

Univerzitet u Novom Sadu,  
Fakultet Tehničkih Nauka

# **Saobraćajna Policija kao Informacioni Sistem u eUpravi**

## **SEMINARSKI RAD**

Tehnologije i sistemi eUprave

Student,  
Stevan Stankovic SR 4/2021

Novi Sad, 2024

# 1.Sažetak

Saobraćajna policija igra ključnu ulogu u održavanju bezbednosti na putevima i regulisanju saobraćaja.

Ovaj seminarski rad opisuje informacioni sistem za saobraćajnu policiju koji bi značajno olakšao svakodnevne kontrole i provere koje sprovode zaposleni u saobraćajnoj policiji. Za implementaciju korišćena je klijent-server arhitektura, za server smo implementirali mikroservisnu aplikaciju, za implementaciju našeg servisa smo koristili programski jezik Go[1], klijent aplikacija je implementirana pomoću React [2] biblioteke. Podaci su skladišteni u MongoDB [3] koja je smeštena u integrisano okruženje zajedno sa svakim servisom, a to je implementirano uz pomoć Docker [4] platforme.

## 2.Ključne reči

- saobraćajna policija
- veb aplikacija
- mikroservisi
- eUprava
- saobraćajni prekršaji
- informacioni sistem
- bezbednost na putevima
- Automatizacija

### **3.Uvod**

Tradicionalno, rad saobraćajne policije prolazio je kroz opsežnu papirnu dokumentaciju, što je često dovodilo do grešaka, sporih procesa i otežane administracije.

Predloženi sistem omogućio bi automatsku proveru ključnih parametara tokom redovne kontrole saobraćaja, kao što su alkohol u krvi vozača, stanje guma na vozilu, vađenje vozačke dozvole, vađenje registracije vozila, i provera zabrane vožnje za određenog vozača. Na osnovu utvrđenih vrednosti i osobina vozača, sistem bi automatski odredio ispravnost ili neispravnost, kreirao odgovarajuću kaznu i odmah prosledio informacije sudskom podsystemu.

Potrebne podatke za provere sistem bi automatski povlačio iz relevantnih baza podataka Ministarstva unutrašnjih poslova (MUP-a), čime bi se smanjila mogućnost grešaka i ubrzala procedura kontrole. Ovakav pristup ne samo da bi unapredio efikasnost rada saobraćajne policije, već bi i povećao transparentnost i pouzdanost u sprovođenju saobraćajnih propisa.

## **4.Srodna istraživanja**

### **1.eCitation (SAD)**

- Elektronski sistem za izdavanje kazni koji omogućava policijskim službenicima da digitalno evidentiraju saobraćajne prekršaje direktno sa terena. Sistem automatski proverava podatke o vozaču, vozilu i statusu vozačke dozvole, te generiše kazne koje se odmah šalju nadležnim sudskim organima.

### **2. ANPR (Automatic Number Plate Recognition) Sistemi (UK, EU):**

- Sistemi za automatsko prepoznavanje registarskih tablica koji se koriste za nadzor i praćenje saobraćaja. Ovi sistemi omogućavaju brzo prepoznavanje vozila sa isteklom registracijom, ukradenih vozila ili onih koja su povezana sa kršenjem saobraćajnih propisa.

### **3. eChallan (Indija)**

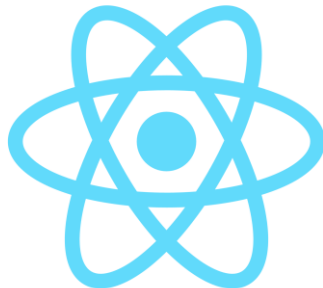
- Digitalni sistem za izdavanje kazni koji omogućava policijskim službenicima da elektronski evidentiraju saobraćajne prekršaje i šalju kazne direktno vozačima putem SMS-a ili emaila. Sistem je povezan sa centralnim bazama podataka za proveru informacija o vozačima i vozilima.

## 5. Korišćene tehnologije

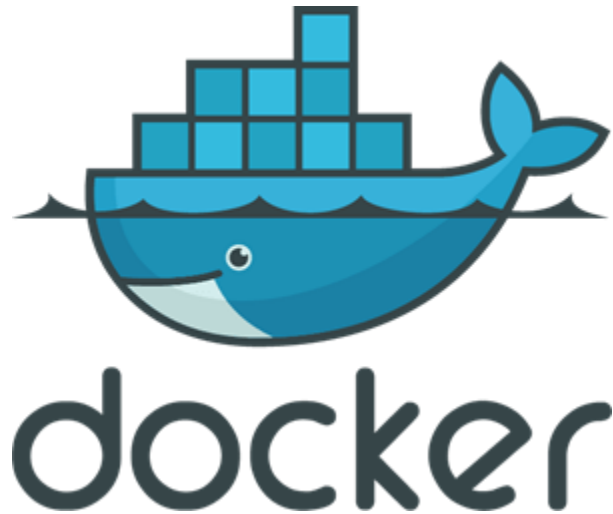
- **Golang** programski jezik razvijen od strane Google-a. Osobine kao jednostavnost i efikasnost ga čine idealnim za izgradnju skalabilnih i visoko performantnih aplikacija. Jezik je kompajliran, sa statičkim tipovima podataka, strukturnim tipovima i sadrži automatsko upravljanje memorijom. Ovaj jezik je prvenstveno bio namenjen sistemskom programiranju.  
Jezik je široko prihvaćen u industriji zbog svoje jednostavnosti, efikasnosti i podrške za konkurentno programiranje, ove osobine ga čine odličnim za implementaciju serverskih aplikacija, mrežnih servisa i mikroservisa.



- **React** Javascript biblioteka razvijena od strane Facebook-a koja se koristi za izgradnju korisničkog interfejsa. Poznat je 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.



- **Docker** otvorena platforma za razvoj, isporuku, testiranje i pokretanje aplikacija. Osnovne stvari kojima Docker rukuje su kontejneri, odnosno jedinice koje sadrže sve što je softveru potrebno za rad, kao što su biblioteke, runtime okruženje i sistemski alati. Docker pakuje i pokreće aplikacije u labavo izolovanom okruženju koje smo već spomenuli a nazivaju se kontejneri, oni se mogu deliti dok radite i budete sigurni da svi sa kojima delite dobijaju isti kontejner koji funkcioniše na isti način.



- **MongoDB** je NoSQL baza podataka koja koristi dokumentno orijentisan model za skladištenje podataka. Umesto tradicionalnih tabela koje se koriste u relacijskim bazama podataka, MongoDB koristi kolekcije dokumenata u BSON formatu (Binary JSON). Neke od ključnih karakteristika su: fleksibilna struktura podataka, skalabilnost, visoke performanse, agregacija i replikacija.

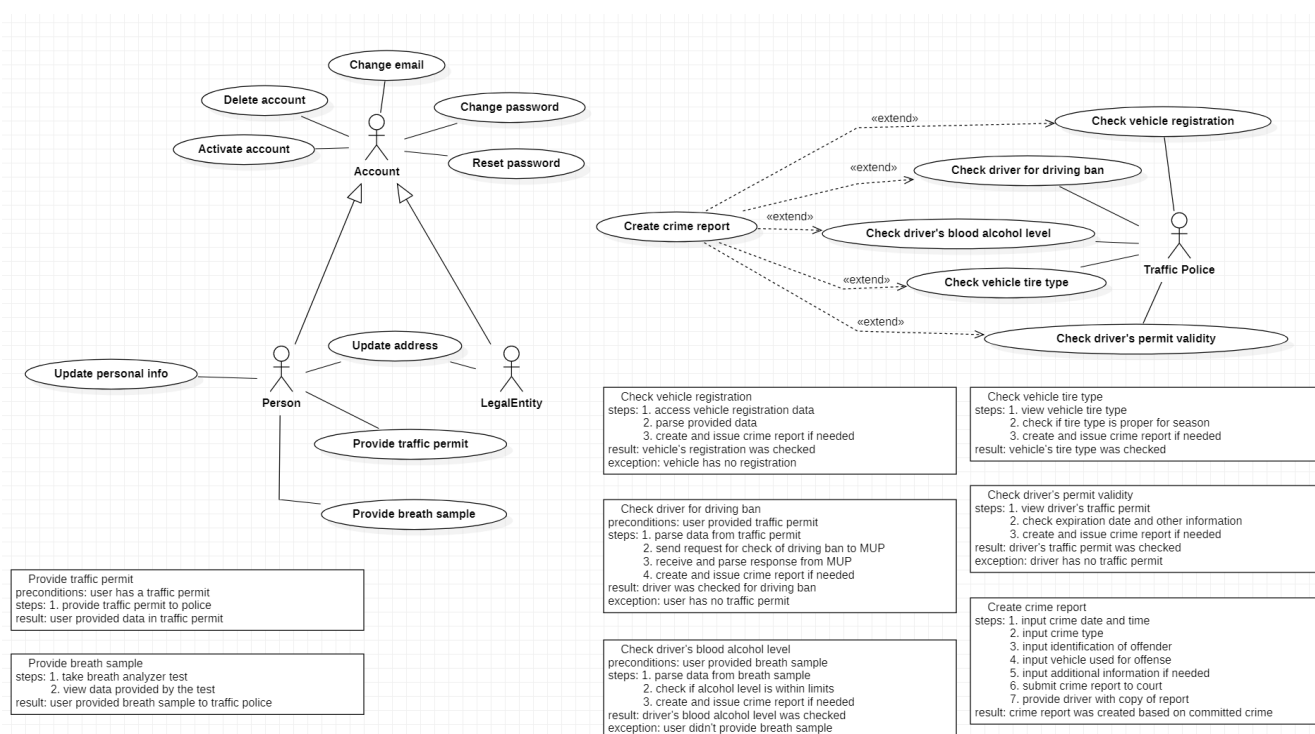


## 6.Specifikacija zahteva

U nastavku će biti objašnjeni funkcionalni i nefunkcionalni zahtevi softverskog rešenja predstavljenog u ovom radu.

### 6.1 Funkcionalni zahtevi

Funkcionalni zahtevi ovog softverskog rešenja su predstavljeni UML dijagramom slučajeva korišćenja, kao na prikazanoj slici ispod (Slika 1).



Slika 1 – Dijagram slučajeva korišćenja za sistem saobraćajne policije

#### 6.1.1 Provera alkohola u krvi vozaču

- Preduslov: Postoji korisnik sa unetim jmbg-om u sistem
- Koraci:
  1. Unosi se podataka o količini alkohola u krvi očitanoj kod vozača
  2. Provera da li alkohol u krvi prelazi granicu dozvoljenosti
  3. Kreiranje kazne ako je prekršen zakon
- Rezultat: Vozač prekršio zakon, kazna se šalje sudu
- Izuzetak: Vozač nije napravio prekršaj, kazna se ne kreira



### **6.1.2 Provera guma na vozilu**

- Preduslov: Postoji korisnik sa unetim jmbg-om u sistemu, gume na vozilu mogu biti samo letnjeg i zimskog tipa.
- Koraci:
  - 1.Unosi se tip guma vozila
  2. Provera da li je dozvoljen taj tip guma u trenutnom periodu godine
  3. Kreiranje kazne ako je prekršen zakon
- Rezultat: Vozač prekršio zakon, kazna se šalje sudu
- Izuzetak: Vozač nije napravio prekršaj, kazna se ne kreira

### **6.1.3 Provera važenja vozačke dozvole**

- Preduslov: Postoji korisnik sa unetim jmbg-om u sistemu, da poseduje vozačku dozvolu
- Koraci:
  - 1.Unosi se jmbg vozača
  - 2.Provera se vrši tako što se dobavlja iz MUP servisa vozačka dozvola na osnovu jmbg vozača
  - 3.Proverava se datum vozačke dozvole na osnovu trenutnog datuma
  - 4.Kreira se kazna ako je prekršen zakon
- Rezultat: Vozač je prekršio zakon, kazna se šalje sudu
- Izuzetak: Vozač nije napravio prekršaj, kazna se ne kreira

### **6.1.4 Provera registracije vozila**

- Preduslov: Postoji korisnik sa unetim jmbg-om i da postoji vozilo sa prosledjenim brojem tablica u sistemu
- Koraci:
  - 1.Unosi se jmbg vozača i broj tablica vozila
  - 2.Provera se vrši tako što se dobavlja iz MUP servisa registracija vozila na osnovu broja tablica
  - 3.Proverava se datum isteka registracije sa trenutnim datumom
  - 4.Kreira se kazna ako je istekla registracija
- Rezultat: Vozilu istekla registracija, kreirana kazna se šalje sudu
- Izuzetak: Vozilu važi registracija, kazna se ne kreira

### **6.1.5 Prover da li je vozač pod zabranom vožnje**

- Preduslov: Postoji korisnik sa unetim jmbg-om
- Koraci:

- 1.Unosi se jmbg vozača
  - 2.Provera se vrši tako što se dobavlja iz MUP servisa, ako ima zabrane voznje, vraća se najnoviji zabrana na osnovu datuma
  - 3.Ako je datum isteka zabrane u buducnosti u odnosu na trenutni datum, vozac je pod zabranom
  - 4.Kreira se kazna i prosleđuje se sudu
- Rezultat: Vozač upravljao vozilom pod zabranom vožnje, kreirana kazna se šalje sudu
  - Izuzetak: Korisnik regularno upravljao vozilom, ne kreira se kazna

## **6.2 Nefunkcionalni zahtevi**

Predstavljam nefunkcionalne zahteve, o kojima softversko rešenje treba da vodi računa. Najčešći nefunkcionalni zahtevi su efikasnost, brzina, pouzdanost, skalabilnost i tako dalje. U nastavku su opisani neki nefunkcionalni zahtevi vezani za datu aplikaciju eUprave.Sistem mora biti otporan na greške, zbog korisničkog iskustva tokom korišćenja softvera.

### **6.2.1 Dockerizacija**

Svaki mikroservis ima svoj kontejner, koristi se Docker Compose alat. Docker olakšava rad u različitim okruženjima bez dodatnih konfiguracija. Bitna osobina Docker-a je ta da osigurava da aplikacija može raditi u različitim okruženjima bez dodatnih konfiguracija.

### **6.2.2 Sigurnost**

Jedan od ključnih aspekt svakog sistema, pa tako i ovog. Samo ovlašćeni korisnici treba da imaju pristup određenim podacima i funkcionalnostima sistema, sigurnost se uveliko postiže autorizacijom i autentifikacijom. Single Sign-On (SSO) tehnologija je korišćena u implementiranim podsistemima, ona omogućava sigurnu autentifikaciju korisnika i osigurava da samo ovlašćeni korisnici imaju pristup određenim funkcionalnostima. Dok je komunikacija izmedju servisa dodatno zaštićena tako što je šifrovana, da bi se zaštitili podaci koji se koriste u toj komunikaciji.

### **6.2.3 Upotrebljivost**

Korisnički interfejs mora biti jednostavan i lak za upotrebu da bi korisnici mogli efikasno i jednostavno da obavljaju potrebne operacije.

### **6.2.4 Pouzdanost**

Jako bitna osobina sistema. Sistem treba da bude visoko dostupan, sa minimalnim vremenom neaktivnosti, kako bi korisnici mogli neprekidno da koriste njegove usluge.

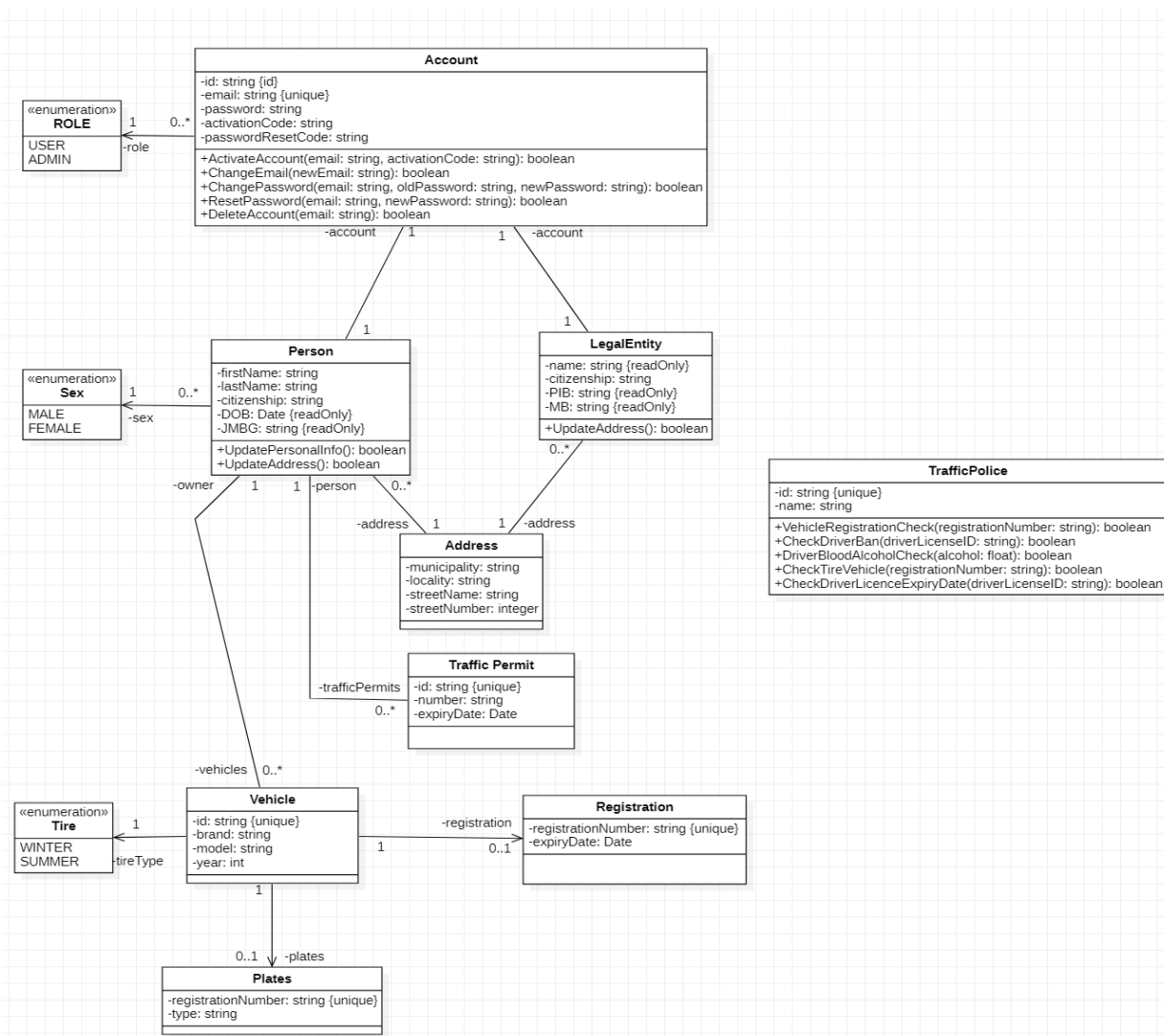
#### **6.2.5 Otpornost na otkaze pojedinačnih sistema**

Prilikom otkaza jednog od sistema, ostali ili drugi sistemi bi trebali da nastave da funkcionišu. Lakša identifikacija problema ukoliko do njega dođe, jer znaćemo u kojem sistemu se tačno dešava problem.

## **6.Specifikacija dizajna**

Naša aplikacija predstavlja mikroservisnu arhitekturu. Svaki podsistem eUprave predstavlja zaseban servis koji ima komunikaciju sa drugim servisima. Naš tim je radio na implementaciji četiri servisa koji predstavljaju MUP, Sud, Saobraćajna policija, Zavod za Statistiku, kao i na SSO servis koji predstavlja autorizacioni servis.. Naš rad je fokusiran na servis saobraćajne policije, prilikom dobavljanja potrebnih podataka naš servis komunicira sa servisom MUP-a, a prosleđuje podatke servisu suda tj.kreirane kazne prilikom provere vozača ili vozila ako su utvrđeni prekršaji kod nečeg navedenog.

Na slici 2. su preko klasnog dijagrama predstavljene komponente koje spadaju u saobraćajne policije podsistem, kao i komponente iz autentifikacionog sistema.



Slika 2. - Klasni dijagram sudskog podsistema

Korisnik može biti fizičko ili pravno lice. Tj. u ovom slučaju predstavlja vozača ili saobraćajnog policajca. Svaki korisnik ima nalog, sa kojim pristupa uslugama saobraćajnog podsistema. Svaki vozač može biti podvrgnut proverama stanja guma, važenja vozačke dozvole, zabrane vožnje, važenja registracije vozila i nivoa alkohola u krvi kako bi se osigurala bezbednost u saobraćaju. Saobraćajni policajac ima na uvid sve kreirane kazne nad fizičkim licima, kao i svako fizičko lice što može da pregleda sve svoje kazne.

## 7.Implementacija

U nastavku tj.u ovom poglavlju će biti prikazan način implementacije funkcionalnosti sistema saobraćajne policije.

```
func (ph *PoliceHandler) CheckAlcoholLevel(w http.ResponseWriter, r *http.Request) {
    var alcoholLevel data.AlcoholRequest

    err := json.NewDecoder(r.Body).Decode(&alcoholLevel)
    if err != nil {
        http.Error(w, "Failed to decode request body", http.StatusBadRequest)
        log.Printf("Failed to decode request body: %v\n", err)
        return
    }

    violation := data.TrafficViolation{
        ID:         primitive.NewObjectID(),
        Time:       time.Now(),
        ViolatorJMBG: alcoholLevel.JMBG,
        Location:   alcoholLevel.Location,
    }

    response := data.Response{}

    if alcoholLevel.AlcoholLevel > 0.2 {
        violation.Reason = fmt.Sprintf("drunk driving: %.2f \n", alcoholLevel.AlcoholLevel)
        violation.Description = "Driver was caught operating a vehicle with a blood alcohol level above the legal limit. \n"
    } else {
        response.Message = "Driver was caught operating a vehicle with a blood alcohol level within the legal limit."
        w.Header().Set("Content-Type", "application/json")
        w.WriteHeader(http.StatusOK)
        json.NewEncoder(w).Encode(response)
        return
    }

    token := ph.extractTokenFromHeader(r)
    _, err = ph.sso.GetPersonByJMBG(r.Context(), alcoholLevel.JMBG, token)
    if err != nil {
        http.Error(w, "Error with services communication", http.StatusBadRequest)
        log.Printf("Error while communicating with SSO service: %s", err.Error())
        return
    }

    err = ph.repo.CreateTrafficViolation(r.Context(), &violation)
    if err != nil {
        http.Error(w, "Failed to create traffic violation", http.StatusBadRequest)
        log.Printf("Failed to create traffic violation: %v\n", err)
        return
    }

    err = ph.court.CreateCrimeReport(r.Context(), violation, token)
    if err != nil {
        http.Error(w, "Failed to send crime report", http.StatusBadRequest)
        log.Printf("Failed to send crime report: %v\n", err)
        return
    }

    response.Message = "Driver has more alcohol in his blood than is allowed."
    response.Data = violation

    w.Header().Set("Content-Type", "application/json")
    w.WriteHeader(http.StatusCreated)
    json.NewEncoder(w).Encode(response)
}
```

Slika 3. - Funkcija za proveru alkohola u krvi kod vozača

Funkcija prima JMBG, kolicinu alkohola u krvi i lokaciju na kojoj je provera izvršena, ti podaci se prosleđuju kroz odgovarajuću strukturu. Vrš se provera ako je prosleđena vrednost alkohola u krvi veća od 0.2, kazna se kreira, vreme kreiranja kazne se setuje na

trenutno, razlog kazne kao i detaljniji opis se upisuju, takođe i lokacija na kojoj je izvršena provera. Kazna ili možemo reći i izveštaj se čuva u bazi podataka u servisu saobraćajne policije, i onda se prosleđuje sudskom servisu na uvid. U slučaju da je vozač upravljao vozilom sa alkoholom u krvi manje od nedozvoljenog, korisniku će biti prikazano obaveštenje sa tom porukom.

```
func (ph *PoliceHandler) CheckDriverBan(w http.ResponseWriter, r *http.Request) {
    var driverBan data.DriverBanAndPermitRequest
    err := json.NewDecoder(r.Body).Decode(&driverBan)
    if err != nil {
        http.Error(w, "Failed to decode request body", http.StatusBadRequest)
        log.Printf("Failed to decode request body: %v\n", err)
        return
    }

    violation := data.TrafficViolation{
        ID:          primitive.NewObjectID(),
        Time:        time.Now(),
        ViolatorJMBG: driverBan.JMBG,
        Location:    driverBan.Location,
    }

    token := ph.extractTokenFromHeader(r)

    jmbgRequest := data.JMBGRequest{
        JMBG: driverBan.JMBG,
    }

    _, err = ph.sso.GetPersonByJMBG(r.Context(), driverBan.JMBG, token)
    if err != nil {
        http.Error(w, "Error with services communication", http.StatusBadRequest)
        log.Printf("Error while communicating with SSO service: %s", err.Error())
        return
    }

    drivingBan, err := ph.mup.CheckDrivingBan(r.Context(), jmbgRequest, token)
    if err != nil {
        http.Error(w, "Failed to check driving ban: "+err.Error(), http.StatusBadRequest)
        log.Printf("Failed to check driving ban: %v\n", err)
        return
    }

    if drivingBan != nil && drivingBan.Duration.After(time.Now()) {
        violation.Reason += "Driving ban is in effect\n"
        violation.Description += "Driver was found to be operating a vehicle under active driving ban. Reason: " + drivingBan.Reason + "\n"
        log.Print("Driving ban is in effect")
    } else {
        response := data.Response{
            Message: "The driver is not under a driving ban.",
        }
        w.Header().Set("Content-Type", "application/json")
        w.WriteHeader(http.StatusOK)
        json.NewEncoder(w).Encode(response)
        return
    }

    err = ph.repo.CreateTrafficViolation(r.Context(), &violation)
    if err != nil {
        http.Error(w, "Failed to create traffic violation", http.StatusInternalServerError)
        log.Printf("Failed to create traffic violation: %v\n", err)
        return
    }

    err = ph.court.CreateCrimeReport(r.Context(), violation, token)
    if err != nil {
        http.Error(w, "Failed to send crime report", http.StatusInternalServerError)
        log.Printf("Failed to send crime report: %v\n", err)
        return
    }

    response := data.Response{
        Message: "Traffic violation created successfully",
        Data:    violation,
    }

    w.Header().Set("Content-Type", "application/json")
    w.WriteHeader(http.StatusCreated)
    json.NewEncoder(w).Encode(response)
}
```

Slika 4. - Funkcija za proveru vozača od zabrane upravljanja vozilom

Funkcija CheckDriverBan proverava da li vozač ima zabranu upravljanja vozilom i u skladu sa tim kreira izveštaj o prekršaju. Proverava se da li vozač ima aktivnu zabranu vožnje. Ako

postoji zabrana koja je na snazi, dodaje se razlog i opis prekršaja u objekat TrafficViolation. Ako zabrane nema, vraća se poruka "The driver is not under a driving ban.". Ako postoji prekršaj, kreira se izveštaj o saobraćajnom prekršaju i šalje se nadležnom sudu.

```
func (ph *PoliceHandler) CheckDriverPermitValidity(w http.ResponseWriter, r *http.Request) {
    var driverBan data.DriverBanAndPermitRequest
    err := json.NewDecoder(r.Body).Decode(&driverBan)
    if err != nil {
        http.Error(w, "Failed to decode request body", http.StatusBadRequest)
        log.Printf("Failed to decode request body: %v\n", err)
        return
    }

    violation := data.TrafficViolation{
        ID:        primitive.NewObjectID(),
        Time:      time.Now(),
        ViolatorJMBG: driverBan.JMBG,
        Location:  driverBan.Location,
    }

    response := data.Response{}

    token := ph.extractTokenFromHeader(r)

    jmbgRequest := data.JMBGRequest{
        JMBG: driverBan.JMBG,
    }

    permit, err := ph.mup.GetDrivingPermitByJMBG(r.Context(), jmbgRequest, token)
    if err != nil {
        log.Printf("Failed to check driving permit: %v\n", err)
        http.Error(w, "Failed to check driving permit", http.StatusBadRequest)
        return
    }

    if permit.Number == "" {
        log.Printf("Not found driving permit.")
        http.Error(w, "Not found driving permit.", http.StatusBadRequest)
        return
    }

    if permit.ExpirationDate.Before(time.Now()) {
        violation.Reason += "Driving permit expired \n"
        violation.Description += "Driver was found to have an expired driving permit. \n"
        log.Print("Driving permit is expired")
    } else {
        response.Message = "The driver permit is valid."
        w.Header().Set("Content-Type", "application/json")
        w.WriteHeader(http.StatusOK)
        json.NewEncoder(w).Encode(response)
        return
    }

    err = ph.repo.CreateTrafficViolation(r.Context(), &violation)
    if err != nil {
        http.Error(w, "Failed to create traffic violation", http.StatusInternalServerError)
        log.Printf("Failed to create traffic violation: %v\n", err)
        return
    }

    err = ph.court.CreateCrimeReport(r.Context(), violation, token)
    if err != nil {
        http.Error(w, "Failed to send crime report", http.StatusInternalServerError)
        log.Printf("Failed to send crime report: %v\n", err)
        return
    }

    response.Message = "Driver has an expired driving permit."
    response.Data = violation

    w.Header().Set("Content-Type", "application/json")
    w.WriteHeader(http.StatusCreated)
    json.NewEncoder(w).Encode(response)
}
```

Slika 5. - Funkcija za proveru važenja vozačke dozvole

Koristi se servis za dobijanje informacija o vozačkoj dozvoli na osnovu JMBG-a. Ako dozvola nije pronađena ili je došlo do greške, vraća se odgovarajuća poruka o grešci. Ako je vozačka dozvola istekla, dodaje se razlog i opis prekršaja u objekat TrafficViolation. Ako dozvola nije istekla, vraća se poruka "The driver permit is valid.". Ako postoji prekršaj, kreira se izveštaj o saobraćajnom prekršaju i šalje se nadležnom sudu. Ako dođe do greške u bilo kom koraku, vraća se odgovarajuća poruka o grešci. Ako je sve uspešno obavljeno, vraća se odgovor sa porukom "Driver has an expired driving permit." zajedno sa podacima o prekršaju.



```

func (ph *PoliceHandler) CheckVehicleTire(w http.ResponseWriter, r *http.Request) {
    var tireType data.VehicleTireCheck
    err := json.NewDecoder(r.Body).Decode(&tireType)
    if err != nil {
        http.Error(w, "Failed to decode request body", http.StatusBadRequest)
        log.Printf("Failed to decode request body: %v\n", err)
        return
    }

    response := data.Response{}
    token := ph.extractTokenFromHeader(r)

    now := time.Now()
    year := now.Year()

    startWinterPeriod := time.Date(year, time.November, 1, 0, 0, 0, time.Local)
    endWinterPeriod := time.Date(year+1, time.April, 1, 0, 0, 0, time.Local)
    startSummerPeriod := time.Date(year, time.April, 1, 0, 0, 0, time.Local)
    endSummerPeriod := time.Date(year, time.November, 1, 0, 0, 0, time.Local)

    if now.Month() >= time.November || now.Month() < time.April {
        endSummerPeriod = time.Date(year+1, time.November, 1, 0, 0, 0, time.Local)
    }

    violation := data.TrafficViolation{
        ID:          primitive.NewObjectID(),
        Time:        now,
        ViolatorJMBG: tireType.JMBG,
        Location:    tireType.Location,
    }

    switch tireType.TireType {
    case "WINTER":
        if now.After(startWinterPeriod) && now.Before(endWinterPeriod) {
            response.Message = "No violation for winter tires"
            w.Header().Set("Content-Type", "application/json")
            w.WriteHeader(http.StatusOK)
            json.NewEncoder(w).Encode(response)
            return
        }
        violation.Reason = "Improper tire usage: WINTER tires outside summer period"
        violation.Description = "Driver was caught operating a vehicle with WINTER tires outside the summer period (April 1 to November 1), which is against regulations."
    case "SUMMER":
        if now.After(startSummerPeriod) && now.Before(endSummerPeriod) {
            response.Message = "No violation for summer tires in the summer period"
            w.Header().Set("Content-Type", "application/json")
            w.WriteHeader(http.StatusOK)
            json.NewEncoder(w).Encode(response)
            return
        }
        violation.Reason = "Improper tire usage: SUMMER tires during winter period"
        violation.Description = "Driver was caught operating a vehicle with SUMMER tires during the winter period (November 1 to April 1), which is against regulations."
    }

    default:
        response.Message = "Invalid tire type specified"
        w.Header().Set("Content-Type", "application/json")
        w.WriteHeader(http.StatusBadRequest)
        json.NewEncoder(w).Encode(response)
        return
    }

    _, err = ph.sso.GetPersonByJMBG(r.Context(), tireType.JMBG, token)
    if err != nil {
        http.Error(w, "Error with services communication", http.StatusBadRequest)
        log.Printf("Error while communicating with SSO service: %s", err.Error())
        return
    }

    err = ph.repo.CreateTrafficViolation(r.Context(), &violation)
    if err != nil {
        response.Message = "Failed to create traffic violation"
        w.Header().Set("Content-Type", "application/json")
        w.WriteHeader(http.StatusBadRequest)
        json.NewEncoder(w).Encode(response)
        log.Printf("Failed to create traffic violation: %v\n", err)
        return
    }

    err = ph.court.CreateCrimeReport(r.Context(), violation, token)
    if err != nil {
        response.Message = "Failed to send crime report"
        w.Header().Set("Content-Type", "application/json")
        w.WriteHeader(http.StatusBadRequest)
        json.NewEncoder(w).Encode(response)
        log.Printf("Failed to send crime report: %v\n", err)
        return
    }

    response.Message = "Traffic violation created successfully."
    response.Data = violation
    w.Header().Set("Content-Type", "application/json")
    w.WriteHeader(http.StatusCreated)
    json.NewEncoder(w).Encode(response)
}

```

Slika 6. - Funkcija za proveru guma vozila kojim upravlja vozač

Definišu se periodi za zimske i letnje gume na osnovu trenutne godine.

Proverava se tip guma (WINTER ili SUMMER) i trenutni datum u odnosu na definisane periode:

- Ako su zimske gume korišćene u zimskom periodu, vraća se poruka "No violation for winter tires".
- Ako su letnje gume korišćene u letnjem periodu, vraća se poruka "No violation for summer tires".
- Ako su gume korišćene van odgovarajućih perioda, kreira se prekršaj sa odgovarajućim razlogom i opisom.

Proverava se postojanje vozača u sistemu koristeći JMBG. Ako vozač nije pronađen ili dođe do greške, vraća se odgovarajuća poruka o grešci.

Ako postoji prekršaj, kreira se izveštaj o saobraćajnom prekršaju i šalje se nadležnom sudu. Ako dođe do greške u bilo kom koraku, vraća se odgovarajuća poruka o grešci.

```

func (ph *PoliceHandler) CheckVehicleRegistration(w http.ResponseWriter, r *http.Request) {
    var checkVehicleRegistration data.CheckVehicleRegistration
    err := json.NewDecoder(r.Body).Decode(&checkVehicleRegistration)
    if err != nil {
        http.Error(w, "Failed to decode request body", http.StatusBadRequest)
        log.Printf("Failed to decode request body: %v\n", err)
        return
    }

    violation := data.TrafficViolation{
        ID:          primitive.NewObjectID(),
        Time:        time.Now(),
        ViolatorJMBG: checkVehicleRegistration.JMBG,
        Location:    checkVehicleRegistration.Location,
    }

    token := ph.extractTokenFromHeader(r)

    plates := data.PlateRequest{
        Plate: checkVehicleRegistration.PlatesNumber,
    }

    registration, err := ph.mup.GetRegistrationByPlate(r.Context(), plates, token)
    if err != nil {
        log.Printf("Failed to check registration by plate: %v\n", err)
        http.Error(w, "Failed to check registration by plate", http.StatusBadRequest)
        return
    }

    if registration.RegistrationNumber == "" {
        w.Header().Set("Content-Type", "application/json")
        w.WriteHeader(http.StatusBadRequest)
        fmt.Fprintf(w, "Wrong plates number in body request")
        return
    }

    if registration.ExpirationDate.Before(time.Now()) {
        violation.Reason = "Vehicle registration expired"
        violation.Description = "Driver was found to be operating a vehicle with an expired registration."
        log.Print("Vehicle registration is expired")
    } else {
        w.Header().Set("Content-Type", "application/json")
        w.WriteHeader(http.StatusOK)
        fmt.Fprintf(w, "The vehicle registration has not expired.")
        log.Print("Vehicle registration has not expired")
        return
    }

    err = ph.repo.CreateTrafficViolation(r.Context(), &violation)
    if err != nil {
        http.Error(w, "Failed to create traffic violation", http.StatusBadRequest)
        log.Printf("Failed to create traffic violation: %v\n", err)
        return
    }

    err = ph.court.CreateCrimeReport(r.Context(), violation, token)
    if err != nil {
        http.Error(w, "Failed to send crime report", http.StatusBadRequest)
        log.Printf("Failed to send crime report: %v\n", err)
        return
    }

    w.Header().Set("Content-Type", "application/json")
    w.WriteHeader(http.StatusCreated)
    json.NewEncoder(w).Encode(violation)
}

```

Slika 7. - Funkcija koja proverava validnost registracije vozila kojom upravlja vozač

Kreira se objekat PlateRequest sa brojem tablica i poziva se funkcija GetRegistrationByPlate da proveriti registraciju vozila. Ako dođe do greške, vraća se odgovarajuća poruka o grešci.

Ako broj registracije nije pronađen, vraća se poruka "Wrong plates number in body request". Ako je registracija istekla, kreira se prekršaj sa odgovarajućim razlogom i opisom. Ako registracija nije istekla, vraća se odgovarajuća poruka.

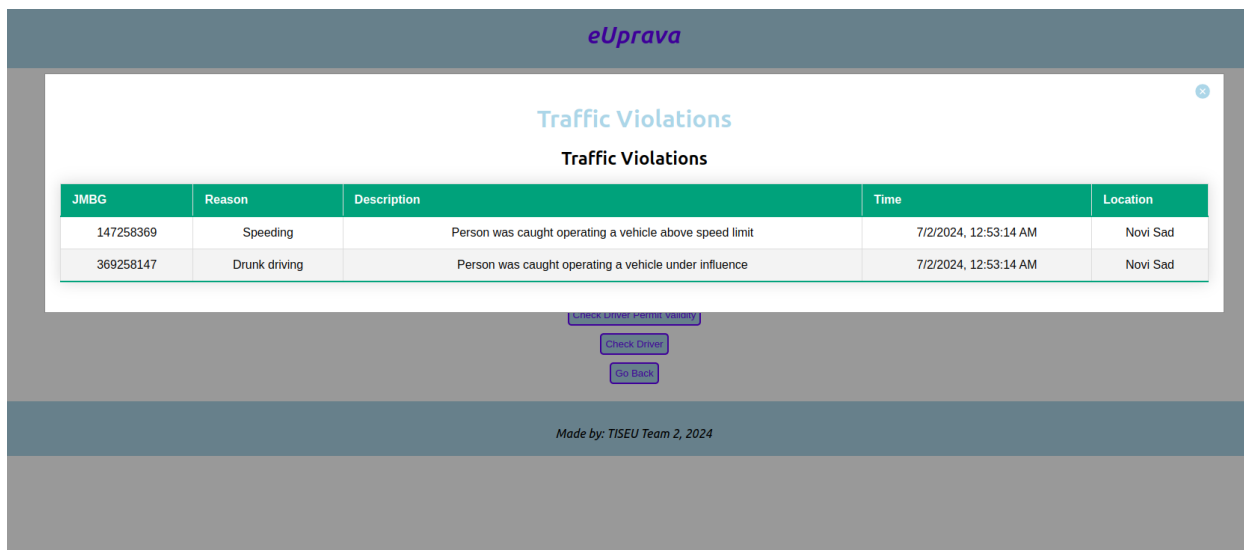
Ako postoji prekršaj, kreira se izveštaj o saobraćajnom prekršaju i šalje se nadležnom

sudu. Ako dođe do greške u bilo kom koraku, vraća se odgovarajuća poruka o grešci.

Takodje, implementirana je funkcija koja vrši sve ove prethodno prikazane provere kroz jedan zahtev.

## 8.Demonstracija

Pregled svih izdatih kazni ili prosleđenih izveštaja ka sudskom servisu



The screenshot shows a web application interface for 'eUprava'. The main content area is titled 'Traffic Violations' and contains a table with the following data:

| JMBG      | Reason        | Description   | Time                  | Location |
|-----------|---------------|---|-----------------------|----------|
| 147258369 | Speeding      | Person was caught operating a vehicle above speed limit | 7/2/2024, 12:53:14 AM | Novi Sad |
| 369258147 | Drunk driving | Person was caught operating a vehicle under influence   | 7/2/2024, 12:53:14 AM | Novi Sad |

Below the table, there are buttons for 'Check Driver Permit Validity', 'Check Driver', and 'Go Back'. The footer text reads 'Made by: TISEU Team 2, 2024'.

Slika 8. - Tabelrani prikaz svih kreiranih kazni od strane saobraćajne policije

Forma za unos podataka tokom provere važenja vozačke dozvole vozača, unosi se JMBG vozača i lokacija na kojoj je vozač zaustavljen od strane saobraćajne patrolle.



The screenshot shows a web application interface for 'eUprava'. The main content area is titled 'Traffic Violations' and contains a form titled 'Check Driver Permit Validity'. The form has two input fields: 'JMBG' and 'Location'. Below the 'Location' field is a button labeled 'Check Driver Permit Validity'. Below the form, there are buttons for 'Check Driver' and 'Go Back'. The footer text reads 'Made by: TISEU Team 2, 2024'.

Slika 9. - Forma za unos podataka tokom provere važenja vozačke dozvole



Forma za unos podataka tokom provere vozača da li upravlja vozilom dok mu je na snazi zabrana upravljanja vozilom, unosi se JMBG vozača i lokacija na kojoj je vozač zaustavljen od strane saobraćajne patrola.

The screenshot shows a web application interface for 'eUprava'. At the top, there is a dark blue header with the text 'eUprava' in white. Below the header, a light blue box contains the title 'Traffic Violations' in blue text. Inside this box, there is a white card titled 'Check Driver Ban' in blue text. The card contains two input fields: 'JMBG' and 'Location'. To the right of the 'Location' field is a blue button labeled 'Check Driver Ban'. Below the card, there are two more buttons: 'Check Driver' and 'Go Back'. At the bottom of the page, there is a dark blue footer with the text 'Made by: TISEU Team 2, 2024' in white.

Slika 10. - Forma za unos podataka tokom provere vozača za upravljanje vozilom pod zabranom upravljanja vozilom

Forma za unos podataka tokom provere vozila kojim upravlja vozač sa prosleđenim JMBG-om, takođe se prosleđuje tip guma i lokacija na kojoj je saobraćajna patrola zaustavila vozača sa vozilom

The screenshot shows a web application interface for 'eUprava'. At the top, there is a dark blue header with the text 'eUprava' in white. Below the header, a light blue box contains the title 'Traffic Violations' in blue text. Inside this box, there is a white card titled 'Check Vehicle Tire' in blue text. The card contains three input fields: 'JMBG', a dropdown menu with 'Winter' selected, and 'Location'. To the right of the 'Location' field is a blue button labeled 'Check Vehicle Tire'. Below the card, there is a blue button labeled 'Check Driver'. At the bottom of the page, there is a dark blue footer with the text 'Made by: TISEU Team 2, 2024' in white.

Slika 11. - Forma za proveru guma na vozilu

Forma za unos podataka tokom provere alkohola u krvi vozaču koji upravlja vozilom u saobraćaju, prosleđuje se alkohol u krvi, koji je očitao saobraćajni policajac sa odgovarajućim aparatom, prosleđuje se JMBG vozača kao i lokacija na kojoj je vozač zaustavljen od strane saobraćajne patrola

The screenshot shows a web interface for the 'eUprava' system. At the top, there is a dark blue header with the text 'eUprava' in white. Below the header, the main content area has a light blue background with the title 'Traffic Violations' in a darker blue font. Centered on this page is a white rectangular form titled 'Check Alcohol Level' in green text. The form contains three input fields: 'JMBG', 'Alcohol Level', and 'Location'. To the right of the 'Location' field is a purple button labeled 'Check Alcohol Level'. At the bottom of the form, there is a small text attribution: 'Made by: TISEU Team 2, 2024'.

Slika 12. - Forma za proveru alkohola u krvi kod vozača

Forma za proveru svih prethodno navedenih parametara kao i proveru registracije vozila, u formu se unosi JMBG vozaca, alkohol u krvi kolicina, tip guma na vozilu, broj tablica i lokacija na kojoj je zaustavljen vozač od strane saobraćajne patrola

The screenshot shows a web interface for the 'eUprava' system, similar to the previous one. It features a dark blue header with 'eUprava' in white. The main content area has a light blue background with the title 'Traffic Violations' in a darker blue font. Centered on this page is a white rectangular form titled 'Check Driver' in green text. The form contains five input fields: 'JMBG', 'Alcohol Level', a dropdown menu currently showing 'SUMMER', 'Plates Number', and 'Location'. To the right of the 'Location' field is a purple button labeled 'Check Driver'.

Slika 13. - Forma za proveru svih implementiranih parametara kod vozača

## **9. Zaključak**

U ovom seminarskom radu je opisan je projekat koji olakšava saobraćajnoj policiji rad sa vozačima i vozilima kojima vozači upravljaju. Ovaj projekat automatizuje proveru vozač i vozila kojim vozač upravlja. Takođe i prilikom nekih neispravnosti kod vozača ili vozila, ovaj sistem automatizuje kreiranje kazne i izveštaja koji se prosleđuje sudu. Stvari koje su morale nekad da se unose ili zapisuju ručno, sada su postale automatizovane i automatski se generišu na osnovu prosleđenih vrednosti.

Kroz projekat je prikazano jako veliko olakšanje obrade podataka i pružen korisnicima intuitivan interfejs i olakšan rad kao zaposlenim licima.



## 10. Literatura

[1] Go programski jezik - <https://go.dev/>

[2] React - <https://react.dev/>

[3] MongoDB - <https://www.mongodb.com/>

[4] Docker - <https://www.docker.com/>