Generator evidencija poslovanja

Ernjak, Dominik

VELEUČILIŠTE U BJELOVARU
PREDDIPLOMSKI STRUČNI STUDIJ RAČUNARSTVO

GENERATOR EVIDENCIJA POSLOVANJA

Završni rad br. 09/RAČ/2020

Dominik Ernjak

Bjelovar, listopad 2020.

obrazac ZR - 001

Veleučilište u Bjelovaru

Trg E. Kvaternika 4, Bjelovar

1. DEFINIRANJE TEME ZAVRŠNOG RADA I POVJERENSTVA

Kandidat: **Ernjak Dominik** Datum: 31.08.2020. Matični broj: 001849

JMBAG: 0069071504

Kolegij: **BAZE PODATAKA**

Naslov rada (tema): **Generator evidencija poslovanja**

Područje: **Tehničke znanosti** Polje: **Računarstvo**

Grana: **Programsko inženjerstvo**

Mentor: **Tomislav Adamović, mag.ing.el.** zvanje: **predavač**

Članovi Povjerenstva za ocjenjivanje i obranu završnog rada:

1. **Krunoslav Husak, dipl.ing.rač., predsjednik**
2. **Tomislav Adamović, mag.ing.el., mentor**
3. **Ivan Sekovanić, mag.ing.inf.et comm.techn., član**

2. ZADATAK ZAVRŠNOG RADA BROJ: 09/RAČ/2020

U radu je potrebno na Oracle bazi podataka izraditi sustav za generiranje aplikacija za vođenje evidencije poslovanja.
Aplikacija podrazumijeva web sučelje preko kojeg će se na bazi podataka moći izgenerirati web stranice za unos/pregled/izmjenu podataka dostupnih tablica. Aplikacija će za komunikaciju koristiti JSON preko kojeg će se omogućiti podesiva poruka o greškama prilikom unosa i izmjene podataka te prilagodljive nazive, redoslijed prikaza i vidljivost stupaca kod pregleda podataka.

Zadatak uručen: 31.08.2020. Mentor: **Tomislav Adamović, mag.ing.el.**

REPUBLIKA HRVATSKA VELEUČILIŠTE U BJELOVARU BJELOVAR

Sadržaj

1. Uvod ... 1

2. JSON ... 2

3. Baza Podataka ... 4

3.1 Programski jezik SQL ... 5

3.1.1 Primjena programskog jezika SQL ... 6

3.2 Oracle ... 7

3.3 SQL Developer ... 7

3.3.1 Primjena SQL developer-a ... 8

4. Web server ... 14

4.1 XAMPP ... 14

4.1.1 PHP ... 15

4.1.2 Primjena PHP-a ... 15

5. Web sučelje ... 17

5.1 HTML ... 17

5.1.1 Pregled web sučelja ... 17

5.2 JavaScript ... 25

5.2.1 Primjena JavaScript-a ... 26

6. ZAKLJUČAK ... 31

7. LITERATURA ... 32

8. OZNAKE I KRATICE ... 34

9. SAŽETAK ... 35

10. ABSTRACT ... 36

1. Uvod

U suvremeno se doba informacija navodi kao najznačajniji resurs te se zbog toga javlja velika potreba za spremanjem te omogućavanjem jednostavnijeg pristupa informacijama bez obzira radi li se o velikim kompanijama ili malim poduzetnicima. Informacije tako postaju svojevrstan resurs te omogućuju jednostavnije vođenje evidencije poslovnog subjekta u svim aspektima poslovnja, od pregleda broja zaposlenih, evidencije prihoda i rashoda, omogućavanja uvida u popis dostupnih proizvoda itd. Prikazivanje evidencije poslovanja korisnicima te pojednostavljivanje njene uporabe za korisnike jest cilj ovoga rada koji omogućuje generiranje evidencije poslovanja bilo kojeg aspekta poslovnog subjekta.

Rad generatora evidencije poslovanja temelji se na *Oracle*-ovoj bazi podataka unutar koje se nalazi cjeloukupna funkcionalnost generatora evidencije poslovanja koja radi operacije poput unosa, izmjene te dohvaćanja podataka, pohranu podataka, te također obrađivanje istih podataka koristeći ugrađene procedure preko kojih se upravlja sustavom. Za upravljanje navedenim sustavom izrađeno je web sučelje namijenjeno njegovom korištenju na način da se kreiraju nove web aplikacije koje služe za pregled evidencije tj. pristup podatcima.

Unutar rada objasnit će se JSON format koji se koristi unutar web sučelja i baze podataka. Nadalje će biti objašnjena sama baza podataka te njezin rad s podatcima podataka u svrhu generiranja evidencije poslovanja. Također će se objasniti funckionalnost web sučelja te njegov način rada i komunikacije istog s bazom podataka preko web servera.

2. JSON

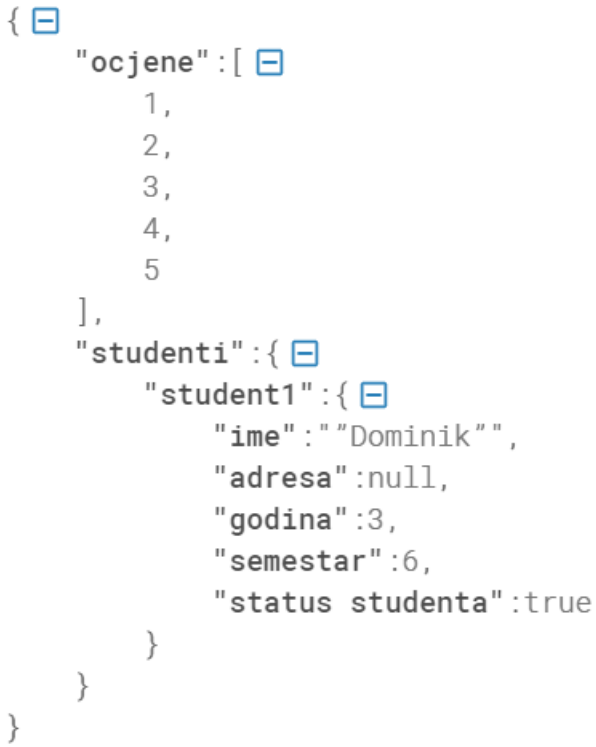
Svi podatci koji se dohvaćaju, šalju i obrađuju u ovome radu temeljeni su na JSON (*engl.* JavaScript Object Notation) formatu koji odlikuje svojom jednostvnošću uporabe i mogućnošću lakog pregleda podataka. JSON je format namijenjen za razmjenu podataka te je lako čitljiv općoj populaciji. Također, vrlo je jednostavan te ne zahtijeva puno računalnih resursa kako bi se podatci generirali i koristili. JSON je baziran na podskupu JavaScript-ovog standarda ECMA-262 trećeg izdanja. Nadalje, JSON je tekstualni format koji je potpuno neovisan o jeziku u kojem se koristi, a koristi pravila poznata programima iz obitelji C-a uključujući *C*, *C++*, *C#*, *Java*, *JavaScript* i sl. Upravo ta svojstva čine JSON idealnim formatom za razmjenu podataka [9].

JSON format sastoji se od dvije strukture, a to su kolekcija parova ključ/vrijednost i lista vrijednosti. Kolekcija parova ključ/vrijednost (nadalje objekt) u raznim jezicima realizira se kao objekt, struktura, riječnik, hash tablica, popis ključeva ili asocijativni niz. Lista vrijednosti (nadalje niz) u većini jezika se ostvaruje kao niz, vektor ili popis. Upravo ove univerzalne strukture podataka koriste i podržavaju gotovo svi moderni programski jezici, a upravo zbog toga što se JSON temelji na strukturama koje koriste većina modernih programskih jezika, ovaj format podataka čini jednostavnim za korištenje [9].

Objekt u JSON formatu zapisuje se unutar vitičastih zagrada gdje otvorena zagrada „{“ označava početak, a zatvorena „}“ kraj. Unutar vitičastinh zagrada zapisuju se podatci u obliku ključa i vrijednosti gdje se ključ zapisuje u tekstualnom obliku tj. zapisuje se unutar navodnika koji mogu biti jednostruki ili dvostruki, a nakon navodnika zapisuje se dvotočka “:”. Nakon dvotočke zapisuje se vrijednost ključa. Nakon zapisane vrijednosti moguće je dodavati naknadne objekte koji se moraju odvojiti zarezom. Primjer objekta u JSON obliku: {“broj”:3, “ocjena”:4}, dakle broj i ocjena označavaju ključeve, a brojevi 3 i 4 označavaju vrijednosti. Nasuprot objektu, niz u JSON obliku zapisuje se unutar uglatih zagrada „[]“ i svaka zapisana vrijednost se odvaja zarezom, osim ako se ta vrijednost ne nalazi na samom kraju niza ili je jedina vrijednost unutar niza. Primjer niza u JSON obliku : [“test”,1,2,3] [7].

Vrijednosti koje se zapisuju u JSON formatu mogu biti u obliku tekstualnog formata koji mora biti naveden u navodnicima, u obliku brojčane vrijednosti, u obliku *boolean*

vrijednosti koja može biti *true* ili *false*, u obliku null vrijednosti, novog objekta te novog niza [9]. Ove strukture unutar JSON formata mogu biti i ugniježdene tako da je moguće zapisati JSON format koji je prikazan na slici 2.1.



Slika 2.1. Prikaz izgleda ugnježdenost JSON formata

Ugnježdenost podataka u JSON-u omogućava jednostavnije i efikasnije korištenje velikih količina podataka prilikom njihove razmjene te se upravo zbog toga JSON format koristi unutar rada.

3. Baza Podataka

Kako bi se detaljnije analizirao način rada samog generatora evidencije poslovanja, najprije je važno definirati temeljne pojmove koji se njega tiču, a jedan od njih je i baza podataka. Baza podataka jest svojevrsna organizirana kolekcija informacija spremljena na računalnom sustavu. Samim time što su informacije pohranjene na računalnom sustavu omogućeno je lako pristupanje i upravljanje informacijama. Baza podataka tipično sadrži iznimno velike količine podataka koji su najčešće u tekstualnom obliku, a upravljanje baze odvija se putem DBMS-a (*engl*. Database Management System). DBMS tako omogućava interakciju s krajnjim korisnikom i aplikacijom, ali i sa samom bazom za dohvaćanje i analiziranje podataka [4]. Također, postoji više tipova DBMS-ova ovisno o baznom modelu koji podržavaju. Relacijski model baza postao je dominantan u 80-im godinama prošlog stoljeća kada su se podatci unutar baze spremali u tablice koje su se sastoje od stupaca i redaka te su indeksirane radi lakšeg pronalaženja podataka [4]. Postojeći DBMS-ovi koji omogućuju upravljanje bazom i njenim podatcima mogu se podijeliti u 4 skupine:

- Definicija podataka– kreiranje, modificiranje i brisanje definicija koji organiziraju podatke
- Ažuriranje – umetanje, modificiranje i brisanje postojećih podataka
- Dohvaćanje
- Administracija – registriranje i nadgledanje korisnika, provođenje sigurnosnih mjera za zaštitu podataka, nadgledanje performansi baze, čuvanje integriteta podataka i dohvaćanje podataka koji su oštećeni nekim iznenadnim događajem

Baza podataka kao i DBMS radi u skladu s određenim baznim modelom. Bazni model je tip modela koji određuje logičku strukturu baze i time određuje koji tipovi podataka se mogu pohraniti, organizirati i manipulirati. Neki od tipova baznih modela su:

- Hijerarhijski model – podatci su organizirani nalik na strukturu drveta te su spremljeni kao zapisi koji su povezani međusobno preko veza. Zapisi su kolekcije polja gdje svako polje sadrži samo jednu vrijednost. Zapisi mogu imati samo jednog *roditelja* (*engl*. parent) i više *djece* (*engl.* child)

- Mrežni model – u odnosu na hijerarhijski model, mrežni model je fleksibilan jer svaki zapis može imati više roditelja i više djece i time formira strukturu nalik na graf

- Relacijski model – pristup upravljanja podataka koristeći strukturu i jezik sličan predikatnoj logici prvog reda gdje se podatci prikazuju kao konačan skup elemenata grupiran s vezama [4].

Za korištenje baze koristi se programski jezik SQL(Structured Query Language) koji može varirati u sintaksi, ali pretežito je sličan za sve baze. Neke od najpopularnijih baza podataka su: *Oracle, MySQL, Microsoft SQL Server, PostgreSQL, MongoDB, IBM DB2, Elasticsearch, Redis, SQLite, Cassandra* i *MariaDB* [4].

3.1 Programski jezik SQL

Kao što je već navdeno, SQL ili strukturni upitni jezik koristi se za dizajniranje i upravljanje podatcima koji se nalaze unutar baze podataka temeljene na relacijskom modelu. Prednost SQL-a nad starijim jezicima je ta što upravo one omogućuje pristup većoj količini podataka s jednom naredbom te također eliminira potrebu za specifiranjem uputa o tome kako pronaći zapis bez obzira postoji li indeks. SQL je koncipiran na temelju relacijske algebre te se sastoji od mnogobrojnih naredbi koje se mogu klasificirati kao podjezici za izradu, traženje, ažuriranje i brisanje podataka unutar baze podataka [6].

Programski jezik SQL je prvobitno razvijen u *IBM*-u i temeljen je na radu relacijskog modela Edgar F. Codd-a iz 70-ih godina dvadesetog stoljeća. Prva verzija jezika zvala se SEQUEL (*engl.* Structured English Query Language), a dizajnirana je tako da manipulira i dohvaća podatke u *IBM*-ovoj originalnoj kvazi-relacijskog bazi pod nazivom „System R“ koja je razvijena tijekom 1970-ih, a akronim SEQUEL kasnije je promjenjen u SQL radi postojeće korporacije pod nazivom SEQUEL u Ujedinjenom Kraljevstvu. Kasnih 1970-ih godina tvrtka pod nazivom *Relational Software, Inc*., danas poznata kao *Oracle Corporation*, uvidjela je potencijal koncepta programskog jezika SQL. Tada je tadašnji *Relational Software, Inc* razvio vlastitu verziju jezika na relacijskom modelu s ciljem prodaje Američkim vladinim agencijama. No, 1979. godine predstavljena je prva komercijalna implementacija programskog jezika SQL pod nazivom *Oracle V2* za VAX računala [18].

3.1.1 *Primjena programskog jezika SQL*

Kada je riječ o programskom jeziku SQL, važno je istaknuti kako se on može podijeliti na nekoliko jezičnih elemenata: klauzula, izraz, iskaz, upit, izvještaj. Klauzula u sastavu programskog jezika SQL može ali i ne mora biti sastavni dio izvještaja i upita kojom primjenjujemo filtere da bih smo dobili željene rezultate. Izrazi su kombinacije jedne ili više vrijednosti, operatora ili SQL funkcija koje se primjenjuju na vrijednost i time se proizvode ili skalarne vrijednosti ili tablice u kojima se nalaze podatci u obliku redaka i stupaca. Iskazi određuju uvjet koji može biti evaluiran u sustavu trojne logike čiji su rezultati točan (*engl.* true), lažan (*engl.* false) i nepoznat ili u obliku Booleove algebre čija struktura sažima osnovne operacije I (*engl.* AND), ILI (*engl.* OR) i NE (*engl.* NOT) za ograničavanje rezultata izvještaja i upita koji dohvaćaju podatke osnovane na određenom kriteriju radi promjene toka programa. Izvještaj je kod koji može biti izvršen samostalno za generiranje rezultate ili pak može biti kombinacija upita koji su u mogućnosti dohvatiti, izmjeniti ili obrisati podatke unutar tablice/a zasnovanih na određenom kriteriju, uvjetu, izrazu ili iskazu. Izvještaji se kategoriziraju u četri različita tipa: jezik za obradu podataka (*engl.* Data Manipulation Language), jezik definiranja podataka (*engl.* Data Definition Language), jezik kontrole podataka (*engl.* Data Control Language) i jezik kontrole transakcija (*engl.* Transaction Control Language). Unutar jezika za obradu podataka koriste se četri naredbe kojima se manipuliraju podatci poput „SELECT“ naredbe kojoj je primarna svrha izvlačenje tj. pregled podataka iz tablica, „INSERT“ naredbe koja ubacuje podatke u tablicu, „UPDATE“ naredbe koja izmjenjuje postojeće podatke i „DELETE“ naredbe kojoj je svrha brisanje podataka iz tablice. Jezik za definiranje podataka za razliku od jezika za obradu podataka koristi se za upravljanje tablicama poput naredbi „CREATE“, „ALTER“, „DROP“ gdje se „CREATE“ koristi za kreiranje novih tablica, „ALTER“ za izmjenu definicije tablice ili dodavanje/brisanje stupca unutar tablice i DROP koji odbacuje cijeli tablicu tj. obriše postojeću tablicu. Jezik kontrole podataka definira kotrolu podataka u bazi s opcijama „GRANT“ gdje se određenom korisniku baze dodjeljuju ovlasti nad tablicama unutar baze koje on može koristiti.Nasuprot „GRANT“-a je „REVOKE“ čija je svrha ukinuti date ovlasti određenom korisniku. U jeziku kontrole podataka koriste se naredbe kojima se upravlja s transakcijama unutar baze za izmjene napravljane od strane jezika za obradu podataka poput naredbe „COMMIT“ koja se koristi za trajno spremanje svih izmjena unutar baze

podataka isto tako postoji opcija „ROLLBACK“ koja vraća stanje baze na prethodno spremljeno stanje [17].

3.2 Oracle

Generator evidencije podataka temelji se na Oracle bazi podataka. *Oracle* je američka korporacija sa sjedištem u Kaliforniji koja se bavi prodajom softvera i tehnologija vezanih uz baze podataka, *cloud* sustava, softvera i proizvoda koji zadovoljavaju potrebe većih organizacija i korporacija. U 2019. Godini *Oracle* je proglašen drugom najvećom softverskom organizacijom po prihodu [15].

Danas poznata kao *Oracle Corporation* osnovana je 1977. godine od strane Larry Ellison-a, Bob Miner-a i Ed Oates-a pod nazivom *Software Development Laboratories* (SDL). Ellison je preuzeo inspiraciju 1970. godine iz članka Edgar F. Codd-a pod nazivom „A Relation Model of Data for Large Shared Data Banks“ koji opisuje relacijski model upravljanja bazama podataka u kojem korisnici imaju ovlasti i mogućnosti upravljati, definirati i održavati bazu. Tadašnji *IBM* razvio je sustav pod nazivom „System R“ gdje je prvi puta korištem relacijski model upravljanja bazom podataka i u kojem je došlo do prve implementacije jezika SQL koji se i danas koristi kao standardni jezik. Nadalje, Ellison je htio napraviti proizvod koji je kompatibilan sa „System R“, ali neuspješno budući da je *IBM* zadržao dio sustava u tajnosti. 1979. godine SDL mijenja ime u *Relational Software, Inc* (RSI), zatim ponovno 1983. u *Oracle System Corporation* kako bi se bliže uskladili sa svojim vodećim proizvodom *Oracle Database*. Naziv *Oracle* dolazi od kodnog imena operacije CIA-inog projekta na kojem je Ellison radio [15].

3.3 SQL Developer

Oracle SQL Developer je besplatno integrirano razvojno okruženje koje pojednostavljaju razvoj i upravljanje *Oracle* bazama u tradicionalnom i *cloud* rješenjima. SQL developer nudi cjeloviti razvoj PL/SQL aplikacija, radni list za pokretanje skripti i upita, DBA konzolu za upravljanje bazom, sučelje izvješća, cjelovito rješenje za modeliranje podataka i platformu za migraciju i premještanje podataka unutar baze (Oracle SQL Developer, 2020).

3.3.1 *Primjena SQL developer-a*

Unutar ovoga rada, logika sustava generatora evidencije poslovanja temelji se na bazi podataka koja je pisana u integriranom razvojnom okruženju SQL developer-a koji je podjeljen u trinaest tablica i tri paketa, a svaka od trinaest tablica ima svoju vlastitu funkcionalnost i način na koji doprinosi radu generatora evidencija poslovanja.

3.3.1.1 Tablice

Tablice unutar baze podataka temeljne su jedinice pohrane podataka gdje se podatci spremaju u stupce i redove. Prema tome, za uspješan rad generatora evidencije poslovanja kreirano je nekoliko tablica koje služe upravo toj svrsi, a neke od njih su : „korisnici“, „page_data“, „page_type“, „user_app“, „user_app_pages“, „user_tables“. Tablica „korisnici“ služi za evidenciju registriranih korisnika koji imaju pristup generatoru evidencije poslovanja. Također služi za verifikaciju korisnika prilikom prijave u web sučelje preko podataka spremljenih unutar tablice poput email adrese i zaporke.

Unutar tablice „page_data“ nalaze se podatci o pojedinačno kreiranoj stranici unutar aplikacije na web sučelju. Te podatke generira web sučelje na temelju odabira konfiguracije korisnika ovisno o tome na koji način korisnik želi da se podatci prikazuju te izboru podataka za pregled ili izmjenu.

Na Slici 3.1. prikazano je web sučelje na kojem korisnik ima opcije: odabir željene tablice iz koje će se podatci prikazivati, promijena naziva stupca prilikom prikazivanja tablice, odabir opcije za prikaz stupca, potreban unos te proizvoljnu poruku greške prilikom unosa pogrešnog podataka. Nadalje, svi navedeni podatci spremaju se u tablicu „page_data“ pritiskom na gumb „SAVE“ u JSON formatu. Prikaz spremljenih podataka vidljiv je na Slici 3.2.

Slika 3.1. Prikaz odabira tablice

```
{
   "table":"testnaTablica",
   "columns":{
      "ID":{
         "show":true,
         "required":false,
         "columnType":"NUMBER",
         "nullable":"N",
         "columnLength":22,
         "errMsg":"Molimo unesite ID",
         "alias":"ID"
      },
      "STUPAC1":{
         "show":true,
         "required":false,
         "columnType":"VARCHAR",
         "nullable":"N",
         "columnLength":60,
         "errMsg":"Molimo unesite STUPAC1",
         "alias":"STUPAC1"
      },
      "STUPAC2":{
         "show":true,
         "required":false,
         "columnType":"NUMBER",
         "nullable":"N",
         "columnLength":22,
         "errMsg":"Molimo unesite STUPAC2",
         "alias":"STUPAC2"
      }
   },
   "pagination":true,
   "perPage":"25"
}
```


Slika 3.2. Prikaz spremljenih podataka

Tablica „page_type“ sadrži dva podatka koja služe za kreiranje novih stranica, a to su ID i naziv stranice koji su zapravo opcije za kreiranje novih tipova stranica. Trenutno tablica „page_type“ pohranjuje 4 vrste stranica: *Login*, *MainPage*, *Izvještaj* i *Forma*. Kod kreiranja nove aplikacije automatski se dodaju stranice *Login* i *Mainpage* koje svaka aplikacija mora sadržavati te na izbor kreiranja nove stranice korisnik ima ili *Izvještaj* ili *Formu,* a izbor i krerianje novih stranica detaljnije je opisan u poglavlju *HTML*.

Tablica „user_app“ sadrži podatke o kreiranim aplikacijama od strane korisnika te njihovo aktivno stanje. Navedena tablica sastoji se od stupaca „user_id“, „app_name“ i „active“ gdje „user_id“ predstavlja ID korisnika koji je kreirao aplikaciju pod nazivom u stupcu „app_name“. Stupac „active“ označava je li aplikacija obrisana postavljanjem vrijednosti u 0, a ako aplikacija nije obrisana, ova vrijednost sadrži broj 1.

Tablica „user_app_pages“ posjeduje podatke o kreiranim stranicama unutar svake aplikacije te se sastoji od stupaca „app_id“, „page_type“, „page_name“, „active“, „used“.

Stupac „app_id“ označava ID aplikacije na kojoj je kreirana stranica te je povezan stranim ključem na stupac ID iz tablice „user_app“ čime je osigurana povezanost između aplikacije i stranica. Stupac „page_type“ sadrži informaciju o tipu stranica koja je prethodno definirana u tablici „page_type“. „page_name“ označava naziv stranice, a stupac „active“ sadrži informaciju o tome je li stranica aktivna. Stupac „used“ ukazuje na to je li navedena stranica prethodno konfigurirana tj. sadrži li podatke za prikaz koje je korisnik prethodno podesio. Ako podatci postoje, oni se spremaju u tablicu „page_data“.
Prilikom kreiranja tablice od strane korisnika također se kreiraju sekvenca i okidač (*engl.* trigger) za navedenu tablicu gdje sekvenca označava brojčanu vrijednost koja se unosi u stupac ID navedene tablice, a za unos sekvence zadužen je okidač koji prije samog unosa podataka u tablicu unese vrijednost sekvence. Podatci o nazivu tablice, sekvence i okidača nalaze se u tablici „user_tables“.

3.3.1.2 Paketi

Paketi unutar *SQL developer*-a imaju zadaću spremanja procedura i funkcija napisanih od strane programera te tako spremljene procedure i funkcije imaju mogućnost pozivanja od strane web sučelja te na taj način izvršavaju svoju zadaću [14]. Nadalje, paketi su podijeljeni tako da svaki paket ima svoju funkcionalnost vezanu za određeni dio generatora evidencije poslovanja koji mora izvršiti. Paketi koji su definirani na bazi su „Generator“, „Appdata“ i „Userapp“.

Paket „Generator“ unutar ovoga rada ima najvažniju svrhu, a to je pozivanje drugih procedura i funkcija koje se nalaze na bazi. Paket „Generator“ je centralna točka baze kroz koju ulaze i izlaze svi podatci. Unutar samog paketa nalazi se jedna javna procedura koja je zadužena za rad cjelokupnog paketa prilikom koje se nakon svakog poziva na bazu poziva upravo ta procedura pod nazivom p_handle_call. Procedura p_handle_call prima parametar u JSON formatu te također vraća podatke u JSON formatu, a ulazni parametar koji se prosljeđuje proceduri izgleda ovako:
{"package":"appdata","procedure":"p_login","input":["admin@vub.hr","admin"]}.

Dakle, *package* označava paket koji se poziva, a to je u ovom slučaju „Appdata“. Potom dolazi ključ *procedure* označavajući naziv procedure koji se mora pozvati, a to je p_login. Nakon naziva paketa i procedure dolaze podatci potrebni za poziv navedene procedure koji se nalaze pod ključem *input* u obliku niza koji u ovome slučaju sadrži

podatke o email adresi i zaporci potrebnoj za prijavu u web sučelje generatora evidencije poslovanja. Naime, unutar navedenog paketa nalazi se i jedna privatna ključna funkcija čiji je zadatak generirati kod za poziv potrebne procedure ili funkcije pod nazivom f_generate_call. Zadatak ove funkcije vrlo je važan jer se na njoj temelji cjelokupna komunikacija unutar baze podataka. Funkcija f_generate_call prima tri parametra, a to su naziv procedure, naziv paketa i input podatci za poziv procedure. Na temelju tih podataka vraća se generirani kod za pozivanje prosljeđene procedure na način da se provjerava postoji li proslijeđena procedura unutar baze podataka i ako postoji, dodaje se kod prema standardnom programskom jeziku SQL za poziv procedure. Na Slici 4. prikazan je izgled generiranog koda prilikom prijave na web sučelje generatora evidencije poslovanja. Naime, unutar navedenih paketa svaka procedura koja se može pozvati ima izlazni parametar u obliku JSON formata te se radi toga na Slici 3.3. nalazi parametar „:o_json“ koji je vezani (*engl.* BIND) parametar koji prilikom izvršenja navedenog koda vraća podatke u varijablu koja je povezana s navedenim parametrom.

```
BEGIN
APPDATA.p_login('admin@vub.hr','admin', :o_json);
END;
```

Slika 3.3. Izgled generiranog koda prema zadanim parametrima

Na Slici 3.4. nalazi se dio koda koji opisuje prethodne korake gdje varijabla „l_sql“ prima generirani kod od strane funkcije f_generate_call te se dalje izvršava navedeni kod naredbom „EXECUTE IMMEDIATE l_sql USING OUT l_json_data“ . Unutar naredbe nalazi se varijabla l_json_data u koju se spremaju podatci dobiveni izvršavanjem koda sa Slike 4. gdje je ta ista varijabla vezana za prethodno navedenu varijablu „:o_json“.
Na temelju ovog opisa izvršavaju se sve procedure koje je moguće pozvati s baze podataka koje služe za dobivanje informacija i podataka potrebnih za rad generatora evidencije poslovanja.

```
l_sql := f_generate_call(l_procedure,l_package,l_input);

EXECUTE IMMEDIATE l_sql USING OUT l_json_data;
```

Slika 3.4 Izvršavanje poziva procedura

Paket „Appdata“ namjenjen je korištenju i upravljanju generatorom evidencije poslovanja na način da omogućava krajnjem korisniku kreiranje novih aplikacija, dodavanje novih stranica u kreiranu aplikaciju, proizvoljan unos, pregled i izmjenu podataka, način i tip na koji će se stranica prikazivati, kreiranje novih tablica te izmjenjivanje postojećih, dodavanje listi vrijednosti itd. Sve navedene mogućnosti generatora evidencije poslovanja omogućuje upravo paket „Appdata“ u korelaciji s posebno kreiranim tablicama koje su navedene u prethodnom potpoglavlju. Paket se sastoji od dvadeset tri javne procedure koje je moguće pozvati s web sučelja, a neke od njih su p_create_new_app, p_get_all_apps, p_get_full_table_info, p_save_page, p_get_column_names i p_delete_page. Navedene procedure funkcioniraju na način da se preko njih unose, izmjenjuju i dohvaćaju podatci u/iz predviđenih tablica. Nadalje, na primjeru procedure p_create_new_app koja je prikazana na Slici 3.5. možemo vidjeti da se cijeli rad procedure temelji na unosu naziva aplikacije te ID-a korisnika koji kreira aplikaciju u tablicu „user_app“. Prema tim unesenim podatcima druga procedura pod nazivom p_get_all_app, koja je zadužena za dohvaćanje naziva aplikacija određenog korisnika, dohvaća podatke iz tablice „user_app“ te ih vraća web sučelju koji tada na temelju primljenih podataka prikazuje aplikacije određenog korisnika. Također, procedura p_get_column_names, koja je zadužena za dohvaćanje informacija o tablici, poput naziva stupca, tipa podataka koji se zapisuje unutar stupca, maksimalne moguće dužine tog podatka i podatka o tome može li stupac sadržavati null vrijednost tj. može li ne sadržavati ikakvu vrijednost. Ovako opisana procedura radi na primjeru SELECT-a koji se može vidjeti na Slici 3.6. Nadalje, procedura pod nazivom p_delete_page karakterizirana je svojom jednostavnošću izmjene podataka u tablici te je zadužena za brisanje stranica na web sučelju. Dakle, procedura radi tako da primi podatke o nazivu stranice koja se želi obrisati te tada unutar tablice „user_app_pages“ izmjenjuje stupac pod nazivom „active“ iz 1 u 0 i time stranica biva obrisana budući da se sve stranice dohvaćaju prema uvjetu „active“ koji predstavlja 1 kao aktivnu stranicu.

```
procedure p_create_new_app(i_name in    varchar2,
                           i_hash in    varchar2,
                           o_json   out JSON_OBJECT_T
)AS
l_id number;
l_msg varchar2(60) := '';
l_code number := 0;
l_json JSON_OBJECT_T;
l_app_id number;
BEGIN
  l_json := JSON_OBJECT_T();
  l_id := f_get_id(i_hash);
  l_app_id := seq_user_app.nextval;

  INSERT INTO user_app (id,user_id,app_name) values (l_app_id,l_id,i_name);

  commit;

  p_create_new_app_pages(l_app_id);

  l_json.put('o_code',l_code);
  l_json.put('o_message',l_msg);
  o_json := l_json;

  EXCEPTION
      WHEN OTHERS THEN
          o_json := f_build_error(sqlcode, sqlerrm, 'p_create_new_app');
END p_create_new_app;
```

Slika 3.5. Prikaz procedure p_create_new_app

```
SELECT
  table_name,
  column_name,
  data_type,
  data_length,
  nullable
FROM
  USER_TAB_COLUMNS
WHERE
  table_name = UPPER(i_table)
ORDER BY
  COLUMN_ID ASC;
```

Slika 3.6. SELECT za dohvaćanje detalja o tablici

4. Web server

Korištenje informacijsko komunikacijske tehnologije danas je gotovo nemoguće bez korištenja pogodnih web servera. Web server može se odnositi na softverski ili hardverski dio ili na kombinaciju oba [12]. Sa softverske strane web server se sastoji od više dijelova koji kontroliraju način na koji korisnik dolazi do podataka. U ovom je radu korišten HTTP (*engl.* HyperText Transfer Protocol) server. HTTP server je softver koji interpretira URL (engl. Uniform Resource Locator) i HTTP zahtjeve te se na temelju njih vraća traženi sadržaj [5].

Kada je riječ pak o hardveskom dijelu web servera, on također predstavlja važnu sastavnicu generatora evidencije poslovanja. Hardverski dio web servera je računalo koje pohranjuje softver servera i komponente web stranice poput HTML dokumenata, slika, CSS-a i JavaScript podataka. Web server se povezuje na mrežu i podržava razmjenu podataka sa ostalim uređajima spojenim na nju. U nastavku će biti objašnjena komunikacija koja se odvija preko web servera koristeći platformu XAMPP koja je zadužena za uspostavljanje web servera [2].

4.1 XAMPP

XAMPP je besplatna platforma otvorenog koda (*engl.* Open-source) koja se sastoji od *Apache* HTTP servera, *MariaDB* baze podataka i programskih jezika PHP i PERL, a namijenjena je jednostavnijem pokretanju web servera bilo da je riječ o lokalnom serveru ili web serveru. Instalacija XAMPP-a potpuno je jednostavna te je za nju potrebno samo preuzimanje instalacijske datoteke s interneta i pokretanje iste kako bi XAMPP instalirao. XAMPP u ovome radu je vrlo bitan budući da s njime dolazi *Apache* HTTP poslužitelj preko kojega, zajedno s programiskim jezikom PHP, web stranica komunicira s bazom podataka i tako omogućava komunikaciju između baze podataka i web sučelja [13].

Prvobitno, za potrebe ovoga rada bila je odabrana platforma WAMP (*Windows*, *Apache*, *MariaDB*, PHP) koja ima istu primjenu kao i XAMPP. Jedina razlika leži u tome što je WAMP namijenjen isključivo za windows sustav. Do promjene platformi došlo je u trenutku kada je jedan od dva servisa WAMP-a prestao raditi iz nepoznatih razloga. Opcija popravka onemogućenih servisa nije omogućila oporavak navedenog servisa te prilikom istraživanja navedenog problema nije pronađeno zadovoljavajuće rješenje pa je zatim

provedeno ponovno instaliranje platforme WAMP. Nadalje, nakon ponovne instalacije navedeni je servis bio funkcionalan, no samo na kratko. Kako bi se izbjegao taj problem za kojega trenutno nema odgovarajućeg rješenja, odlučeno je prebaciti se na platformu XAMPP.

4.1.1 *PHP*

PHP ili Hypertext Preprocessor je skriptni jezik otvorenog koda koji je široko korišten jezik čija je opća namjena pogodna za web razvoj i ima mogućnost umetanja unutar HTML-a. PHP je danas jedna od nakorištenijih i najnaprednih *server-side* skriptnih tehnologija u uporabi [6]. Sintaksa programskog jezika PHP slična je mnogim drugim jezicima, a čak ima i istoznačne funkcije kao i neki drugi jezici poput C ili Perl-a. PHP je široko korišten radi svoje podrške za manipuliranje velikim spektrom baza podataka poput *MySQL, PostgreSQL, dBase, MariaDB, Oracle* itd. Također PHP sadrži bogatu zbirku funkcija za manipuliranje različitim tipovima sadržaja od grafike (jpg, flash, png) pa sve do .NET modula i XML-a [10].

4.1.2 *Primjena PHP-a*

Za potrebe ovoga rada bilo je potrebno konfigurirati PHP i dodati datoteku potrebnu za komuniciranjem PHP i Oracle baze podataka. PHP u ovome radu služi kao posrednik koji podatke koje prima s web sučelja proslijeđuje bazi podataka koja ih nakon obrade putem PHP-a vraća natrag webu na daljnje korištenje. Na Slici 4.1. prikazano je kako se „POST“ i „GET“ zahtjevi koji se dobiju s weba pretvaraju u JSON oblik te se nadalje u „IF“ uvjetu provjerava naziv akcije koji se želi izvršiti.

```
switch ($_SERVER['REQUEST_METHOD']) {
    case 'POST':
        $injson = json_encode($_POST);
        break;
    case 'GET':
        $injson = json_encode($_GET);
        break;
    default:
        echo $request_err;
        return;
}
```

Slika 4.1. Pretvaranje dolaznih podataka u JSON format

Nadalje, unutar PHP nalazi se funkcija pod nazivom „handleCall“ koja je zadužena za kreiranje veze s bazom preko funkcije „getDBCon“ (Slika 4.2.) i komuniciranje sa istom na način da se podatci koji su poslani ponovno pretvore u JSON oblik tako da se proslijedi naziv paketa koji se poziva. Zatim se navedu podatci koje će baza obrađivati te se poziva procedura na bazi pod nazivom „p_handle_call“ koja dalje izvršava potrebnu proceduru te vraća podatke u varijablu o_json.

```
31  function handleCall($injson){
32      $data = json_decode($injson);
33      $conn = getDBCon();
34      $proc = $data->procedure;
35      $i_json = '{"package":"' . $data->package . '","input":[' . $data->parameters . ']}';
36      $sql = "BEGIN generator.p_handle_call(:i_proc,:i_json,:o_json); END;";
37      $stmt = oci_parse($conn, $sql);
38
39      oci_bind_by_name($stmt, ':i_proc', $proc);
40      oci_bind_by_name($stmt,':i_json', $i_json);
41      oci_bind_by_name($stmt,':o_json', $o_json, 30000);
42
43      oci_execute($stmt);
44      echo $o_json;
45
46  }
```

Slika 4.2. Kreiranje veze s bazom i slanje podataka za daljnju obradu

5. Web sučelje

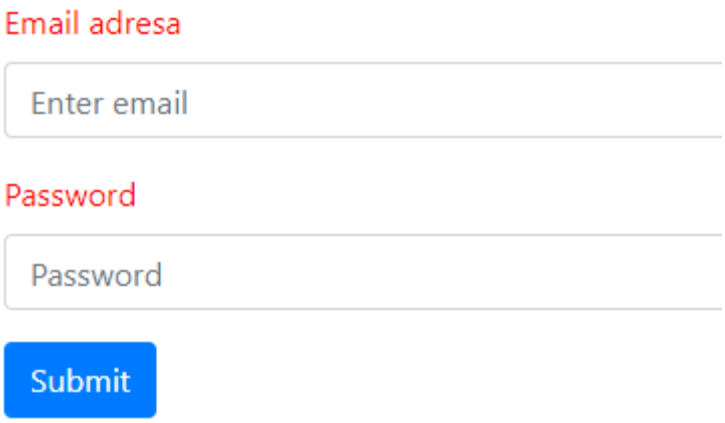
Za potrebe ovoga rada izrađeno je web sučelje kako bi se detaljnije prikazao sustav za generiranje aplikacija za vođenje evidencije poslovanja na kojoj je moguće generirati web stranicu za unos/pregled/izmjenu podataka dostupnih tablica i izvještaja raznih oblika.
U izradi web sučelja korišten je HTML za prikaz podataka, JavaScript za funkcionalnost i ponašanje stranice i CSS za uređivanje izgleda web stranice [1].

5.1 HTML

HTML (*engl.* HyperText Markup Language) je standardni jezik za kreiranje web stranica koji definira značenje i strukturu sadržaja. HTML se sastoji od niza elemenata koji se zatvaraju ili umeću u različite dijelove sadržaja kako bi se prikazali ili djelovali na određeni način [16].

5.1.1 *Pregled web sučelja*

Za potrebe rada web sučelja generatora evindecije poslovanja kreirane su tri HTML datoteke pod nazivom indexVuxom.html, appLogin.html i userApp.html gdje indexVuxom služi kao stranica za prijavu i registraciju korisnika (Slika 5.1.).



Slika 5.1. Prikaz prijave u web sučelje generatora evidencije poslovanja

Nadalje, nakon uspješne prijave pojavljuje se glavni izbornik koji je prikazan na Slici 5.2. koji sadrži opcije za kreiranje nove aplikacije, pregled te uređivanje prethodno kreiranih aplikacije te opcija za uvoz aplikacije.

Slika 5.2. Prikaz glavnog izbornika web sučelja

HTML datoteke koje su zadužene za prikaz web sučelja generatora evidencije poslovanja imaju jednostvanu, ali značajnu ulogu koja služi samo za prikaz. Na slici 5.3. prikazan je cijeli kod koji se nalazi u samom tijelu dokumenta indexVuxon. Tijelo datoteke je podjeljeno na dva div-a pri čemu u prvom div-u s ID-jem „container“ koji predstavlja „spremnik“ omogućava ubacivanje podataka za prikaz, a drugi div sa oznakom modal služi kao skočni prozor koji ima razne primjene.

```
<body >
    <div class="container" id="container">
    </div>
    <div class="modal fade" id="modal" role="dialog">
        <div class="modal-dialog" id="modalSize">
          <div class="modal-content">
            <div class="modal-header" id ="modalHeader">
            </div>
            <div id ="modalOptions">
            </div>
            <div id ="modalBodyOpt">
            </div>
            <div class="modal-body" id="modalBody">
            </div>
            <div class="modal-footer" id="modalFooter">
            </div>
          </div>
        </div>
      </div>
</body>

</html>
```

Slika 5.3. Izgled HTML tijela web sučelja

Također, važno je napomenuti da se u navedenoj datoteci koriste dodatne biblioteke poput AJAX-a čiji zadatak je [3], komunikacija s prethodno navedenim web serverom *Apache*, *jQuery*-a koji služi kao biblioteka JavaScripta koja olakšava manipulaciju i prijelaze HTML datoteka (Slika 5.4.). Unutar navedenih biblioteka nalazi se JavaScript dokument pod nazivom „handlePages“ koji je zadužen za kreiranje i generiranje potrebnih podataka za prikaz web sučelja i ubacivanje istih u kontejner koji će detaljnije biti opisan u sljedećem poglavlju.

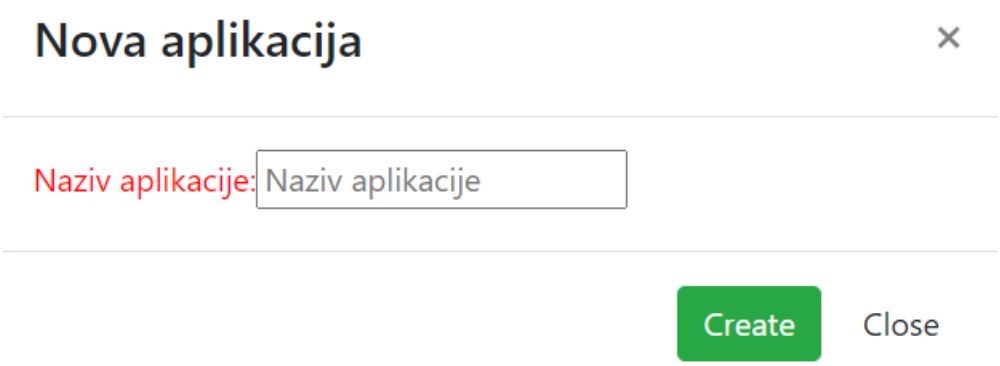
```
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">
<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.3.1/css/all.css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.9.3/Chart.js"></script>
<script src="https://cdn.jsdelivr.net/npm/sweetalert2@9"></script>
<script src="https://code.jquery.com/ui/1.12.1/jquery-ui.js"></script>

<script src = "../javascript/handlePages.js"></script>
```

Slika 5.4. Prikaz uporabnih biblioteka

Na glavnom izborniku web sučelja generatora evidencije poslovanja nalaze se tri opcije koje omogućuju kreiranje nove aplikacije, pregled postojećih aplikacija i uvoz nove aplikacije. Naime, klikom na gumb „Nova aplikacija“ otvara se skočni izbornik koji omogućuje stvaranje nove aplikacije koji je prikazan na Slici 5.5.



Slika 5.5. Prikaz skočnog izbornika prilikom kreiranja nove aplikacije

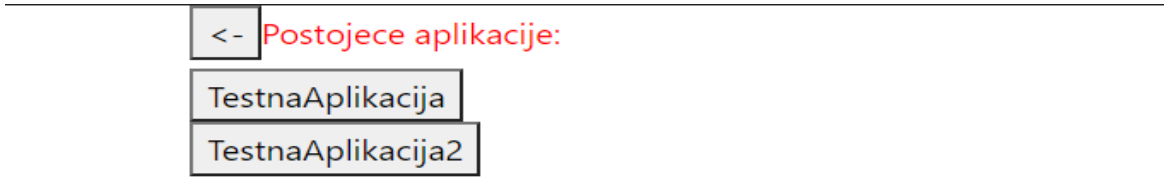
Nadalje, pritiskom na gumb za dodavanje nove aplikacije pokreće se funkcija pod nazivom newAppModal koja je prikazana na Slici 5.6. Funkcijom newAppModal može se vidjeti način na koji se upravlja skočnim prozorom prilikom čega funkcija pod nazivom editModal postavlja naziv skočnog prozora, funkcija alterModalBody određuje što će biti prikazano u tijelu skočnog izbornika. U ovome je slučaju to forma za unos naziva nove aplikacije. Nadalje, funkcija editModalFooter zadužena je za prikazivanje mogućih akcija unutar skočnog prozora poput gumba „CREATE“ koji kreira novu aplikaciju. Prilikom kreiranja nove aplikacije automatski se kreiraju dvije stranice pod nazivom *Login* i *MainPage* koje su zadužene za prijavu korisnika u aplikaciju prilikom pokretanja te za prikaz kreiranih stranica unutar aplikacije.

```
function newAppModal(){
    var modalName = "Nova aplikacija";
    var btnCreate = '<button type="button" class = "btn btn-success" id="btnCreateApp">Create</button>';
    var modalBody = '<label for="inputNaziv">Naziv aplikacije: </label>';
    modalBody += '<input  id="inputNaziv"  placeholder="Naziv aplikacije"><br>';

    editModal(modalName);
    alterModalBody(modalBody);
    editModalFooter(btnCreate);
}
```

Slika 5.6. Prikaz upravljanja skočnim izbornikom

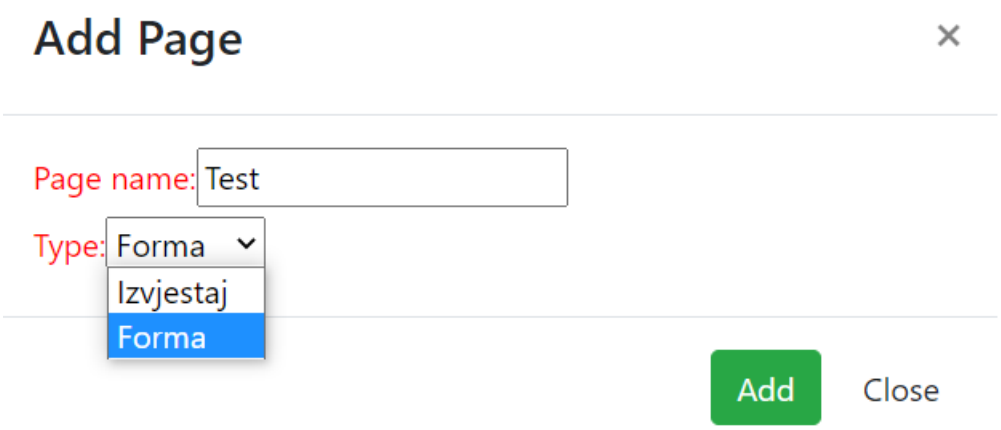
Nadalje, pritiskom na gumb „POSTOJEĆE APLIKACIJE“ otvara se izbornik prikazan na Slici 5.7. u kojemu se nalaze prethodno kreirane aplikacije od strane korisnika koje se dohvaćaju s baze podataka. Pritiskom na jednu od tih kreiranih aplikacija otvara se novi izbornik koji je prikazan Slikom 5.8. te se u njemu nalaze opcije za uređivanje sadržaja i izgleda aplikacije. U ovome izborniku korisnik ima opcije za kreiranje i uređivanje tablica iz kojih će se podatci prikazivati, mogućnost kreiranja listi vrijednost koje omogućavaju jednostavniji i pregledniji unos podataka te opcije za brisanje aplikacije, pokretanje iste i kreiranje novih stranica koje će biti prikazana unutar aplikacije.



Slika 5.7. Prikaz izbornika postojećih aplikacija

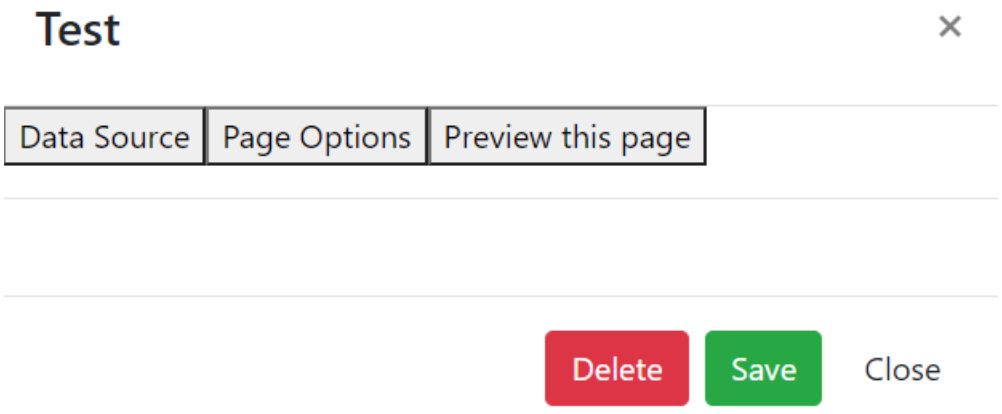
Slika 5.8. Prikaz izbornika za uređivanje aplikacije

Pritiskom na gumb “ADD PAGE” otvara se skočni izbornik prikazan koji omogućava dodavanje nove stranice unutar aplikacije unosom naziva nove stranice te odabirom tipa stranice iz padajućeg izbornika koji može biti *Izvještaj* ili *Forma*. Naime, odabirom izvještaja korisnik ima mogućnost kreiranja prikaza podataka u obliku tablice ili grafa raznih oblika, a odabirom forme korisnik također ima mogućnost pregleda, izmjene te unosa novih podataka koji se prikazuju u obliku tablice (Slika 5.9.).



Slika 5.9. Prikaz kreiranja nove stranice

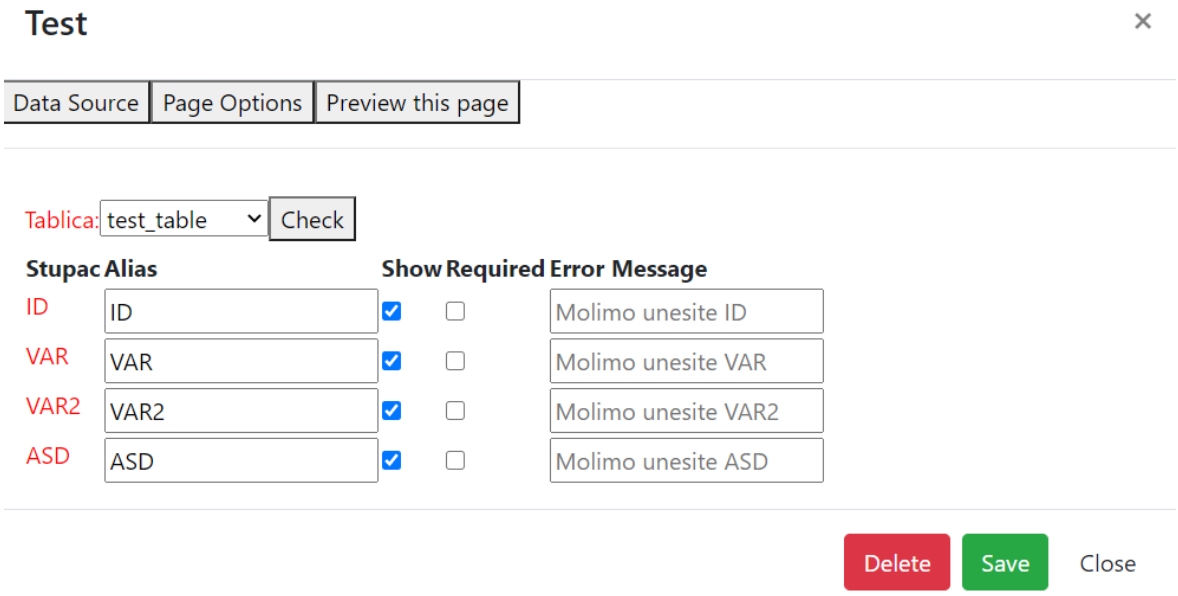
Nadalje, pritiskom na novo kreiranu stranicu *Test* ponovno se otvara skočni izbornik koji sadrži tri opcije stranice: izvor podataka, opcije stranice i pregled stranice (Slika 5.10.).



Slika 5.10. Prikaz opcija kreirane stranice

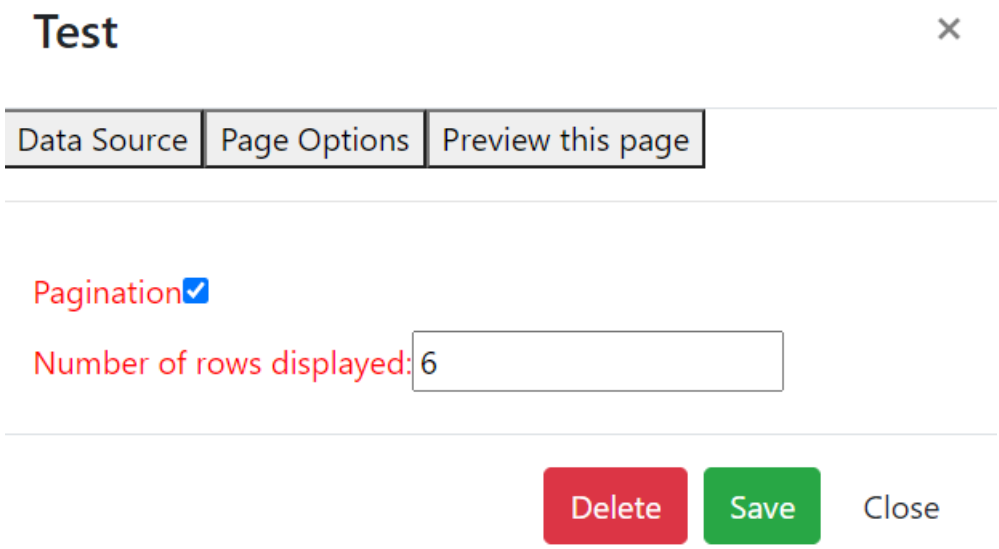
Odabirom opcije Izvor podataka unutar skočnog izbornika pojavljuje se padajući izbornik sa izborom tablica iz kojih želimo prikazati podatke te se klikom na gumb „CHECK“ projerava dostupnost tablice i dohvaćaju se podatci o nazivima stupaca i

ograničenjima vezanih uz izabranu tablicu. Na Slici 5.11. prikazane su opcije koje korisnik može proizvoljno uređivati poput aliasa koji označava naziv stupca čija je prvobitna vrijednost ista nazivu stupca unutar odabrane tablice te se može izmjeniti u bilo koji naziv. Također, prilikom pregleda stranice ili pokretanja aplikacije mijenja se naziv stupca u naziv koji je zapisan pod aliasom. Nadalje, generator evidencije poslovanja također sadrži opciju „SHOW“ koja je potvrdni okvir preko koje se može odabrati koji stupci će biti prikazani unutar stranice. Opcija „REQUIRED“ ima zadaću, prilikom unosa podataka, provjeriti jesu li podatci važeći te postoje li tj. onemogućava neodgovarajuć unos podataka u tablicu te tu dolazi opcija „ERROR MESSAGE“ koja omogućava unos proizvoljne poruke prilikom krivog unosa podataka.



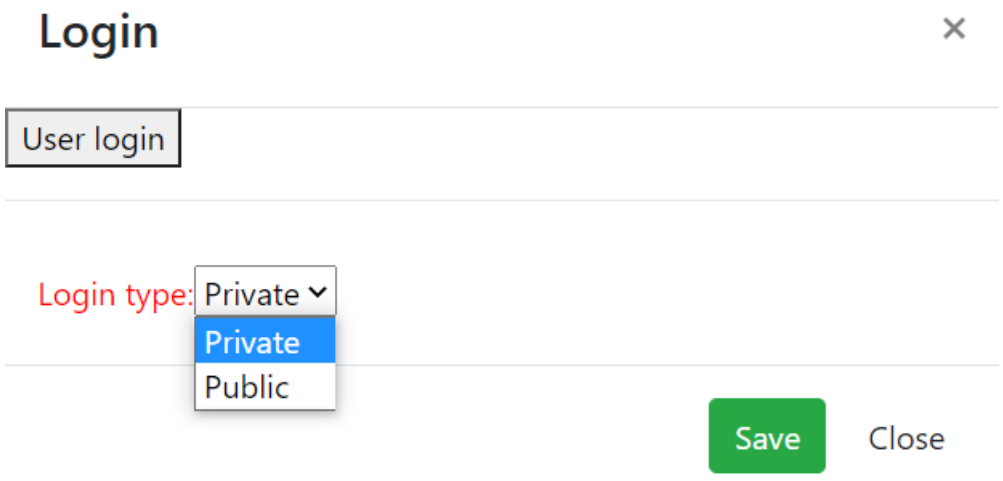
Slika 5.11. Prikaz odabira izvora podataka

Iduća opcija na skočnom izborniku je „Opcija stranice“ koja omogućava podešavanje količine prikazanih podataka na stranici odabiom potvrdnog okvira te unosom brojčane vrijednosti podataka željnih za prikaz (Slika 5.12.). Također na skočnom izborniku se nalazi i gumb „DELETE“ koji služi da brisanje trenutno odabrane stranice koja omogućava da jednim klikom na navedeni gumb stranica više ne postoji.



Slika 5.12. Prikaz opcija stranice

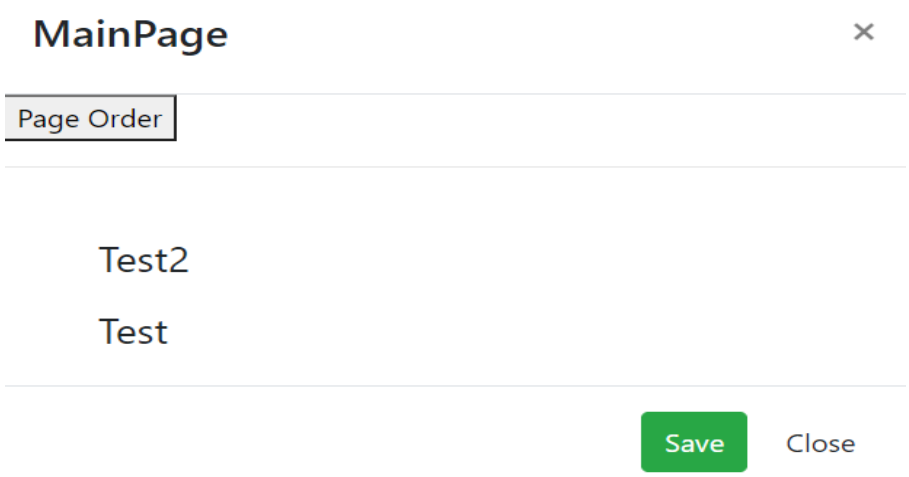
Odabirom stranice *Login* koja je prikazana na Slici 5.13. moguće je odabrati tip prijave na stranicu koji može biti privatan ili javan. Privatan tip podrazumijeva prijavu samo od strane korisnika koji je kreirao aplikaciju, a odabirom javnog tipa pojavljuje se padajući izbornik tablica koje je moguće odabrati koji nudi mogućnost prijave korisnika iz odabrane tablice.



Slika 5.13. Prikaz odabira načina prijave

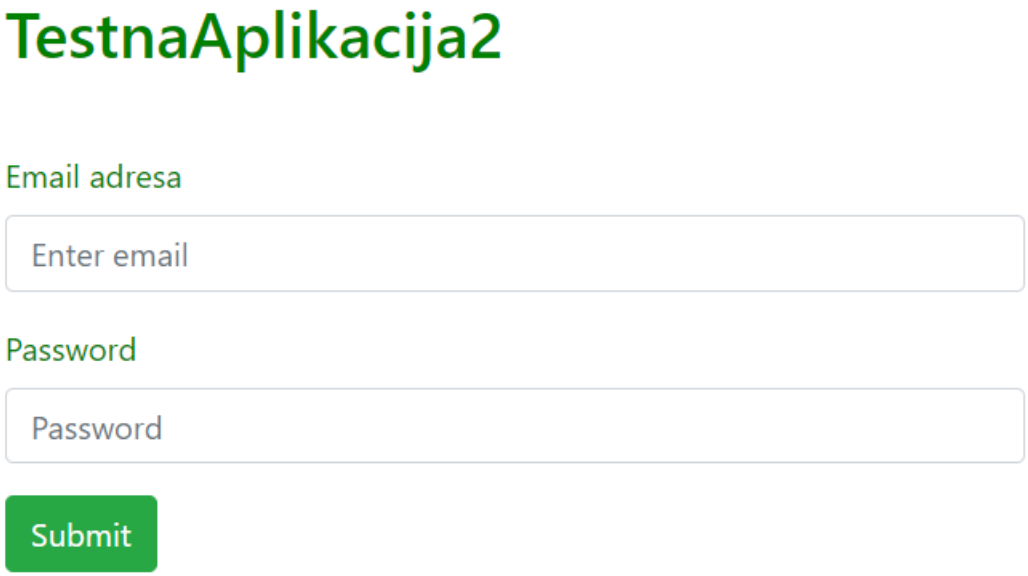
Odabirom stranice *Mainpage* pojavljuje se skočni izbornik prikazan koji omogućava odabir redoslijeda prikaza stranica prilikom pokretanja odabrane aplikacije. Odabir redoslijeda funkcionira na principu povlačenja i ispuštanja tako da se pritisne na

željenu stranicu kojoj se želi promjeniti redoslijed te se potom povuče ispod ili iznad drugih stranice te ju se ispusti (Slika 5.14.).



Slika 5.14. Prikaz opcija glavnog menija

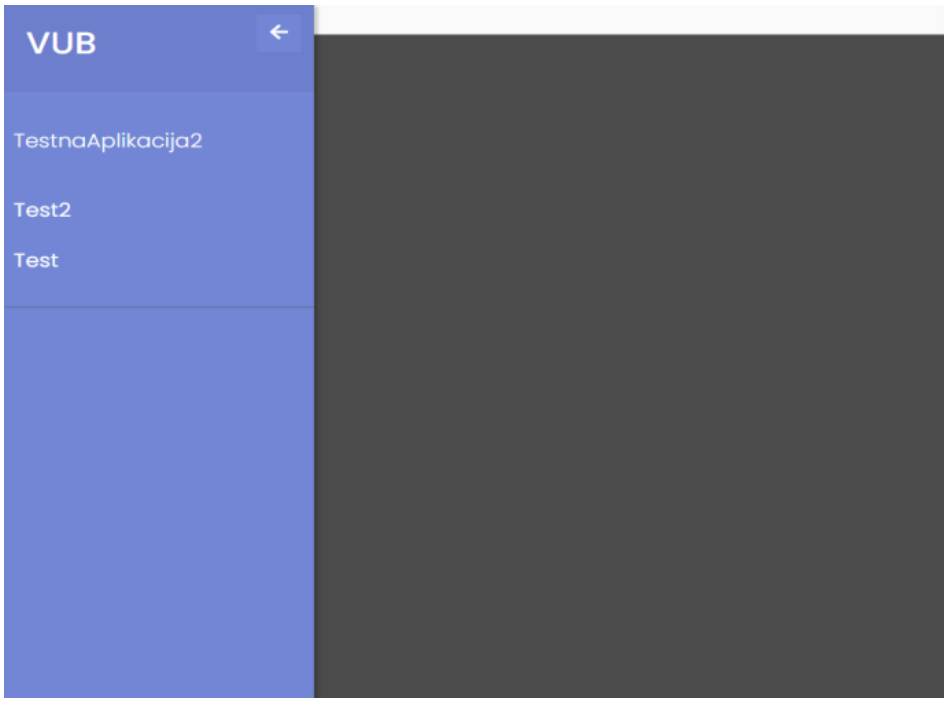
Nadalje, pritiskom na gumb „RUN“ otvara se nova kartica u web pregledniku te se pojavljuje forma za prijavu u odabranu aplikaciju kao što je prikazano na Slici 5.15. Pošto je na login stranici odabran privatni tip prijave potrebno je unijeti identične podatke za prijavu kao i na web sučelje generatora evidencije poslovanja.



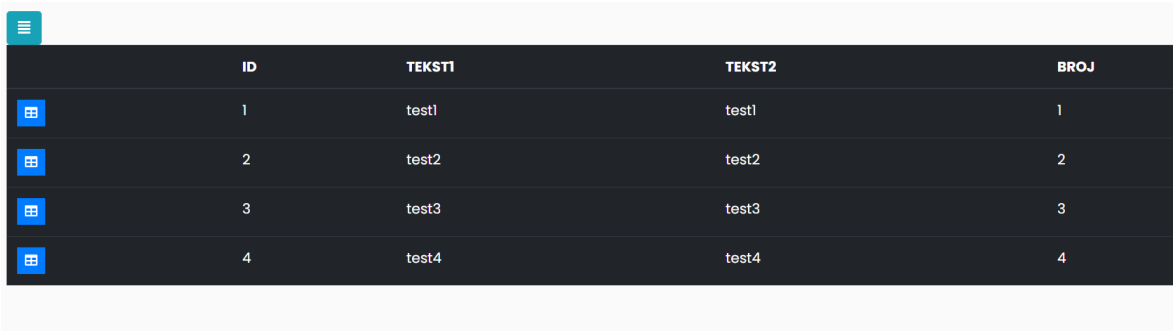
Slika 5.15. Prikaz prijave u kreiranu aplikaciju

Nakon unosa traženih podataka prikazuje se početna stranica aplikacije zajedno s bočnom navigacijskom trakom (Slika 5.16.). Na navedenoj bočnoj navigacijskoj traci

prikazan je naziv aplikacije kreirane stranice unutar aplikacije. Pritiskom na stranicu „Test“ pojavljuje se tablica s odabranim podatcima (Slika 5.17.).



Slika 5.16. Prikaz navigacijske trake unutar pokrenute aplikacije



ID	TEKST1	TEKST2	BROJ
1	test1	test1	1
2	test2	test2	2
3	test3	test3	3
4	test4	test4	4

Slika 5.17. Prikaz izgleda stranice tipa *Forma*

5.2 JavaScript

JavaScript koji je također poznat pod nazivom JS je objektno orijentiran programski jezik. JavaScript je ujedno i jedan od najpoznatijih skriptnih jezika za razvoj web stranice, ali se koristi i u mnogim drugim okruženjima. Budući da se JavaScript pokreće na klijentskoj strani tj. u web pregledniku, time omogućuje izmjenu sadržaja web stranice bez njenog ponovnog učitavanja te čini web stranicu dinamičnom [8]. Web sučelje generatora evidencije poslovanja namijenjeno je funkciji web aplikacije te je upravo zbog toga izabran

JavaScript koji nudi dinamična svojstva izmjene sadržaja web stranice bez ponovnog učitavanja cijele stranice. Upravo zbog toga što web stranicu čini dinamičnom, rad cijelog web sučelja generatora evidencije poslovanja temeljen je na JavaScript-u koji izmijenjuje sadržaj u već navedenom HTML-u [11].

5.2.1 *Primjena JavaScript-a*

Prethodno je navedeno kako se web sučelje generatora evidencije poslovanja dinamički generira te time se dolazi do korištenja JavaScript-a koji se nalazi u datoteci pod nazivom „handlePages“. Datoteka „handlePages“ zadužena je za svu funkcionalnost web sučelja. Prilikom učitavanja web sučelja generira se stranica za prijavu (Slika 5.18.) pri čemu se poziva funkcija „loadLoginPage“ koja je prikazana na Slici 5.19. Funkcija „loadLoginPage“ u prethodno navedeni spremnik umeće podatke koje dobije iz funkcije „loginForm“ (Slika 5.20.) te se na web sučelju prikazuje forma za prijavu u web sučelje koje je prethodno detaljnije prikazano u prethodnim poglavljima.

```
$(document).ready(function (e) {
    loadLoginPage();
    sessionStorage.setItem("hash","");
});
```

Slika 5.18. Prikaz učitavanja web sučelja

```
function loadLoginPage(){
    $("#container").html(loginForm);
}
```

Slika 5.19. Prikaz umetanja podataka na web sučelje

```
function loginForm() {
    var output = "<br><br><form><div class='form-group'>";

    output += '<button type="button" id ="btnTestPage">TestNewPage</button>';

    output += '<button type="button" id ="btnFastLogin">FastLogin</button>';

    output += "<label for='inputEmail'>Email adresa</label>";
    output += "<input type='email' class='form-control' id='inputEmail' aria-describedby='emailHelp' placeholder='Enter email'></div>";
    output += "<div class='form-group'><label for='inputPassword'>Password</label>";
    output += "<input type='password' class='form-control' id='inputPassword' placeholder='Password'></div>";
    output += "<button type='button' class='btn btn-primary' id='btnSubmit'>Submit</button></form>";
    return output;
}
```

Slika 5.20. Prikaz izgleda koda stranice za prijavu

Na prethodno opisanom principu funkcionira cijelo web sučelje pri čemu je izgled sučelja definiran unutar funkcija koje prilikom njihovog pozivanja vraćaju HTML kod koji se potom ubacuje u prethodno kreirani spremnik unutar HTML datoteke. Nadalje, kada se na web sučelju pojavi forma za prijavu korisnika te kada se unesu traženi podatci, pritiskom na gumb „SUBMIT“ okida se dio koda (Slika 5.21.) pri čemu se provjerava jesu li traženi podatci uneseni te ako jesu, poziva se funkcija *login* koja vraća vrijednost *true* ili *false* na temelju dobivenih podataka. Ako funkcija *login* vrati vrijednost *true*, tada se poziva funkcija *mainScreen* koja generira glavni izbornik na prethodno opisanom principu za prijavu na web sučelje.

```
$(document).on('click', '#btnSubmit', function () {
    var email = $('#inputEmail').val();
    var password = $('#inputPassword').val();
    if (email == null || email == "") {
        Swal.fire('Molimo unesite email adresu');
    } else if (password == null || password == "") {
        Swal.fire('Molimo unesite zaporku');
    } else {
        var passed = login(email, password);
        if (passed){
            mainScreen();
            pageData.currentPage = "mainScreen";
            document.title = "Vuxon";
        }
        else{
            displayError();
        }
    }
})
```

Slika 5.21. Prikaz funkcije za prijavu u web sučelje

Funkcija *login* pri prijavi u web sučelje prima dva parametra koji su potrebni za daljni rad web sučelja. Podatci koji su proslijeđeni tijekom unosa podataka tijekom prijave, potom se spajaju znakom dolara („$“) te tako se spojeni podatci spremaju u varijablu naziva „params“. Naime, svi parametri koji se šalju prema bazi podataka spojeni su sa znakom dolara radi jednostavnijeg baratanja s njima te poziva prema bazi koji se odvija u JSON formatu. Tako spojeni podatci dalje se prosljeđuju funkciji „callDB“ čiji je glavni zadatak komunikacija s bazom podataka te se zajedno s navedenim podatcima šalje i podatak o proceduri koja se treba izvršiti na bazi podataka koja je u ovom slučaju p_login. Nadalje, kada baza izvrši potrebnu proceduru, podatci se vraćaju u obliku JSON formata i spremaju u varijablu naziva json. Također, unutar „IF“ uvjeta koristeći funkciju *errorControll* provjeravaju se dobiveni podatci s baze. Ako nije došlo do pogreške unutar baze podataka i ako su prosljeđeni podatci za prijavu na web sučelje ispravni, tada se u sesiju stranice postavlja podatak „hash“ s vrijednošću generiranom na bazi podataka.

Funkcija *login* tada vraća vrijednost *true* što znači da je provjera podataka prošla (Slika 5.22.).

```
function login(email, password){
    var params = email + "$" + password;
    var json = callDB("p_login", params)

    if (errorControll(json)){
        sessionStorage.setItem("hash", json.o_hash);
        return true;
    }
    else{
        return false;
    }

}
```

Slika 5.22. Prikaz funkcije za prijavu u web sučelje

Nadalje, važno je opisati i dvije funkcije koje su najviše korištene unutar rada web sučelja, a to su *callDB* i *errorControl*. Funkcija *callDB* ima zadatak pripremiti primljene podatke te putem prethodno navedenog AJAX-a poslati ih prema bazi podataka (Slika 5.23.) Kada se potrebna procedura izvrši na bazi, funkcija *callDB* vraća primljenje podatke od strane baze. Unutar navedene funkcije primljeni podatci se pripremaju preko funkcije *parseParams* koja prethodno spojene podatke sa znakom dolara razdvaja te ih vraća spremljene u obliku niza JSON formata.

```
function callDB(i_proc, i_params, i_package = ""){
    var out;
    var params = parseParams(i_params);
    if (i_package == null || i_package == ""){
        i_package = "GENERATOR";
    }
    $.ajax({
        type: 'POST',
        url: "http://localhost/mid/test.php",
        data: { "action":"handleCall" ,"package": i_package , "procedure": i_proc , "parameters" : params},
        success: function (data) {
            // alert(data);
            var jsonBody = JSON.parse(data);

            out = jsonBody;
        },
        async: false

    });
    return out;
}
```

Slika 5.23. Prikaz funkcije poziva baze podataka

Nadalje, zadaća funkcije *errorControl* je provjera dobivenih podataka s baze. Unutar primljenih podataka, koji su u JSON formatu, nalaze se dvije kontrolne varijable pod imenom o_message i o_error koje generira baza prilikom vraćanja podataka. Kontrolne varijable o_message i o_error služe kao inidikatori pogrešaka kojima se glavna funkcija odnosi na provjeru pogrešaka unutar baze. Vrijednost varijable o_error postavlja se na brojčanu vrijednost koja je definirana unutar baze, dok se u o_message nalazi tekstualni opis pogreške koja se dogodila. U slučaju da nije došlo do pogreške, vrijednost varijable o_error postavljena je na 0 te o_message ne sadrži vrijednost. Slika 5.24. prikazuje izgled funkcije *errorControll* u slučaju kada nije došlo do pogreške pri kojemu funkcija vraća vrijednost true te u slučaju kada dolazi do pogreške pri čemu funkcija vraća vrijednost false.

```
function errorControll(i_json){
    var msg = i_json.o_message;
    var err = i_json.o_error;

    if (msg == "" || err == 0){

        return true;
    }
    else{
        if (err >= 300 && err <= 400){
            Swal.fire(
                'Greska',
                msg,
                'error'
            );
        }
        return false;
    }
}
```

Slika 5.24. Prikaz funkcije za provjeru dobivenih podataka s baze podataka

Također, kada je riječ o funkcionalnosti web sučelja, važno je napomenuti kako se prilikom otvaranja stranice unutar aplikacije te pojave skočnog izbornika koji je opisan u prethodnom poglavlju, sve informacije o odabranoj stranici spremaju u varijablu naziva dataJSON koja je prikazana na Slici 5.25. Navedena varijabla, koja je definirana kao objekt, ima vrlo važnu ulogu u konfiguriranju odabrane stranice upravo zato što se svaki podatak koji je potreban za kreiranje te stranice nalazi u istoj. Prilikom svake korisničke izmjene unutar stranice događa se i izmjena unutar varijable dataJson. Primjer izgleda navedene varijable vidjiv je na Slici 5.26.

```
var dataJson = {
    table : "",
    columns: "",
    pagination : false,
    perPage : 6,

}
```

Slika 5.25. Prikaz spremanja informacija o kreiranoj stranici


```
{
    "table":"testnaTablica",
    "columns":{
        "ID":{
            "show":true,
            "required":false,
            "columnType":"NUMBER",
            "nullable":"N",
            "columnLength":22,
            "errMsg":"Molimo unesite ID",
            "alias":"ID"
        },
        "STUPAC1":{
            "show":true,
            "required":false,
            "columnType":"VARCHAR",
            "nullable":"N",
            "columnLength":60,
            "errMsg":"Molimo unesite STUPAC1",
            "alias":"STUPAC1"
        },
        "STUPAC2":{
            "show":true,
            "required":false,
            "columnType":"NUMBER",
            "nullable":"N",
            "columnLength":22,
            "errMsg":"Molimo unesite STUPAC2",
            "alias":"STUPAC2"
        }
    },
    "pagination":true,
    "perPage":"15"
}
```


Slika 5.26. Prikaz podataka prilikom uređivanja stranice

Nadalje, funkcionalnost web sučelja koja se odvija unutar JavaScripta temelji se na prethodno opisanim primjerima pri kojima se izgled web sučelja izmijenjuje ovisno o odabranoj opciji unutar web sučelja [8]. Sve navedene opcije za uređivanje, dodavanje, kreiranje, izmjenu ili brisanje bilo kojeg aspekta generatora evidencije poslovanja koji se odvija unutar skočnog prozora izmijenjuje se na temelju dohvaćenih informacija iz baze podataka.

6. ZAKLJUČAK

Ovaj se rad temelji na sustavu za generiranje aplikacija koji ne mora nužno biti namjenjen samo u svrhu poslovanja nego i u općenitoj primjeni evidencije podataka. Sustav se nalazi unutar baze podataka koja omogućuje izradu web sučelja koje je pak namjenjeno za korištenje izrađenog sustava. Pri izradi sustava korištena je *Oracle XE* (eXpress Edition) baza podataka zajedno s alatom *SQL developer* unutar kojega je razrađen kod za rad baze. Za potrebe korištenja generatora evidencije poslovanja izrađeno je web sučelje koje se zasniva na tehnologijama HTML, JavaScript i CSS. Web sučelje se dinamički generira preko objektno orijentiranog programskog jezika Javascript koji podatke potrebne za prikaz web sučelja dohvaća s baze podataka preko web servera Apache. Dakle, za komunikaciju između baze podataka i web sučelja korištena je platforma XAMPP koja u sebi sadrži navedeni web server Apache koji omogućava komunikaciju između web sučelja i same baze. Također je korišten i skriptni jezik PHP koji ostvaruje vezu s bazom.

Temelj rada čine procedure napisane u integriranom razvojnom okruženju SQL developer koje koriste JSON format koji pojednostavljuje prikaz podataka koji su namjenjeni za slanje prema bazi te dohvaćanje podataka s baze. Zajedno s procedurama koje se koriste za unos, izmjenu i dohvaćanje podataka, koriste se i posebno izrađene tablice unutar kojih se spremaju sve informacije potrebne za rad generatora evidencije poslovanja. Navedene procedure pozivaju se unutar web sučelja koje je posebno kreirano u svrhu vizualnog prikaza rada generatora evidencije poslovanja. Web sučelje se sastoji od spremnika i skočnog prozora koji sadrži sve potrebne opcije za korištenje sustava s baze.

Ograničenja ovoga rada vezana su uz samu bazu na kojoj se temelji rad upravo radi toga što izrađeni sustav radi samo na Oracle-ovoj bazi. Dakle, za korištenje generatora evidencije poslovanja korisnik mora imati postavljenu Oracle bazu te svi podatci koji su namjenjeni za evidenciju moraju se pohranjivatu u nju. Uzevši to u obzir, kako bi se rad dodatno unaprijedio, bilo bi poželjno izraditi dodatan sustav koji bi imao mogućnost odabira željene baze podataka te izrada generatora evidencije poslovanja na istoj. Daljnja unaprijeđenja uključuju izradu samostalne web aplikacije koja je generirana od strane generatora evidencije poslovanja pri čemu bi se moglo direktno pristupiti izrađenoj web aplikaciji umjesto pokretanja iste s web sučelja generatora evidencije poslovanja.

7. LITERATURA

[1] Abu Elsoud M.,El-Bakry H., Hassan A., Kandel M., Mastorakis N., Riad A., Shohieb S. Adaptive user interface for web applications. Recent Advances in Bussines Administration. 2010. str. 190-211.

[2] Agarrwal A., Gupta P., Kumar Goyal M., Patel S. Low Cost Hardware Design of a Web Server for Home Automation Systems. Atlantis Press. 2013. str. 521-525.

[3] Akdaş D., Hüseyin G. Jeftini i modularni kućni automatski sustav utemeljen na Web-u. Tehnički vjesnik. Vol 23. 2016. str. 533-538.

[4] Arcuri J., Hickey M. Databases. Hands on Hacking. 2020.

[5] Chihana S., Kunda D., Muwanei S. Web Server Performance of Apache and Nginx: A Systematic Literature Review. Computer Engineering and Intelligent Systems. Vol 8. 2017. str. 43-52.

[6] Havaš L., Lesar M. Primjena SQL-a u programima otvorenog koda. Tehnički glasnik. Vol 6, 2012. str. 164-170.

[7] JSON: „Introducing JSON“, https://www.json.org/json-en.html 10.10.2020.

[8] Kila C. JavaScript. Website Development. 2015. str. 1-10.

[9] Kotta F. Introducing JSON. Json 2016. str. 1-10.

[10] Lozić D.; Šimec A., Tepeš Golubić L. Mjerenje brzine rada php modula. Informatologija. Vol 50. 2017. str. 95-100.

[11] Mozilla: „JavaScript“, https://developer.mozilla.org/en-US/docs/Web/JavaScript 10.10.2020.

[12] Mozilla: „What is a web server?“, https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_web_server 10.10.2020.

[13] Muteti M. How To Download And Install XAMPP On Windows 7 64-Bit. 2019.

[14]Oracle: „Oracle SQL Developer“, https://www.oracle.com/database/technologies/appdev/sqldeveloper-landing.html (10.10.2020.)

[15] O'Regan G. Pillars of Computing A Compendium of Select, Pivotal Technology Firms. Springer International Publishing. 2015.

[16] Sharma A. Introduction to HTML (Hyper Text Markup Language) - A Review Paper. International Journal of Science and Research. Vol 7. 2018. str. 1337-1339.

[17] Taipalus T., Seppänen V. SQL Education: A Systematic Mapping Study and Future Research Agenda. ACM Transactions on Computing Education. Vol. 20. 2020. str. 1946-6226.
[18] Queiroz M., Silva Y., Almeida I. SQL: From Traditional Databases to Big Data. Conference paper. 2016. str. 1-6.

8. OZNAKE I KRATICE

CSS – Cascading Style Sheet

DBA – DataBase Administrator

HTML – HyperText Markup Language

HTTP – HyperText Transfer Protocol

IBM – International Business Machines

JS – JavaScript

JSON – JavaScript Object Notation

PHP – Hypertext Preprocessor

PL/SQL – Language/SQL

RSI – Relational Software, Inc

SDL – Development Laboratories

SQL – Structured Query Language

SUBP – Sustav za Upravljanje bazom podataka

URL – Uniform Resource Locator

XE – eXpress Edition

XML – eXtensible Markup Language

9. SAŽETAK

Naslov: Generator evidencije poslovanja

Ovaj rad predstavlja sustav na Oracle bazi podataka za generiranje aplikacija za vođenje evidencije poslovanja pri čemu se sustav temelji na navedenoj bazi koja omogućuje pregled, unos, izmjenu te brisanje željenih podataka. Sustav podrazumijeva kreiranje novih aplikacija unutar kojih je moguće dodati stranice koje se koriste za pregled podataka iz odabrane tablice u obliku forme ili izvještaja pri čemu se može odabrati koji će se podatci prikazati te na koji način će izgledati prikaz odabranih podataka. Pregled podataka u obliku fome podrazumijeva pregled podataka u obliku tablice te mogućnosti unosa, izmjene te brisanja podataka, dok se pregled podataka u obliku izvještaja prikazuje u obliku tablice ili grafa raznih oblika i omogućuje prikaz skupa sume, prosjeka ili zbroja odabranih podataka. Za sustav generatora evidencije poslovanja izrađeno je web sučelje koje služi za prikaz funkcionalnosti izrađenog sustava.

Ključne riječi: generator, web sučelje, baza podataka, web aplikacija.

10. ABSTRACT

Title: Generating business management application

This paper represents the system based on the Oracle database which aim is generating applications for bussines management that is based on mentioned database that enables previewing, insertion, updating and erasing data. System implements creating new applications that has an ability to add pages used for viewing data from choosen table in the shape of form or report among which it is possible to choose which data will be listed and on which way. Viewing data in the shape of form understands preview of the data in the shape of table and possibility of inserting, updating and erasing data. In contrast to that, viewing data in the shape of report understands preview of data in the shape of table or many different shapes of graphs and enables insight in various aggregations of sum, average or count of choosen data. For system of generating bussines managemenet application it has been made web interface which purpose is to enable viewing functionality of whole system.

Keywords: generator, web interface, database, web application.

IZJAVA O AUTORSTVU ZAVRŠNOG RADA

Pod punom odgovornošću izjavljujem da sam ovaj rad izradio/la samostalno, poštujući načela akademske čestitosti, pravila struke te pravila i norme standardnog hrvatskog jezika. Rad je moje autorsko djelo i svi su preuzeti citati i parafraze u njemu primjereno označeni.

Mjesto i datum	Ime i prezime studenta/ice	Potpis studenta/ice
U Bjelovaru, 13.10.2020	DOMINIK ERMAK	Ermak

Prema Odluci Veleučilišta u Bjelovaru, a u skladu sa Zakonom o znanstvenoj djelatnosti i visokom obrazovanju, elektroničke inačice završnih radova studenata Veleučilišta u Bjelovaru bit će pohranjene i javno dostupne u internetskoj bazi Nacionalne i sveučilišne knjižnice u Zagrebu. Ukoliko ste suglasni da tekst Vašeg završnog rada u cijelosti bude javno objavljen, molimo Vas da to potvrdite potpisom.

Suglasnost za objavljivanje elektroničke inačice završnog rada u javno dostupnom nacionalnom repozitoriju

DOMINIK ERNJAK

ime i prezime studenta/ice

Dajem suglasnost da se radi promicanja otvorenog i slobodnog pristupa znanju i informacijama cjeloviti tekst mojeg završnog rada pohrani u repozitorij Nacionalne i sveučilišne knjižnice u Zagrebu i time učini javno dostupnim.

Svojim potpisom potvrđujem istovjetnost tiskane i elektroničke inačice završnog rada.

U Bjelovaru, 18.10.2020

potpis studenta/ice