, a možda i ne. Ovde ima sve lepo na jednom mestu uz mogućnost uvida recenzija prethodnih korisnika. Ne mora da pretražuje silne stranice i kataloge kako bi pronašao servisera da otkloni neki kvar koji mu se desio. Ovde takođe može lako da ih pronađe bez većih poteškoda (ukoliko ne pronađe iz prve majstora, administratori mu pomažu oko toga).

## 6. LITERATURA


[1] *Client/Server applications*
https://en.wikipedia.org/wiki/Client%E2%80%93server_model
[2] *Backend*
https://learntocodewith.me/posts/backend-development/
[3] *SOAP*
https://www.tutorialspoint.com/soap/what_is_soap.htm
[4] *REST*
https://www.codecademy.com/articles/what-is-rest
[5] *Frontend*
https://en.wikipedia.org/wiki/Front-end_web_development
[6] *HTML*
https://www.w3schools.com/html/
[7] *JavaScript*
https://en.wikipedia.org/wiki/JavaScript
[8] *MySQL*
https://en.wikipedia.org/wiki/MySQL


### Kratka biografija:




**Vladimir Jovičić** rođen je u Somboru 1995. god. Master rad na Fakultetu tehničkih nauka iz oblasti Računarstvo i automatika - Primenjene računarske nauke i informatika - Elektronsko poslovanje odbranio je 2019.god.

kontakt: vlada.jova@yahoo.com