



УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У
НОВОМ САДУ




Бојан Давогић

Систем за обраду и визуелизацију података у заводу за статистику

СЕМИНАРСКИ РАД
- Основне струковне студије -

Нови Сад, 2024.

	<p>УНИВЕРЗИТЕТ У НОВОМ САДУ • ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА 21000 НОВИ САД, Трг Доситеја Обрадовића 6</p> <p>КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА</p>
---	---

Редни број, РБР:		
Идентификациони број, ИБР:		
Тип документације, ТД:	Монографска публикација	
Тип записа, ТЗ:	Текстуални штампани материјал	
Врста рада, ВР:	Семинарски рад	
Аутор, АУ:	Бојан Давогић	
Ментор, МН:	Марко Марковић	
Наслов рада, НР:	Систем за обраду и визуелизацију података у заводу за статистику	
Језик публикације, ЈП:	Српски	
Језик извода, ЈИ:	Српски	
Земља публикавања, ЗП:	Република Србија	
Уже географско подручје, УГП:	АП Војводина	
Година, ГО:	2024.	
Издавач, ИЗ:	Ауторски репринт	
Место и адреса, МА:	Нови Сад, Трг Доситеја Обрадовића 6	
Физички опис рада, ФО: <small>(поглавља/страна/ цитата/табела/слика/графика/прилога)</small>	8/24/7/1/6/0/0	
Научна област, НО:	Електротехничко и рачунарско инжењерство	
Научна дисциплина, НД:	Технологије и системи еУправе	
Предметна одредница/Кључне речи, ПО:	Завод за статистику, еУправа, веб апликација	
УДК		
Чува се, ЧУ:	Библиотека Факултета техниких наука, Универзитет у Новом Саду	
Важна напомена, ВН:	Нема	
Извод, ИЗ:	У овом семинарском раду описана је имплементација информационог система завода за статистику, који укључује претрагу и приказ статистичких података о саобраћајним прекршајима и регистрованим возилима.	
Датум прихватања теме, ДП:		
Датум одбране, ДО:		
Чланови комисије, КО:	Председник:	
	Члан:	Потпис ментора
	Члан, ментор:	




UNIVERSITY OF NOVI SAD • FACULTY OF TECHNICAL SCIENCES
21000 NOVI SAD, Trg Dositeja Obradovića 6

KEY WORDS DOCUMENTATION

Accession number, ANO :	
Identification number, INO :	
Document type, DT :	Monographic publication
Type of record, TR :	Textual printed material
Contents code, CC :	Seminar paper
Author, AU :	Bojan Davogić
Mentor, MN :	Marko Marković
Title, TI :	System for data processing and visualization in the Institute for Statistics.
Language of text, LT :	Serbian
Language of abstract, LA :	Serbian/English
Country of publication, CP :	Republic of Serbia
Locality of publication, LP :	Autonomous Province of Vojvodina
Publication year, PY :	2024.
Publisher, PB :	Author's reprint
Publication place, PP :	Novi Sad, Trg Dositeja Obradovića 6
Physical description, PD : (chapters/pages/ref./tables/pictures/graphs/appendixes)	8/24/7/1/6/0/0
Scientific field, SF :	Electrical and computer engineering
Scientific discipline, SD :	Technologies and systems of eUprava
Subject/Key words, S/KW :	Institute for Statistics, react, docker, golang, mongoDB
UC	
Holding data, HD :	Library of the Faculty of Technical Sciences, University of Novi Sad
Note, N :	None
Abstract, AB :	This seminar paper describes the implementation of the information system of the Institute for Statistics, which includes the search and display of statistical data on traffic violations and registered vehicles.
Accepted by the Scientific Board on, ASB :	

Defended on, DE:		
Defended Board, DB:	President:	
	Member:	Menthor's sign
	Member, Mentor:	

	УНИВЕРЗИТЕТ У НОВОМ САДУ • ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА 21000 НОВИ САД, Трг Доситеја Обрадовића 6	Датум:
	ЗАДАТАК ЗА ИЗРАДУ ДИПЛОМСКОГ (BACHELOR) РАДА	Лист/Листова:
		1/1

(Податке уноси предметни наставник - ментор)

Врста студија:	<input checked="" type="checkbox"/> Основне струковне студије
Студијски програм:	Софтверске и информационе технологије
Руководилац студијског програма:	др Синиша Николић

Студент:	Бојан Давогић	Број индекса:	CP 27/2021
Област:	ЕЛЕКТРОТЕХНИЧКО И РАЧУНАРСКО ИНЖЕЊЕРСТВО		
Ментор:	Марко Марковић		
НА ОСНОВУ ПОДНЕТЕ ПРИЈАВЕ, ПРИЛОЖЕНЕ ДОКУМЕНТАЦИЈЕ И ОДРЕДБИ СТАТУТА ФАКУЛТЕТА ИЗДАЈЕ СЕ ЗАДАТАК ЗА ДИПЛОМСКИ (Bachelor) РАД, СА СЛЕДЕЋИМ ЕЛЕМЕНТИМА:			
<div>- проблем – тема рада;</div> <div>- начин решавања проблема и начин практичне провере резултата рада, ако је таква провера неопходна; - литература</div>			

НАСЛОВ СЕМИНАРСКОГ РАДА:

Систем за обраду и визуелизацију података у заводу за статистику

ТЕКСТ ЗАДАТКА:

У оквиру семинарског рада потребно је:

- Представити савремене трендове у области статистичке анализе података, са освртом на примену у државним институцијама и посебно у заводу за статистику;
- Описати архитектуру информационог система завода за статистику, укључујући структуру података, проток информација и интеграцију са другим државним институцијама (МУП, саобраћајна полиција, суд);
- Дати кратак приказ коришћених технологија у развоју информационог система, укључујући Golang за backend и React за frontend;
- Дефинисати и имплементирати функције за анализу података о саобраћајним прекршајима и регистрованим возилима, укључујући прикупљање, обраду и визуелизацију података;
- Приказати резултате развоја и тестирања система, укључујући примере корисничких извештаја, претрага и генерисања PDF докумената са статистичким подацима; □ Извести одговарајуће закључке.

Приликом израде рада користити одговарајућу литературу из области.

Руководилац студијског програма:	Ментор рада:
Синиша Николић	Марко Марковић

Примерак за: ☐ - Студента; ☐ - Ментора

Образац Q2.HA.15-04 - Издање

SADRŽAJ

1. Uvod	7
2. Srodna rešenja i pregled korišćenih tehnologija.....	Greška! Obeleživač nije definisan.
2.1 Srodna rešenja	8
2.2 Korišćene tehnologije	Greška! Obeleživač nije definisan.
3. Specifikacija zahteva	Greška! Obeleživač nije definisan.
3.1 Funkcionalni zahtevi	Greška! Obeleživač nije definisan.
3.2 Nefunkcionalni zahtevi	Greška! Obeleživač nije definisan.
4. Specifikacija dizajna.....	Greška! Obeleživač nije definisan.
4.1 Arhitektura sistema.....	Greška! Obeleživač nije definisan.
4.2 Model podataka	Greška! Obeleživač nije definisan.
5. Implementacija	Greška! Obeleživač nije definisan.
5.1 Implementacija funkcije za izveštaj o saobraćajnim prekršajima	Greška! Obeleživač nije definisan.
5.2 Implementacija funkcije za broj registrovanih vozila po godinama.	Greška! Obeleživač nije definisan.
6. Demonstracija	Greška! Obeleživač nije definisan.
6.1 Scenario korišćenja.....	Greška! Obeleživač nije definisan.
7. Zaključak.....	22
7.1 Analiza postignutih rezultata	Greška! Obeleživač nije definisan.
7.2 Poređenje sa srodnim rešenjima	Greška! Obeleživač nije definisan.
7.3 Predlozi za buduća rešenja.....	Greška! Obeleživač nije definisan.
8. Literatura	24

1. UVOD

U ovom seminarskom radu opisan je razvoj informacionog sistema za zavod za statistiku, sa fokusom na integraciju i obradu podataka o saobraćajnim prekršajima i registrovanim vozilima. Problem koji se rešava ovim radom jeste složenost obrade velikih količina podataka, što bez automatizovanog sistema zahteva značajne resurse i vreme. Motivacija za razvoj ovakvog sistema proizilazi iz potrebe za boljom dostupnošću i efikasnošću u radu državnih institucija, kao i za tačnijim i bržim donošenjem odluka koje utiču na bezbednost i organizaciju saobraćaja.

Istorijski gledano, problem obrade i analize velikih količina podataka postoji od pojave prvih statističkih zavoda, ali je tehnološki napredak omogućio razvoj savremenih rešenja koja značajno olakšavaju ovaj proces. Ostatak rada je organizovan na sledeći način: Drugo poglavlje predstavlja pregled trendova u oblasti statistike i tehnologija koje se koriste u obradi podataka. Treće poglavlje opisuje arhitekturu i razvoj sistema. U četvrtom poglavlju detaljno je objašnjena metodologija rada. Peto poglavlje pokriva analizu i prikaz podataka. U šestom poglavlju su prikazani rezultati i diskusija, dok sedmo poglavlje donosi zaključke i pravce budućih istraživanja.

2. SRODNA REŠENJA I PREGLED KORIŠĆENIH TEHNOLOGIJA

U ovom poglavlju je dat pregled postojećih rešenja za obradu i analizu saobraćajnih prekršaja i registrovanih vozila, kao i tehnologije koje omogućavaju rešavanje ovih problema. Opisana rešenja i tehnologije pružaju osnovu za razumevanje kako su slični problemi rešavani u drugim kontekstima i koje su prednosti i mane korišćenih pristupa.

2.1 Srodna rešenja

Ovaj odeljak donosi pregled aplikacija namenjenih za obradu i analizu podataka o saobraćajnim prekršajima i registrovanim vozilima.

Informacioni sistem zavoda za statistiku u Nemačkoj

Informacioni sistem zavoda za statistiku u Nemačkoj koristi napredne algoritme za analizu podataka iz različitih sektora, uključujući saobraćaj. Ovaj sistem je integrisan sa drugim državnim institucijama, omogućavajući razmenu podataka i donošenje informisanih odluka. Sistem obezbeđuje tačne i pravovremene informacije koje pomažu u planiranju i implementaciji saobraćajnih politika. Ipak, složenost integracije sa postojećim sistemima drugih institucija može predstavljati izazov.

Platforma za analizu podataka o saobraćaju u Kanadi

Kanadski zavod za statistiku koristi platformu zasnovanu na mašinskom učenju za predikciju saobraćajnih trendova i identifikaciju kritičnih tačaka u saobraćajnoj infrastrukturi. Ova platforma analizira podatke o saobraćajnim prekršajima i registrovanim vozilima kako bi pružila precizne i brze informacije za donošenje odluka. Prednost ovog sistema je njegova sposobnost predikcije budućih saobraćajnih problema, dok je njegov nedostatak u visokoj ceni implementacije i održavanja .

Informacioni sistem zavoda za statistiku u Srbiji

Zavod za statistiku Srbije koristi različite metode za prikupljanje i analizu podataka, ali postoji potreba za modernizacijom i integracijom sistema sa drugim državnim institucijama kao što su MUP, saobraćajna policija i sud. Trenutni sistem omogućava osnovnu obradu podataka, ali nedostaju napredne funkcije za analizu i vizualizaciju. Planirana modernizacija ima za cilj da poboljša efikasnost i tačnost obrade podataka, što će omogućiti bolje donošenje odluka u oblasti saobraćaja .

2.2 Korišćene tehnologije

Ovaj odeljak pruža pregled i objašnjenja tehnologija pomoću kojih je moguće rešavanje problema obrade i analize saobraćajnih podataka.

Golang

Golang [4] je proceduralni i statički otkucani programski jezik koji ima sintaksu sličnu programskom jeziku C. Ponekad se naziva programskim jezikom Go. Pruža bogatu standardnu biblioteku, sakupljanje smeća i mogućnost dinamičkog kucanja. Razvili su ga 2007. Robert Grizemer, Rob Pajk i Ken Thompson iz Gugla, ali je lansiran 2009. kao programski jezik otvorenog koda i uglavnom se koristi u Google-ovim proizvodnim sistemima. Golang je jedan od najpopularnijih programskih jezika među programerima.

React

React [5] je moćna JavaScript biblioteka za izgradnju dinamičkih i interaktivnih korisničkih interfejsa (UI). Razvija ga Facebook. React je poznat po svojoj arhitekturi zasnovanoj na komponentama koja vam omogućava da kreirate elemente korisničkog interfejsa za višekratnu upotrebu, čineći složene veb aplikacije lakšim za upravljanje i održavanje. React se koristi za pravljenje aplikacija na jednoj stranici.

MongoDB

MongoDB [6] je program baze podataka koji je dostupan na više platformi za dokumente za skladištenje velikog obima. Klasifikovan kao NoSQL program baze podataka, MongoDB koristi dokumente slične JSON-u sa opcionim šemama.

Docker

Docker [7] je otvorena platforma za razvoj, isporuku i pokretanje aplikacija. Docker vam omogućava da odvojite svoje aplikacije od infrastrukture kako biste mogli brzo da isporučite softver. Uz Docker, možete upravljati infrastrukturom na isti način na koji upravljate svojim aplikacijama. Korišćenjem prednosti Docker-ovih metodologija za isporuku, testiranje i primenu koda, možete značajno smanjiti kašnjenje između pisanja koda i njegovog pokretanja u proizvodnji.

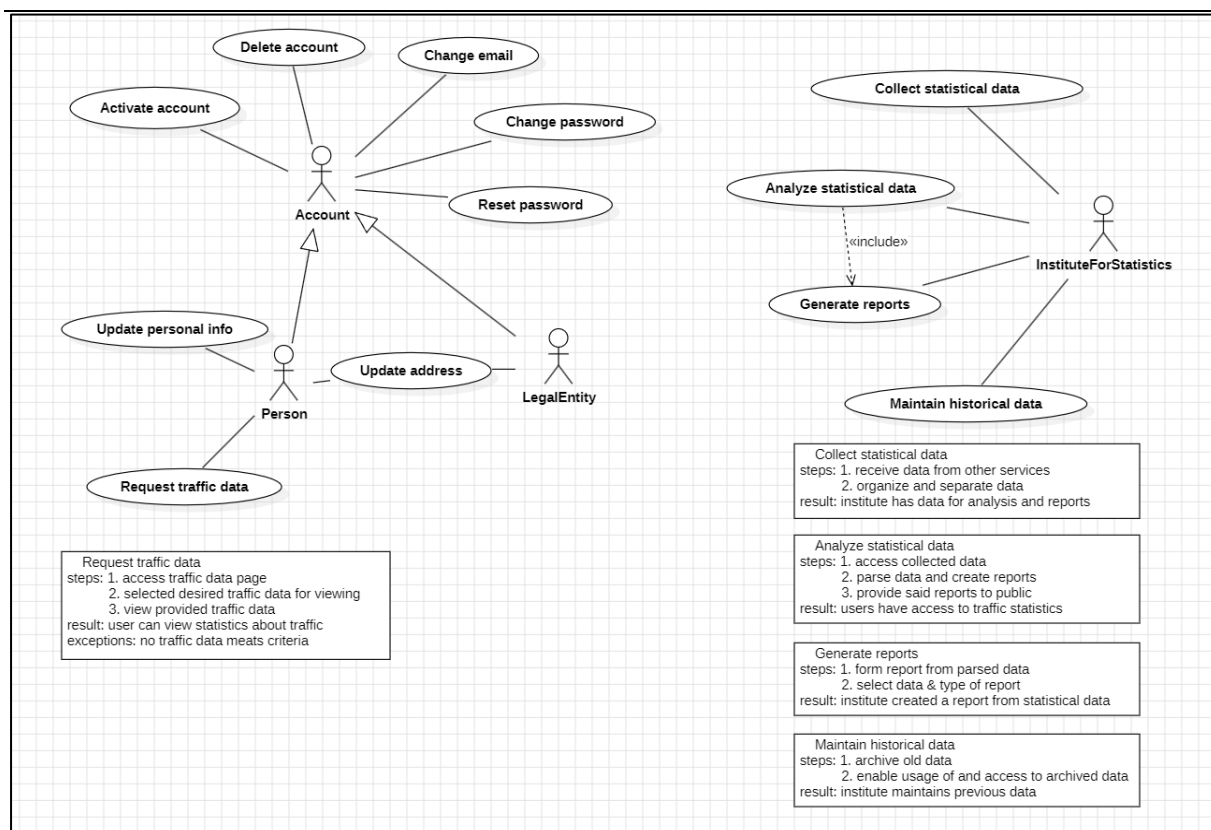
Docker pruža mogućnost pakovanja i pokretanja aplikacije u labavo izolovanom okruženju koje se zove kontejner. Izolacija i bezbednost vam omogućavaju da pokrenete više kontejnera istovremeno na datom hostu. Kontejneri su lagani i sadrže sve što je potrebno za pokretanje aplikacije, tako da ne morate da se oslanjate na ono što je instalirano na hostu. Možete da delite kontejnere dok radite i budete sigurni da svi sa kojima delite dobijaju isti kontejner koji funkcioniše na isti način.

3. SPECIFIKACIJA ZAHTEVA

U ovom poglavlju su objašnjeni funkcionalni i nefunkcionalni zahtevi softverskog rešenja predstavljenog u ovom radu. Specifikacija zahteva dokumentuje sva očekivanja budućeg korisnika softverskog rešenja. Sve što je specificirano u zahtevima mora biti realizovano u rešenju, dok će se sve što nije specificirano smatrati nepotrebnim za sistem.

3.1 Funkcionalni zahtevi

U ovom odeljku su opisani funkcionalni zahtevi koje je potrebno da ispunjava softversko rešenje za zavod za statistiku. Funkcionalni zahtevi ovog softverskog rešenja su predstavljeni UML dijagramom slučajeva korišćenja, kao što je prikazano na slici ispod (Slika 1).



Slika 1 - Dijagram slučajeva korišćenja za sistem zavoda za statistiku.

Dijagram slučajeva korišćenja prikazuje funkcionalne zahteve sistema za zavod za statistiku. Na dijagramu su prikazani akteri, njihove interakcije sa sistemom, kao i koraci potrebni za izvršavanje određenih funkcija. Slučajevi korišćenja su grupisani po akterima i aktivnostima koje ti akteri mogu izvoditi. Dijagram je podeljen na dva dela: funkcionalnosti vezane za korisnički nalog i funkcionalnosti vezane za zavod za statistiku.

Opis slučaja korišćenja pretrage saobraćajnih prekršaja

Slučaj korišćenja pretrage saobraćajnih prekršaja predstavlja jednu od ključnih funkcionalnosti sistema za statistiku. Ova funkcionalnost omogućava korisnicima, kao što su službenici zavoda za statistiku, saobraćajne policije i drugih relevantnih institucija, da efikasno pristupe i analiziraju podatke o saobraćajnim prekršajima. Prikupljeni podaci koriste se za generisanje izveštaja, donošenje odluka i formulaciju strategija za poboljšanje saobraćajne sigurnosti i upravljanje saobraćajem.

Pretraga saobraćajnih prekršaja omogućava korisnicima da pristupe podacima o prekršajima koji su zabeleženi u određenom vremenskom periodu, prema različitim kriterijumima kao što su vrsta prekršaja, lokacija, vreme i učesnici. Tabela 1 prikazuje opis slučaja korišćenja "Pretraga saobraćajnih prekršaja".

Slučaj korišćenja	Pretraga saobraćajnih prekršaja
Učesnici	Korisnik
Preduslovi	Korisnik je ulogovan u sistem.
Koraci	1. Korisnik pristupa stranici za pretragu saobraćajnih prekršaja. 2. Korisnik unosi kriterijume za pretragu.
Rezultat	Sistem prikazuje rezultate pretrage
Izuzeci	Nema dostupnih podataka koji ispunjavaju kriterijume pretrage.

Tabela 1. Opis slučaja korišćenja "Pretraga saobraćajnih prekršaja"

3.2 Nefunkcionalni zahtevi

U ovom odeljku opisani su nefunkcionalni zahtevi sistema zavoda za statistiku. Nefunkcionalni zahtevi definišu karakteristike i ponašanje sistema koji nisu direktno povezani sa specifičnim funkcionalnostima, ali su ključni za efikasno, pouzdano i sigurno funkcionisanje sistema. Oni uključuju performanse, sigurnost, upotrebljivost, održavanje, kompatibilnost i druge aspekte kvaliteta sistema.

Performanse

Sistem bi trebalo da ima brz odziv na upite pretrage, čime bi korisnicima omogućio efikasno dobijanje potrebnih informacija. Takođe, sistem treba da bude sposoban da obrađuje velike količine podataka bez gubitka performansi, čime se osigurava njegova pouzdanost i stabilnost čak i pri velikom opterećenju.

Sigurnost

Sigurnost je ključni aspekt svakog sistema, pa tako i ovog. Samo ovlašćeni korisnici treba da imaju pristup podacima i funkcionalnostima sistema, što se postiže putem autentifikacije i autorizacije.

Upotrebljivost

Interfejs sistema treba da bude intuitivan i lak za upotrebu kako bi korisnici mogli brzo i jednostavno da obavljaju potrebne operacije.

Održavanje

Održavanje sistema treba da bude jednostavno i efikasno. Modularni dizajn omogućava lako proširenje funkcionalnosti i popravke bez uticaja na ostatak sistema. Detaljna tehnička i korisnička dokumentacija je neophodna kako bi se olakšalo korišćenje i održavanje sistema.

Pouzdanost

Pouzdanost sistema je od suštinskog značaja. Sistem treba da bude visoko dostupan, sa minimalnim vremenom neaktivnosti, kako bi korisnici mogli neprekidno da koriste njegove usluge. Takođe, sistem treba da ima sposobnost oporavka od grešaka sa minimalnim uticajem na korisnike, što osigurava kontinuitet rada.

4. SPECIFIKACIJA DIZAJNA

Ovo poglavlje objašnjava dizajn softverskog rešenja za sistem zavoda za statistiku. U narednim delovima će biti objašnjena organizacija sistema, njegove komponente, i način na koji međusobno komuniciraju. Takođe, biće opisane pojedinosti dizajna sistema koristeći UML dijagrame klasa.

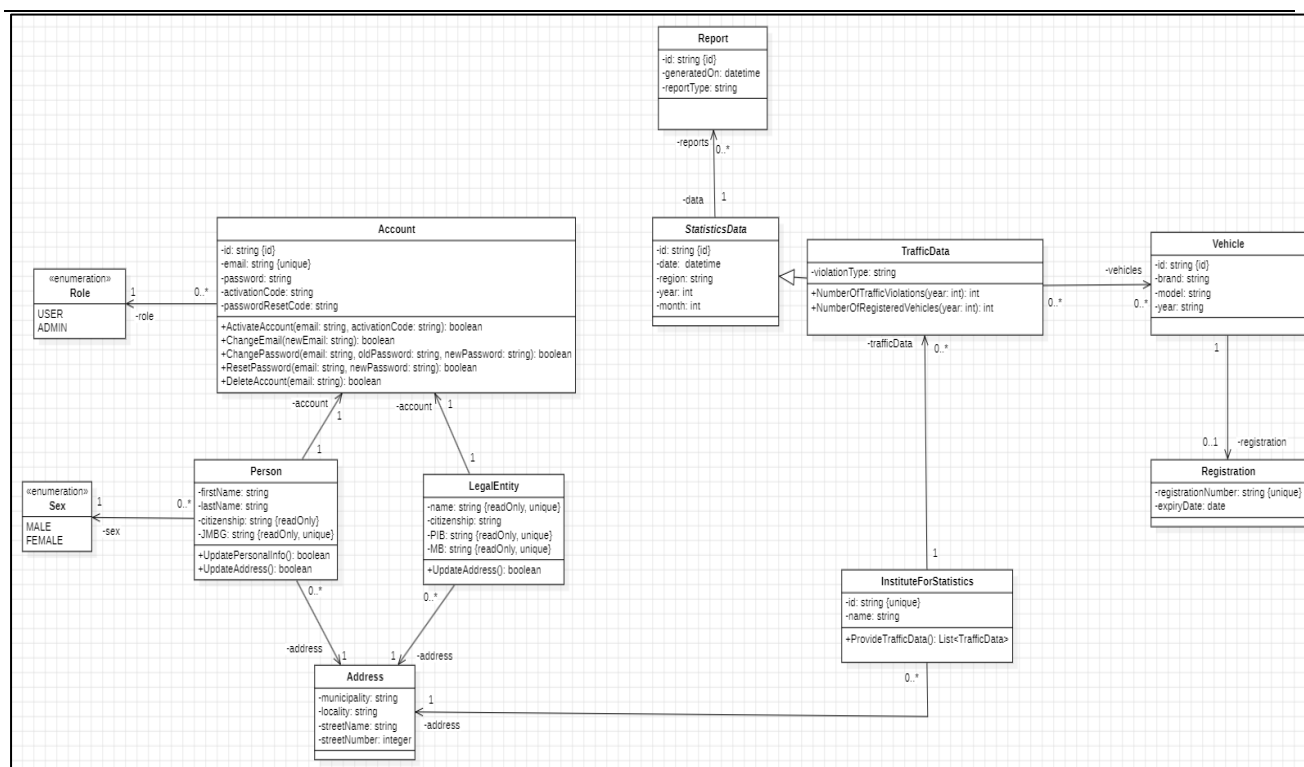
4.1 Arhitektura sistema

Arhitektura sistema je organizovana tako da omogući efikasnu i pouzdanu obradu podataka, kao i jednostavnu integraciju sa postojećim informacionim sistemima. Sistem se sastoji od nekoliko ključnih komponenti:

1. **Korisnički interfejs:** Ova komponenta omogućava korisnicima da pristupe sistemu, izvrše pretrage i pregledaju rezultate. Interfejs je dizajniran tako da bude razumljiv i lako upotrebljiv.
2. **Servisni sloj:** Ova komponenta obrađuje poslovnu logiku, izvršava zahteve korisnika i komunicira sa bazom podataka. Servisni sloj je odgovoran za validaciju podataka, autentifikaciju korisnika i upravljanje sesijama.
3. **Baza podataka:** Skladišti sve potrebne podatke o korisnicima, vozilima, saobraćajnim prekršajima i statistikama. Baza podataka je optimizovana za brzo pretraživanje velikih količina podataka.

4.2 Model podataka

Objektni model podataka je organizovan putem nekoliko ključnih klasa koje predstavljaju entitete u sistemu. UML dijagram klasa (Slika 2) prikazuje glavne klase i njihove odnose.



Slika 2 – Dijagram klasa za sistem zavoda za statistiku

Na slici 2 je pomoću UML dijagrama klasa predstavljen objektni model sistema.

U sistemu su definisani različiti entiteti koji omogućavaju funkcionalnost pretrage sistema avoda za statistiku. Korisnički nalozi su predstavljeni klasom “Account”, koja sadrži informacije kao što su email, lozinka i status aktivacije naloga, te metode za aktivaciju naloga, promenu email adrese i lozinke, resetovanje lozinke i brisanje naloga. Fizička i pravna lica su modelovana klasama “Person” i “LegalEntity”. “Person” uključuje lične podatke kao što su ime, prezime, državljanstvo i jedinstveni matični broj građana (JMBG), dok “LegalEntity” obuhvata podatke o pravnim licima, kao što su naziv, državljanstvo, PIB i matični broj (MB). Adrese povezane sa fizičkim i pravnim licima modelovane su klasom “Address”, koja uključuje podatke kao što su opština, mesto, naziv ulice i broj ulice. Vozila registrovana u sistemu predstavljena su klasom “Vehicle”, koja sadrži informacije o marki, modelu i godini proizvodnje vozila. Podaci o registraciji vozila, uključujući jedinstveni broj registracije i datum isteka registracije, nalaze se u klasi “Registration”. Podaci o saobraćajnim prekršajima beleže se u klasi “TrafficData”, koja ima metode za prebrojavanje saobraćajnih prekršaja i registrovanih vozila po godinama. Statistički podaci, uključujući datum, region, godinu i mesec, nalaze se u klasi “StatisticsData”, koja je povezana sa klasom “Report” za generisanje izveštaja. Ključna komponenta sistema je “InstituteForStatistics”, koja pruža podatke o saobraćajnim prekršajima i registrovanim vozilima i komunicira sa klasom “TrafficData” kako bi pružila odgovarajuće informacije.

5. IMPLEMENTACIJA

Ovo poglavlje prikazuje način na koji su implementirane neke od funkcija sistema zavoda za statistiku. Objasnjena je implementacija ključnih delova sistema, uključujući funkcije za generisanje izveštaja o saobraćajnim prekršajima i broju registrovanih vozila.

5.1 Implementacija funkcije za izveštaj o saobraćajnim prekršajima

U ovom delu je predstavljena implementacija server-side funkcije za generisanje izveštaja o saobraćajnim prekršajima, Implementacija je prikazana na listingu 1.


```

func (sh *StatisticsHandler) GetTrafficViolationsReport(rw http.ResponseWriter, r *http.Request) {
    vars := mux.Vars(r)
    yearStr := vars["year"]
    year, err := strconv.Atoi(yearStr)
    if err != nil {
        http.Error(rw, "Invalid year", http.StatusBadRequest)
        return
    }

    token := sh.extractTokenFromHeader(r)
    violations, err := sh.police.GetTrafficViolations(r.Context(), token)
    if err != nil {
        sh.logger.Println("Failed to retrieve traffic violations:", err)
        http.Error(rw, "Failed to retrieve traffic violations", http.StatusInternalServerError)
        return
    }

    report := make(map[string]int)
    totalViolations := 0

    for _, violation := range violations {
        if violation.Time.Year() == year {
            report[violation.Reason]++
            totalViolations++
        }
    }

    report["Total Violations"] = totalViolations

    rw.Header().Set("Content-Type", "application/json")
    rw.WriteHeader(http.StatusOK)
    if err := json.NewEncoder(rw).Encode(report); err != nil {
        sh.logger.Println("Failed to encode traffic violations report:", err)
        http.Error(rw, "Failed to encode traffic violations report", http.StatusInternalServerError)
    }
}

```

Listing 1 – Generisanje izveštaja o saobraćajnim prekršajima

U ovoj funkciji, na osnovu prosledene godine, funkcija prvo validira unos i zatim pribavlja podatke o saobraćajnim prekršajima komunicirajući sa servisom za saobraćajnu policiju i pozivajući funkciju “`police.GetTrafficViolations`”. Nakon filtriranja prekršaja po godini, funkcija generiše izveštaj koji se šalje kao JSON odgovor.

5.2 Implementacija funkcije za broj registrovanih vozila po godinama

Ovaj deo prikazuje implementaciju server-side funkcije koja vraća broj registrovanih vozila za određenu godinu. Implementacija je prikazana na listingu 2.

```

func (sh *StatisticsHandler) GetRegisteredVehiclesByYear(rw http.ResponseWriter, r *http.Request) {
    vars := mux.Vars(r)
    yearStr := vars["year"]

    year, err := strconv.Atoi(yearStr)
    if err != nil {
        http.Error(rw, "Invalid year", http.StatusBadRequest)
        return
    }

    ctx := r.Context()
    token := sh.extractTokenFromHeader(r)

    vehicles, err := sh.mup.GetAllRegisteredVehicles(ctx, token)
    if err != nil {
        sh.logger.Println("Failed to retrieve registered vehicles:", err)
        http.Error(rw, "Failed to retrieve registered vehicles", http.StatusInternalServerError)
        return
    }

    count := 0
    for _, vehicle := range vehicles {
        if vehicle.Year == year {
            count++
        }
    }

    rw.Header().Set(ContentType, ApplicationJson)
    rw.WriteHeader(http.StatusOK)
    if err := json.NewEncoder(rw).Encode(map[string]int{"count": count}); err != nil {
        sh.logger.Println("Failed to encode registered vehicles count:", err)
        http.Error(rw, "Failed to encode registered vehicles count", http.StatusInternalServerError)
    }
}

```

Listing 2 – Broj registrovanih vozila po godinama

U ovoj funkciji, nakon validacije godine, funkcija dobavlja sva registrovana komunicirajući sa servisom za mup i pozivajući funkciju “mup.GetAllRegisteredVehicles”. Filtriranjem vozila po godini, funkcija izračunava i vraća broj vozila za zadatu godinu kao JSON odgovor.

Objašnjene funkcije demonstriraju kako su implementirani neki od ključnih delova sistema zavoda za statistiku. Ove funkcije koriste spoljašnje servise za pribavljanje podataka, vrše potrebne validacije i filtriranja, te vraćaju odgovore u formatu pogodnom za dalju obradu ili prikaz na klijentskoj strani.

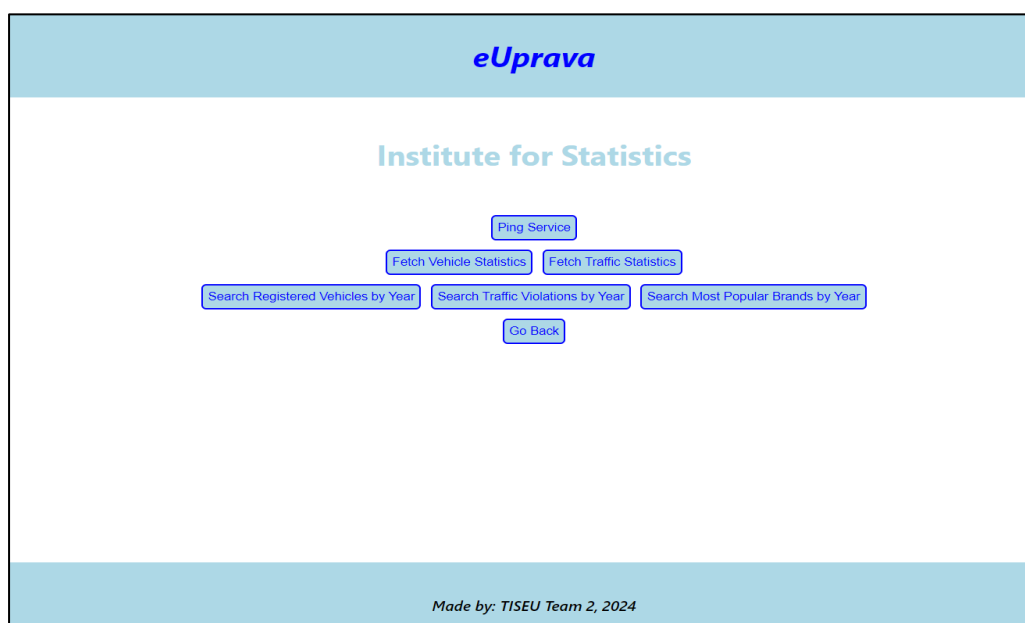
6. DEMONSTRACIJA

Ovo poglavlje prikazuje način korišćenja aplikacije zavoda za statistiku za generisanje i pregled statističkih izveštaja. Opisani su tipični koraci koje korisnik može preduzeti kako bi dobio relevantne statističke podatke iz baze.

6.1 Scenario korišćenja

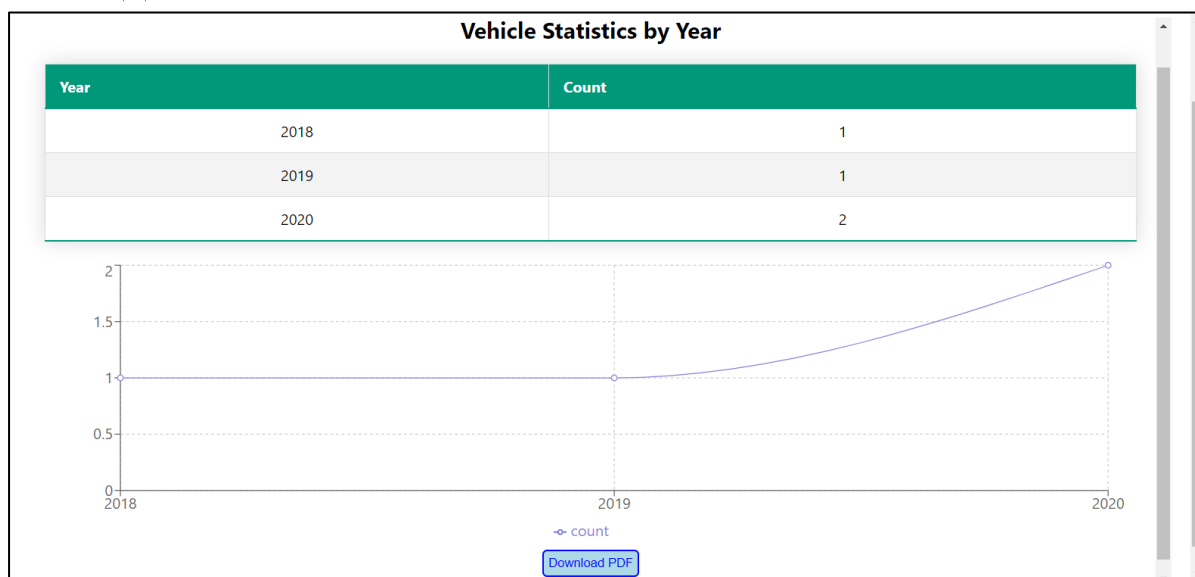
Scenario 1: Generisanje izveštaja o registrovanim vozilima

1. Korisnik pristupa sistemu zavoda za statistiku putem web pregledača.
2. Na početnoj stranici, korisniku se prikazuju opcije za pretragu statističkih podataka (Slika 3).



Slika 3 – Prikaz početne stranice u sistemu zavoda za statistiku

- Klikom na dugme “Fetch Vehicle Statistics” korisniku se otvara modal u kojem korisnik može da vidi detaljan i jasan prikaz koliko je vozila registrovano za određenu godinu (Slika 4).



Slika 4 – Prikaz modala za dobavljanje statistike vozila

- Klikom na dugme “Download PDF” korisnik ima mogućnost preuzimanja izveštaja u PDF formatu.

Scenario 2: Generisanje izveštaja o saobraćajnim prekršajima po godini

- Korisnik pristupa sistemu zavoda za statistiku putem web pregledača.
- Na početnoj stranici, korisniku se prikazuju opcije za pretragu statističkih podataka.
- Klikom na dugme “Search Traffic Violations by Year” korisniku se otvara modal u kojem treba da unese godinu po kojoj želi da pretraži saobraćajne prekršaje (Slika 5).

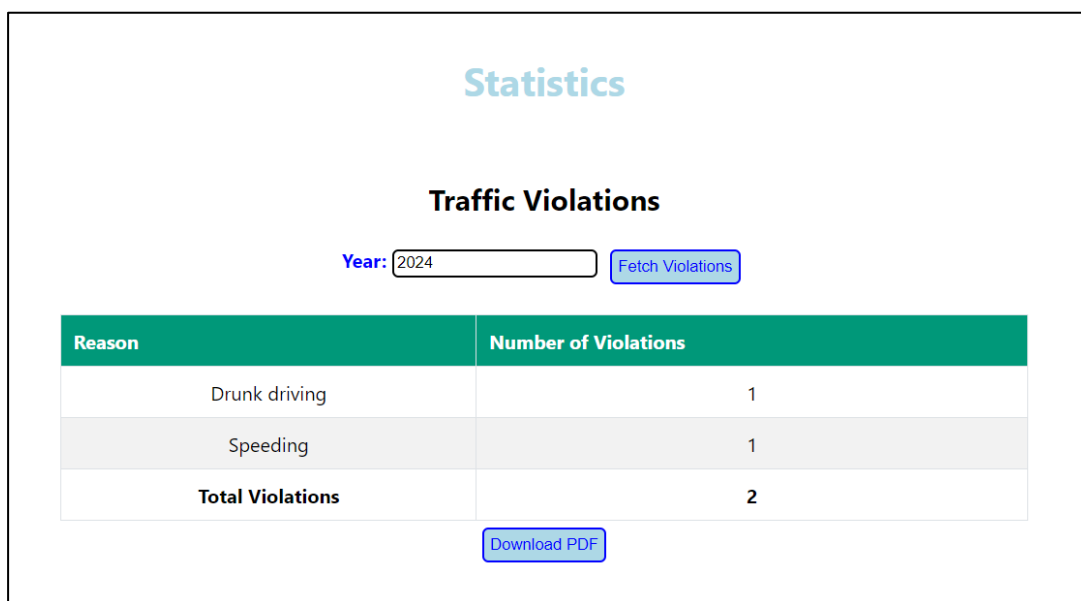
Statistics

Traffic Violations

Year:

Slika 5 – Prikaz modala za pretragu saobraćajnih prekršaja

- Klikom na dugme “Fetch Violations” korisniku se prikazuje statistički izveštaj o saobraćajnim prekršajima za traženu godinu (Slika 6).



Slika 6 – Prikaz izveštaja o saobraćajnim prekršajima

5. Klikom na dugme “Download PDF” korisnik ima mogućnost preuzimanja izveštaja u PDF formatu.

7. ZAKLJUČAK

U ovom radu je predstavljeno softversko rešenje za generisanje i pregled statističkih izveštaja u zavodu za statistiku. Prikazano rešenje omogućava korisnicima da putem web aplikacije lako pristupe statističkim podacima o registrovanim vozilima i saobraćajnim prekršajima, čime se značajno olakšava analiza i donošenje odluka na osnovu relevantnih podataka. Sistem obezbeđuje pregledne i detaljne izveštaje koji su ključni za rad saobraćajne policije, MUP-a, sudova i zavoda za statistiku.

7.1 Analiza postignutih rezultata

Prikazano softversko rešenje uspešno rešava problem pristupa i analize statističkih podataka o saobraćaju. Sistem omogućava korisnicima da brzo i efikasno generišu izveštaje o registrovanim vozilima i saobraćajnim prekršajima, što predstavlja značajnu prednost u odnosu na ručno pretraživanje i analizu podataka. Dobra strana rešenja je jednostavan korisnički interfejs koji omogućava lako korišćenje aplikacije čak i za korisnike sa osnovnim tehničkim znanjima. Automatizacija procesa generisanja izveštaja smanjuje mogućnost grešaka i ubrzava dobijanje rezultata.

Međutim, prikazano rešenje ima i određene nedostatke. Na primer, sistem trenutno ne podržava napredne analitičke funkcije kao što je prediktivna analiza. Takođe, mogućnosti vizualizacije podataka su ograničene i mogu se dodatno unaprediti. Rešenje ovih problema može se postići integracijom sa naprednim analitičkim alatima i razvojem dodatnih funkcionalnosti za vizualizaciju podataka.

7.2 Poređenje sa srodnim rešenjima

U poređenju sa srodnim rešenjima, prikazani sistem pruža jednostavnost korišćenja i osnovne funkcionalnosti potrebne za rad zavoda za statistiku. Većina komercijalnih rešenja nudi širi spektar funkcionalnosti i bolje mogućnosti za analizu podataka, ali su često skuplja i kompleksnija

za implementaciju. Prikazano rešenje je, s druge strane, prilagođeno specifičnim potrebama zavoda za statistiku i omogućava brzu implementaciju i jednostavno održavanje.

7.3 Predlozi za buduća rešenja

Iako je prikazano rešenje odgovorilo na osnovne zahteve problema, postoje brojni pravci za njegovo unapređenje. Jedna od mogućnosti je razvoj mobilne aplikacije koja bi omogućila pristup sistemu putem mobilnih uređaja, čime bi se povećala dostupnost sistema korisnicima. Takođe, integracija sa naprednim analitičkim alatima omogućila bi dublju analizu podataka i prediktivnu analitiku.

8. LITERATURA

- [1] Federal Statistical Office of Germany. 2024. Preuzeto 29.06.2024. sa: https://www.destatis.de/EN/Home/_node.html
- [2] Statistics Canada. Preuzeto 28.06.2024. sa: <https://www.statcan.gc.ca/eng/start>
- [3] Republički zavod za statistiku. Preuzeto 28.06.2024. sa: <https://www.stat.gov.rs/>
- [4] Golang. 2024. Preuzeto 29.06.2024. sa: <https://www.geeksforgeeks.org/golang/>
- [5] React Tutorial. 2024. Preuzeto 29.06.2024. sa: <https://www.geeksforgeeks.org/react-tutorial/>
- [6] MongoDB. 2024. Preuzeto 29.06.2024. sa: <https://docs.digitalocean.com/products/databases/mongodb/>
- [7] Docker overview. 2024. Preuzeto 29.06.2024. sa: <https://docs.docker.com/guides/docker-overview/>