

Web-temeljen sustav za upravljanje osobljem i resursima vojske

Budimir, Franko

Undergraduate thesis / Završni rad

2024

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Electrical Engineering and Computing / Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva**

Permanent link / Trajna poveznica: <https://urn.nsk.hr/urn:nbn:hr:168:179080>

Rights / Prava: [In copyright](#) / [Zaštićeno autorskim pravom.](#)

Download date / Datum preuzimanja: **2025-03-14**



Repository / Repozitorij:

[FER Repository - University of Zagreb Faculty of Electrical Engineering and Computing repozitory](#)



SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1588

**WEB-TEMELJEN SUSTAV ZA UPRAVLJANJE OSOBLJEM I
RESURSIMA VOJSKE**

Franko Budimir

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1588

**WEB-TEMELJEN SUSTAV ZA UPRAVLJANJE OSOBLJEM I
RESURSIMA VOJSKE**

Franko Budimir

Zagreb, lipanj 2024.

ZAVRŠNI ZADATAK br. 1588

Pristupnik: **Franko Budimir (0036532049)**
Studij: Elektrotehnika i informacijska tehnologija i Računarstvo
Modul: Računarstvo
Mentor: doc. dr. sc. Mario Brčić

Zadatak: **Web-temeljen sustav za upravljanje osobljem i resursima vojske**

Opis zadatka:

Ovaj završni rad treba pokazati proces razvoja sveobuhvatnog web-temeljenog sustava za upravljanje sadržajem (engl. Content Management System, CMS) dizajniranog za vojne svrhe, s fokusom na upravljanje osobljem i resursima. Sustav treba uključivati značajke poput jednostavne analitike podataka, formiranje vojnih jedinica te interaktivne karte za poboljšanje strategijskog planiranja kao i geolociranje vojnih jedinica. Implementacija mora koristiti moderni programski okvir.

Rok za predaju rada: 14. lipnja 2024.

Sadržaj

Uvod	1
1. Zahtjevi.....	2
1.1. Korisnički zahtjevi.....	2
1.2. Funkcionalni zahtjevi	2
2. Model rješenja	4
2.1. Konceptualni model baze podataka.....	4
2.1.1. User.....	4
2.1.2. Soldier.....	5
2.1.3. Unit.....	5
2.1.4. Order.....	6
2.1.5. Equipment.....	6
2.2. Koncept sučelja.....	7
2.2.1. Primjer korištenja	9
2.3. Koncept pretpostavljenog oblika mreže i sigurnost podataka	10
2.3.1. Koncept toka podataka	12
3. Arhitektura rješenja	14
3.1. Korištene tehnologije u aplikaciji za sučelje	14
3.2. Korištene tehnologije u poslužiteljskoj aplikaciji	17
3.3. Korištene tehnologije u bazi podataka	18
4. Implementacija	19
4.1. Implementacija aplikacije za sučelje	19
4.1.1. Stranica za prijavu	20
4.1.2. Glavna nadzorna ploča	20
4.1.3. Stranice s pregledom entiteta.....	21

4.1.4.	Stranice pojedinačnih entiteta.....	22
4.1.5.	Enkripcija i dekrepcija podataka.....	22
4.1.6.	Interaktivna karta.....	23
4.1.7.	Analitika.....	24
4.2.	Implementacija poslužiteljske aplikacije.....	25
4.2.1.	Usmjerivači i kontroleri.....	26
4.2.2.	Enkripcija i dekrepcija podataka.....	26
5.	Upute za lokalno pokretanje.....	27
5.1.	Potrebni programi i preduvjeti.....	27
5.2.	Konfiguracija baze podataka.....	27
5.3.	Pokretanje aplikacija.....	28
	Zaključak.....	29
	Literatura.....	30
	Sažetak.....	33
	Summary.....	34

Uvod

Ovaj rad istražuje razvojni proces sveobuhvatnog web-baziranog sustava za upravljanje sadržajem (engl. Content Management System, CMS) nazvanog Parnassus, dizajniranog za vojne potrebe, s posebnim fokusom na upravljanje osobljem i resursima. Sustav je namijenjen da sadrži mogućnosti kao što su jednostavna analiza podataka, formiranje vojnih jedinica i interaktivne karte za poboljšanje strateškog planiranja i geolociranja vojnih jedinica. Implementacija koristi moderni softverski skup koji se sastoji od Next.js-a i Reacta za aplikaciju koja se bavi sučeljem te Node.js s Expressom, Prismom i Supabaseom s PostgreSQL-om za odvojenu aplikaciju koja ima ulogu servera.

Učinkovito upravljanje vojnim osobljem i resursima ključni je aspekt modernih vojnih operacija. Uz sve veću složenost vojne logistike i raspoređivanja osoblja, robustan i skalabilan CMS postaje nezamjenjiv. Ovaj sustav ima cilj pojednostaviti te procese pružanjem intuitivnih sučelja i strateških alata koji mogu pomoći u donošenju odluka i operativnom planiranju. Uz to, uzimajući u obzir potencijalne prijetnje kakvima se često izlažu takvi sustavi u svijetu, prioritet sigurnosti prijenosa podataka u ovome sustavu je na povećanoj razini. Predviđeni korisnici ovog sustava su administrativni radnici imenovani od strane vlasnika sustava, koji imaju privilegije i odgovornosti upravljanja resursima i jedinicama, kao i strategiziranja operativnih aktivnosti. Od sada nadalje, u ovom radu, takvi će biti referencirani jednostavno kao „korisnici“

U prvom poglavlju prolazimo kroz sve korisničke i funkcionalne zahtjeve. U drugom poglavlju prikazani su modeli rješenja tih zahtjeva, primjer toka podataka od kreiranje nove vojne jedinice do povratka informacija te dijagram procesa komunikacije između aplikacija (od slanja zahtjeva klijenta do dobivanja odgovora). Treće poglavlje bavi se arhitekturom rješenja, tj. korištenim tehnologijama i komponentama od kojih su aplikacije sastavljene. U četvrtom poglavlju opisano je kako su svi zahtjevi implementirani te su prikazane slike ekrana aplikacije. Konačno, u posljednjem poglavlju prikazane su upute za lokalno pokretanje izvornog koda.

1. Zahtjevi

1.1. Korisnički zahtjevi

Sustav mora imati:

- Intuitivni i responzivni dizajn
- Mogućnost stvaranja, dobivanja, uređivanja i brisanja korisnika, vojnika, vojnih jedinica, opreme i naredbi
- Mogućnost korištenja interaktivne karte u svrhu strategiziranja i geolociranja vojnih jedinica
- Mogućnost čitanja osnovne analitike resursa

1.2. Funkcionalni zahtjevi

Autorizacija i autentifikacija:

Sustav mora pružati osnovni oblik autentifikacije i autorizacije korisnika kako bi se izbjeglo korištenje sustava sa strane korisnika malicioznih motiva.

Mogućnost upravljanja resursima:

Sustav mora pružati mogućnost stvaranja, dobivanja, uređivanja i brisanja korisnika, vojnika, vojnih jedinica, opreme i naredbi na efikasan način.

Interaktivna karta:

Sustav mora pružati mogućnost prikazivanja interaktivne karte s opcijama za crtanje oblika te geolociranje vojnih jedinica u svrhe strategiziranja izvedbi izdanih naredbi.

Osnovna analitika:

Sustav treba pružati vizualno očitavanje osnovne analitike stanja resursa u bazi podataka, npr. vizualni prikaz količine vojnika poginulih u borbi (KIA) u usporedbi s količinom onih koji su nestali u borbi (MIA).

Aplikacija za sučelje:

Sustav mora sadržavati odvojenu aplikaciju zaduženu za prikaz sučelja i komunikaciju s korisnicima. Od sada nadalje, ova aplikacija će biti referencirana kao „Aplikacija za sučelje“.

Poslužiteljska aplikacija:

Sustav mora sadržavati odvojenu aplikaciju zaduženu za komunikaciju s aplikacijom za sučelje i bazom podataka. Od sada nadalje, ova aplikacija će biti referencirana kao „Poslužiteljska aplikacija“.

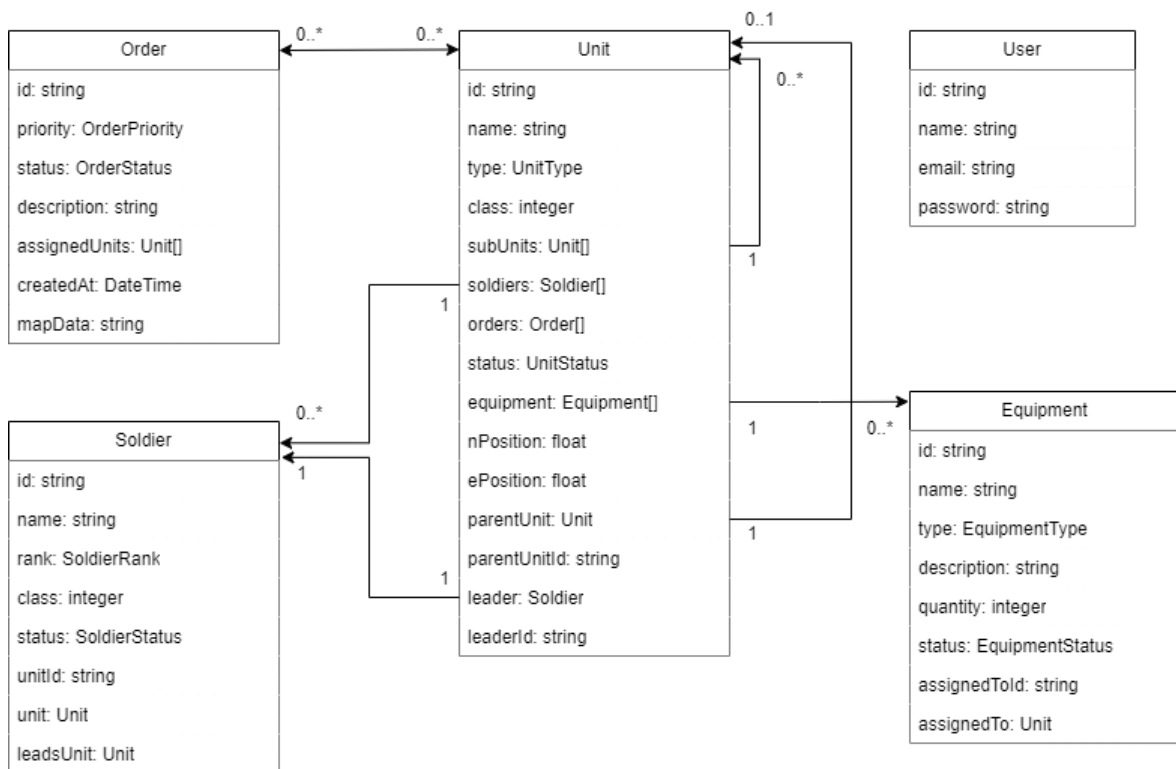
Sigurnost prijenosa podataka:

Sustav mora pružati povjerljivost i integritet podataka u prijenosu između aplikacija.

2. Model rješenja

2.1. Konceptualni model baze podataka

Konceptualiziranje modela većine entiteta nije bio problem zbog intuitivne i jednostavne povezanosti među njima, no najveći izazov u stvaranju modela baze podataka bio je odlučivanje oblika vojnih jedinica. Slika 2-1 prikazuje finalni koncept modela baze podataka.



Slika 2- 1

2.1.1. User

Model entiteta User (hrv. Korisnik) je jednostavna reprezentacija administrativnog tijela s privilegijama upravljanja svih aspekata sustava te sadrži ime korisnika, e-poštu i lozinku.

2.1.2. Soldier

Model entiteta Soldier (hrv. Vojnik) sadrži detalje individualnog vojnika. Svaki vojnik ima klasu (1-5) kojoj pripada koja predstavlja kojoj klasi jedinice smije biti vođa, tj. vojnik smije voditi jedinicu iste ili niže klase od svoje. SoldierRank je enumeracija formalne titule koju vojnik ima (npr. SGT za *Sergeant*), a SoldierStatus je njegovo trenutno stanje (npr. ACTIVE što ukazuje da je vojnik trenutno raspoloživ).

SoldierRank enumeracija uključuje: PVT, PFC, CPL, SGT, SSGT, SFC, MSGT, FSG, SGM, SLT, FLT, CPT, MAJ, LTC, COL, BG, MG, LTG, GEN.

SoldierStatus enumeracija uključuje: ACTIVE, KIA, MIA, WIA, RETIRED, AWOL.

2.1.3. Unit

Model entiteta Unit (hrv. Jedinica) predstavlja vojnu jedinicu koja kompozitnim obrascem može sadržavati (i imati iste funkcionalnosti kao) njezine podjedinice (subUnits) i roditeljsku jedinicu (parentUnit) čime formira stablo jedinica što će nam biti korisno u realizaciji implementacije stvaranja realističnih vojnih jedinica i formacija.

Primjer oblika vojne jedinice prikazan je na slici 2-2.

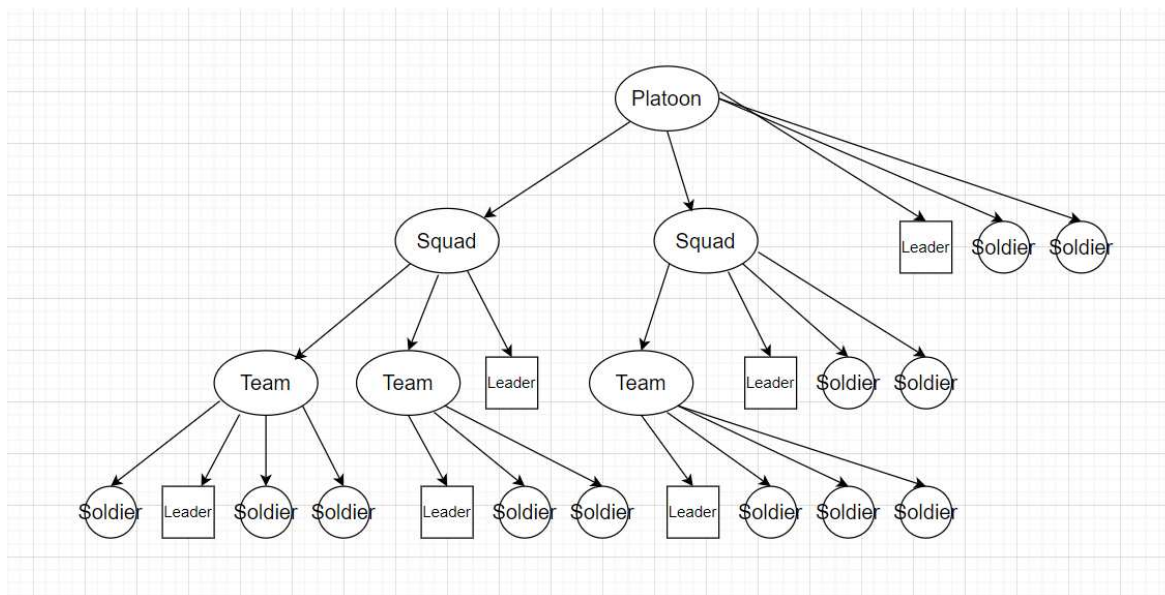
UnitType enumeracija uključuje: TEAM, SQUAD, PLATOON, COMPANY, BATTALION i TASK_FORCE.

Ako nije posebno navedeno, u kreaciji jedinice, klasa se automatski pridjeljuje vrsti (*type*) jedinice: Team je klase 1, Squad klase 2, Platoon klase 3, Company klase 4, Battalion klase 5 te Task Force klase 6. Jedinicu mora voditi vođa (*Leader*) koji je stoga i vojnik u toj jedinici te vođa mora biti iste ili više klase od te jedinice. Roditeljska jedinica neke jedinice mora biti više klase, a podjedinice niže klase od te jedinice. Task Force je posebna vrsta jedinice te njoj može biti vođa bilo koji vojnik bilo koje klase.

Jedinici se, također, može pridjeliti oprema (*Equipment*) i naredbe (*Order*).

UnitStatus enumeracija uključuje: ACTIVE i DECOMMISSIONED.

Kada se *decommissioned* status pridijeli nekoj jedinici, vođa, svi vojnici, sva oprema i sve naredbe te jedinice se automatski odvajaju od nje.



Slika 2- 2

2.1.4. Order

Model entiteta Order (hrv. Naredba) predstavlja naredbu izdanu određenim vojnim jedinicama. Naredba sadrži informacije poput zaduženih vojnih jedinica, deskripcija zadatka te podatke o oblicima i crtežima nacrtanima na interaktivnoj karti.

OrderStatus enumeracija uključuje: PENDING, IN_PROGRESS, COMPLETED, FAILED i CANCELLED, a OrderPriority može biti: LOW, MEDIUM i HIGH.

2.1.5. Equipment

Model entiteta Equipment (hrv. Oprema) predstavlja osnovne informacije neke opreme kao tip opreme, ime opreme, stanje opreme te kojoj vojnoj jedinici (samo jednoj) je ta oprema pridružena.

EquipmentStatus enumeracija može biti: ACTIVE, UNDER_MAINTENANCE, DECOMMISSIONED. Mijenjanjem statusa neke opreme u *decommissioned* odvaja tu opremu od jedinice kojoj je bila pridružena.

2.2. Koncept sučelja

Koncept sučelja ovog vojnog CMS-a može se raščlaniti na nekoliko ključnih komponenti:

Stranica za prijavu:

Početna ulazna točka sustava je stranica za prijavu. Za pristup svim ostalim dijelovima sustava korisnik mora biti prijavljen, čime se osigurava sigurnost i kontrolirani pristup.

Glavna nadzorna ploča:

Nakon prijave, korisnik se usmjerava na glavnu nadzornu ploču. Ova nadzorna ploča djeluje kao središnje čvorište aplikacije koje pruža brzi pristup različitim dijelovima sustava. Sadrži gumbe ili poveznice koje vode na pregledne tablice različitih entitete baze podataka: vojnike, jedinice, opreme, naredbe i administratore. Dodatno, postoji odjeljak za analitiku.

Stranice s pregledom entiteta:

Vojnici, jedinice, oprema, naredbe, administratori: Svaki entitet ima posebnu stranicu koja prikazuje tablicu s popisom svih zapisa za taj entitet. Na primjer, stranica Soldiers navodi sve vojnike s detaljima poput imena, ranga, statusa i jedinice. Svaka tablica uključuje opciju "Add new" za dodavanje novog zapisa.

Stranice pojedinačnih entiteta:

Klikom na određeni zapis u tablici bilo kojeg entiteta vodi korisnika do detaljnog prikaza tog zapisa.

Vojnici: Detaljne informacije o vojniku, kao što su ime, rang, status i jedinica kojoj pripadaju.

Jedinice: Informacije o jedinici, uključujući vođu, matičnu jedinicu, podjedinice, dodijeljenu opremu i izravne/neizravne naredbe.

Oprema: Detaljan prikaz opreme, uključujući njezin status i namjenu.

Naredbe: Detaljne informacije o naredbi, uključujući opis, prioritet, status i interaktivnu kartu za izradu strategije.

Administratori: Pojediniosti o korisnicima.

Stranica analitike:

Dostupna putem nadzorne ploče, ova stranica pruža vizualne prikaze podataka (grafikone) koji se odnose na različite entitete kao što su statusi vojnika, statusi pojedinih tipova jedinica itd. To pomaže u analizi i razumijevanju podataka i statusa unutar sustava.

2.2.1. Primjer korištenja

Ovdje istražujemo jedan specifičan primjer korištenja sustava sa strane korisnika (administratora).

Prijava: Korisnik se prijavljuje u sustav.

Pristup nadzornoj ploči: Korisnik se odvodi na nadzornu ploču.

Navigacija entiteta: Korisnik klikne na “*Units*“ za pregled svih jedinica.

Kreiranje entiteta: Korisnik klikom na “*Add new*” i izrađuje novu jedinicu koju povezuje s već postojećim podjedinicama i vojnicima.

Detaljan prikaz: Korisnik klikne na tu određenu jedinicu da vidi detaljne informacije o toj jedinici.

Pristup analitici: Korisnik, također, može otići do analitičke stranice za pregled grafikona i statistike o podacima sustava.

Ovaj sustav osigurava da su podaci organizirani, dostupni i sigurni, s jasnom strukturom za navigaciju između općih pregleda i detaljnih informacija. Dodavanje interaktivnih elemenata kao što je interaktivna karta za naredbe (u *Order* stranici) poboljšava funkcionalnost, posebno za strateško planiranje i vizualizaciju.

2.3. Koncept pretpostavljenog oblika mreže i sigurnost podataka

Ovdje istražujemo pretpostavljen oblik mreže u kojoj operira ovaj sustav kako bismo kasnije poviše objasnili sigurnost i zaštitu podataka u određenim koracima toka podataka.

Slika 2-3 prikazuje pretpostavljenu shemu i primjer oblika mreže u kojoj ovaj sustav operira kako bi se maksimizirala sigurnost podataka.

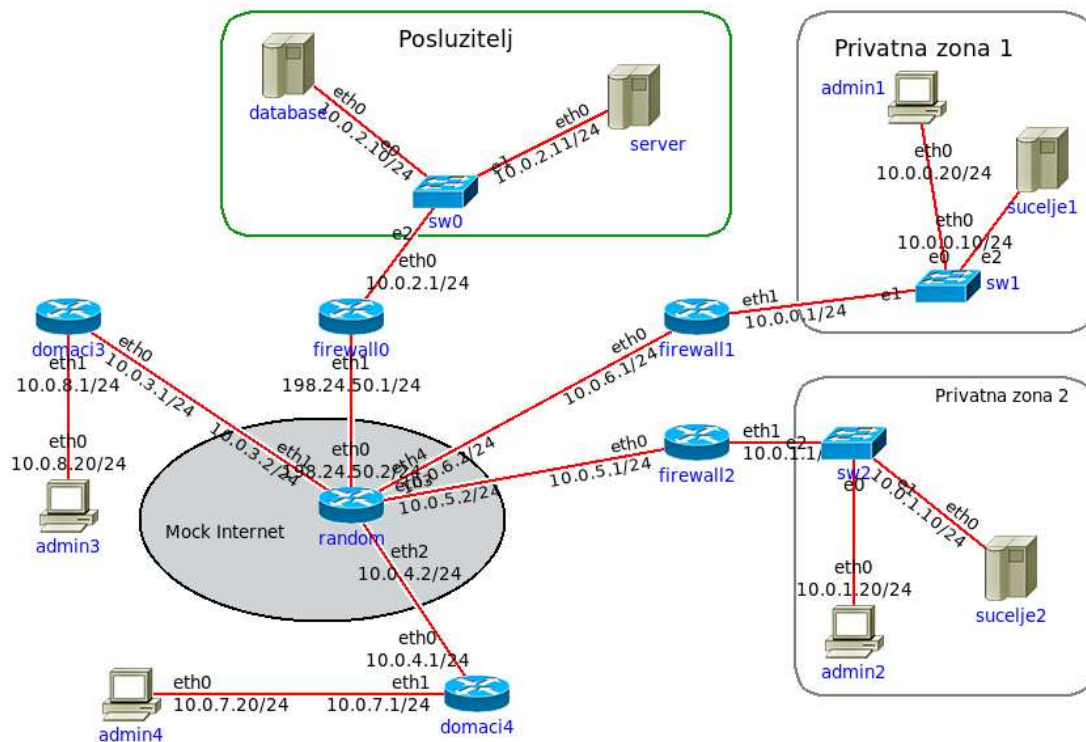
Naime, zbog prirode namjene ovoga sustava, dopustiti da se aplikaciji za sučelje pristupa preko interneta je generalno loša ideja u kontekstu sigurnosti podataka, stoga je, u sklopu cilja ovog rada, pretpostavljeno da se individualne instance pokrenutih aplikacija za sučelje vrte na osobnim računalima korisnika ili na prikladno zaštićenoj lokalnoj mreži (npr. strateška točka, vojna baza) u kojoj se nalazi www-poslužitelj aplikacije za sučelje skupa s korisnicima. Nadalje, uzeći u obzir da je pretpostavljeno da je samo jedan www-poslužitelj poslužiteljske aplikacije sustava, komunikacija te aplikacije i individualnih instanci pokrenutih aplikacija za sučelje treba biti zaštićena jer se obavlja preko Interneta.

Pretpostavka je da se ne koristi nužno HTTPS, već da podaci moraju biti zaštićeni i u okruženju s običnim HTTP protokolom.

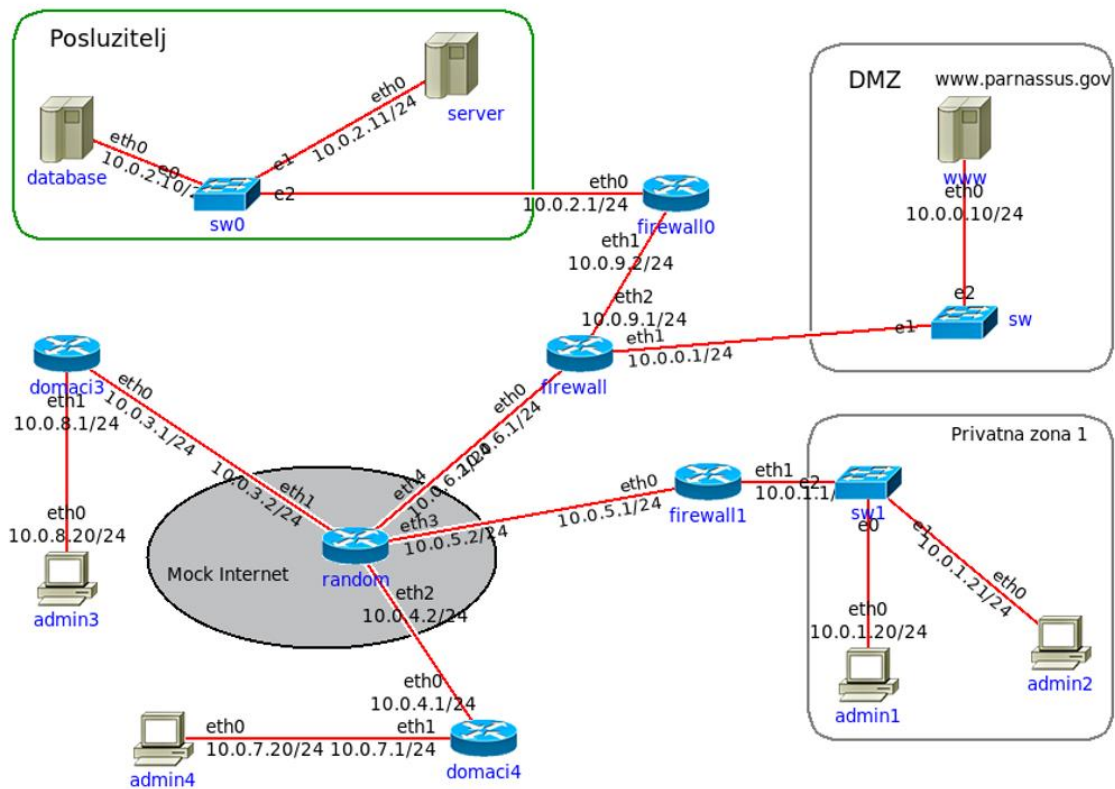
Enkripcija podataka i autorizacijskog tokena u zaglavlju zahtjeva poslanih između aplikacije za sučelje i poslužiteljske aplikacije je simetrična enkripcija s istim tajnim ključevima kojih posjeduju korisnici (administratori) i sama poslužiteljska aplikacija, a specifična tehnologija korištena za enkripciju je poviše objašnjena kasnije.

Još jedna pretpostavka je da se baza podataka nalazi u istoj zasebnoj sigurnoj lokalnoj mreži kao i www-poslužitelj poslužiteljske aplikacije.

Aplikacija za sučelje se, alternativno, može nalaziti i pokretati na vlastitom jedinstvenom sjedištu u nekakvoj demilitariziranoj zoni (slika 2-4), no takav pristup zahtjeva povećanu sigurnost samog sustava i toka podataka između klijentske i poslužiteljske strane aplikacije za sučelje (HTTPS, javni i tajni ključevi, digitalni potpisi itd.) koja nije u sklopu cilja ovog rada.



Slika 2- 3



Slika 2- 4

2.3.1. Koncept toka podataka

Ovdje istražujemo jedan primjer korištenja aplikacije kako bi poviše objasnili tok podataka od korisnikove akcije do povrata informacija s poslužiteljske strane.

Stvaranje zahtjeva na klijentskoj strani: Ažurirana je vojna jedinica na klijentskoj strani aplikacije za sučelje.

Slanje podataka s klijentske strane na poslužiteljsku stranu aplikacije za sučelje: Ovaj proces prijenosa nije posebno zaštićen jer se pretpostavlja sigurnost lokalne mreže u kojoj je pokrenuta aplikacija za sučelje (pogledati sliku 2-3).

Enkripcija i slanje podataka na poslužiteljsku aplikaciju: Ova komunikacija je zaštićena tako da se podaci enkriptiraju jednim, a sjednica pohranjena u autorizacijskom tokenu zaglavlja drugim tajnim ključem. Enkriptirani podaci s autorizacijskim tokenom u zaglavlju se šalju na poslužiteljsku aplikaciju.

Provjera autentifikacije: Token sjednice u autorizacijskom zaglavlju se dekriptira u *middleware* funkciji tajnim ključem za enkripciju autorizacijskog tokena te se kao dodatna mjera sigurnosti provjerava postojanost korisnika sjednice.

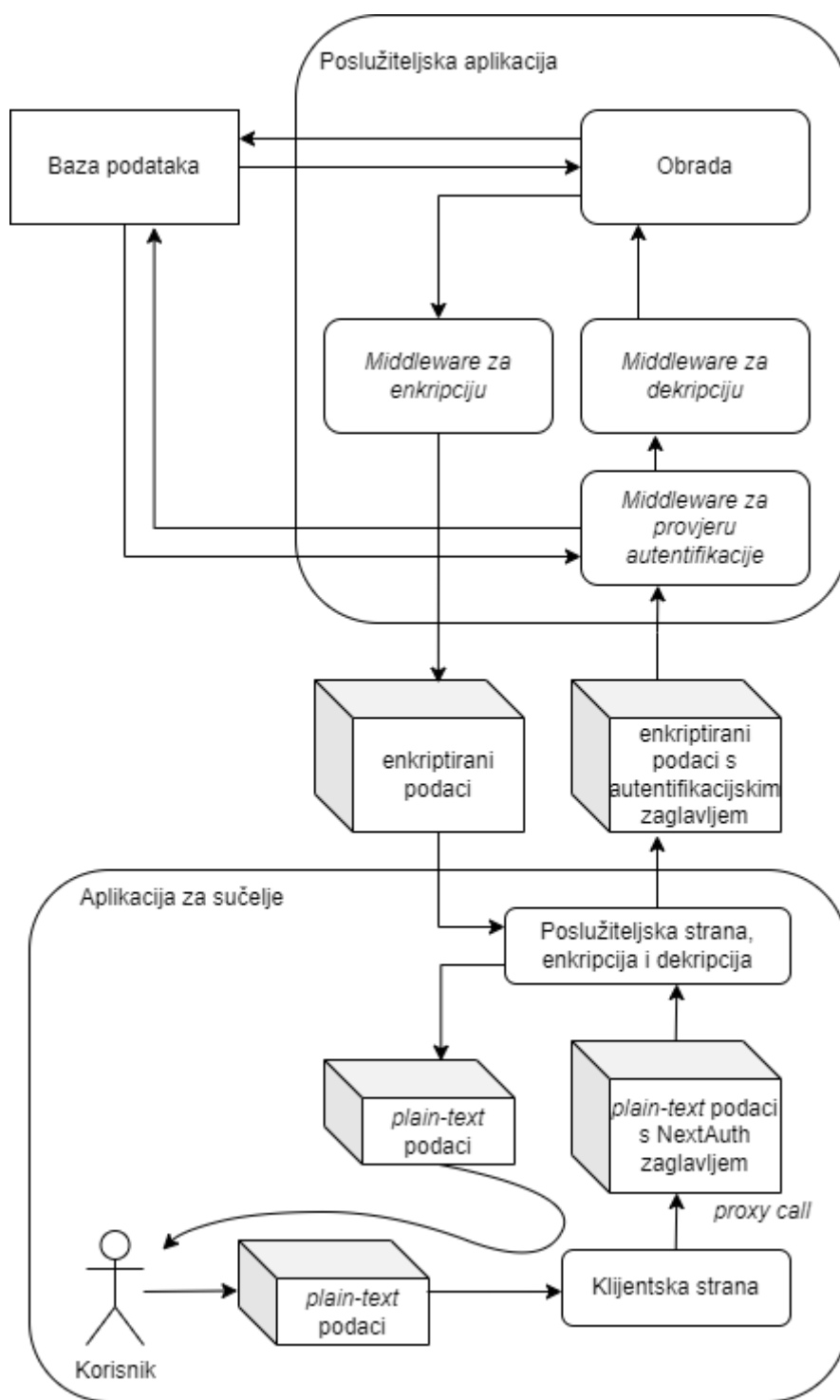
Dekripcija podataka: Podaci se dekriptiraju za to zaduženim tajnim ključem čime se dodatno provjerava autorizacija i, uz to, integritet podataka.

Obrada podataka: Podaci se prikladno obrađuju na poslužiteljskoj aplikaciji i pristupa se bazi podataka po potrebi (koja je, po pretpostavci, u sigurnoj komunikaciji s poslužiteljskom aplikacijom).

Enkripcija i slanje podataka nazad na aplikaciju za sučelje: Podaci u odgovoru se enkriptiraju s istim tajnim ključem za enkripciju podataka te se takvi šalju kao odgovor nazad aplikaciji za sučelje.

Dobivanje podataka nazad na aplikaciji za sučelje: Enkriptirani podaci u odgovoru poslužiteljske aplikacije se dekriptiraju tajnim ključem za enkripciju podataka te se šalju na klijentsku stranu samom korisniku koji, zatim, vidi ažuriranu vojnu jedinicu.

Dijagram toka podataka je prikazan na slici 2-5.



Slika 2- 5

3. Arhitektura rješenja

3.1. Korištene tehnologije u aplikaciji za sučelje

React:

React je široko korištena JavaScript biblioteka koju je razvio Facebook za izgradnju korisničkih sučelja, posebno jednostraničnih aplikacija gdje su učinkovito rukovanje podacima i dinamičko ažuriranje sadržaja ključni. Omogućuje razvojnim programerima stvaranje komponenti korisničkog sučelja za višekratnu upotrebu, koje sadrže njihovo vlastito stanje i ponašanje, što dovodi do modularnog koda koji se može održavati.

Next.js:

Next.js moćan je i svestran okvir izgrađen na temelju Reacta, dizajniran da omogući programerima da s lakoćom stvaraju optimizirane web aplikacije visokih performansi. Pojednostavljuje proces izgradnje i web-stranica prikazanih na poslužiteljskoj strani i statički generiranih web-stranica pružanjem robusnog skupa značajki kao što su automatsko dijeljenje koda, renderiranje na strani poslužitelja (SSR), generiranje statičkih web-mjesta (SSG) i API rute. Next.js također podržava inkrementalnu statičku regeneraciju, koja programerima omogućuje ažuriranje statičkog sadržaja nakon vremena izgradnje.

Tailwind CSS:

Tailwind CSS je CSS okvir koji pruža sveobuhvatan skup unaprijed definiranih klasa za brz razvoj korisničkog sučelja. Za razliku od tradicionalnih CSS okvira, Tailwind omogućuje razvojnim programerima da stiliziraju svoje komponente izravno unutar HTML-a koristeći pomoćne klase, što rezultira visoko prilagodljivim i responzivnim sustavom dizajna bez pisanja prilagođenog CSS-a.

NextAuth.js:

NextAuth.js je robusna biblioteka za provjeru autentifikacije dizajnirana za Next.js aplikacije, pružajući jednostavan i siguran način za rukovanje provjerom autentifikacije. Podržava više metoda autentifikacije, uključujući *email/lozinku*, OAuth pružatelje (kao što su Google, Facebook i GitHub) i prilagođene vjerodajnice. NextAuth.js nudi ugrađenu podršku za sigurne sesije, upravljanje korisnicima i rukovanje tokenima, što olakšava integraciju i upravljanje tokovima provjere autentifikacije u Next.js projektu.

Leaflet:

Leaflet je nezahtjevna JavaScript biblioteka otvorenog koda za interaktivne karte, koja nudi API-je jednostavne za korištenje za izradu prilagodljivih karata na *web-u*. Podržava različite slojeve karte, markere, skočne prozore i vektorske slojeve, što ga čini idealnim za ugrađivanje dinamičkih karata u web aplikacije.

Recharts:

Recharts je biblioteka grafikona izgrađena s Reactom, koja nudi jednostavan i intuitivan API za stvaranje širokog raspona prilagodljivih grafikona i grafikona. Podržava različite vrste grafikona, kao što su linijski, trakasti, kružni i raspršeni, te omogućuje jednostavnu integraciju interaktivnih i responzivnih vizualizacija. Recharts koristi SVG elemente i vrlo je prilagodljiv, što ga čini izvrsnim izborom za vizualni prikaz podataka u ovom radu.

shadcn/ui:

shadcn je sveobuhvatna biblioteka komponenti za React aplikacije, koja pruža zbirku unaprijed dizajniranih i prilagodljivih komponenti korisničkog sučelja. Naglašava jednostavnost korištenja i besprijekornu integraciju s modernim React projektima, omogućujući programerima da brzo izgrade i stiliziraju korisnička sučelja.

jsonwebtoken:

jsonwebtoken je popularna JavaScript biblioteka koja se koristi za potpisivanje, provjeru i dekodiranje JSON web tokena (JWT). Olakšava sigurnu autentifikaciju na temelju tokena dopuštajući stvaranje potpisanih tokena koji se mogu provjeriti kako bi se osigurao integritet i povjerljivost podataka. Prema zadanim postavkama koristi algoritam HS256 (HMAC sa SHA-256) za potpisivanje i provjeru tokena, koristeći strategiju simetričnog ključa. To osigurava da i izdavatelj i primatelj mogu provjeriti autentičnost tokena koristeći isti tajni ključ.

Ostale korištene tehnologije:

Među ostalim korištenim tehnologijama su cors, biblioteka za upravljanje dijeljenjem resursa s različitim izvorima u aplikacijama Node.js, omogućavajući sigurne zahtjeve s različitih izvora, biblioteka axios za lakše stvaranje i rukovanje HTTP zahtjeva, zod biblioteka za provjeru valjanosti sheme koja pruža jednostavan i robustan način za definiranje i provjeru valjanosti struktura podataka, date-fns biblioteka koja nudi opsežan skup funkcija za manipuliranje i oblikovanje datuma u JavaScriptu te dotenv biblioteka za lakše upravljanje konfiguracijom u aplikacijama.

3.2. Korištene tehnologije u poslužiteljskoj aplikaciji

Node.js:

Node.js je JavaScript *runtime* koji omogućuje pokretanje JavaScripta na strani poslužitelja. Izgrađen je na Chromeovom V8 motoru i koristi I/O model vođen događajima, koji ne blokira, što ga čini učinkovitim i brzim. Node.js je izvrstan za izgradnju skalabilnih mrežnih aplikacija i usluga u stvarnom vremenu. Dolazi s bogatim skupom modula koji pomažu programerima da jednostavno kreiraju različite vrste aplikacija.

Express:

Express je web okvir za Node.js koji olakšava izradu web aplikacija i API-ja. Omogućava definiranje ruta za različite HTTP metode i primjenu *middleware* funkcija koje procesiraju zahtjeve prije nego što stignu do kontrolera koji onda komuniciraju s bazom podataka.

Prisma ORM:

Prisma je moderan skup alata za baze podataka koji programerima pomaže u radu s bazama podataka u web-aplikacijama. Čini pristup bazi podataka lakšim i sigurnijim pružajući alat za sastavljanje upita koji je siguran za tip i jednostavan način za definiranje shema baze podataka. Prisma podržava više baza podataka kao što su PostgreSQL, MySQL i SQLite i generira učinkovite upite baze podataka.

Postman:

Postman je moćan alat koji se koristi za testiranje API-ja. Programerima omogućuje slanje HTTP zahtjeva, provjeru odgovora i automatiziranje tijeka rada testiranja. S Postmanom se može jednostavno stvarati, spremati i organizirati API zahtjeve, čineći jednostavnijim uklanjanje pogrešaka i učinkovit razvoj API-ja.

Ostale korištene tehnologije:

Među ostalim korištenim tehnologijama su jsonwebtoken biblioteka u implementaciji enkripcije, cors i dotenv biblioteke (isto kao i u aplikaciji za sučelje) te bcrypt biblioteka za hashiranje lozinki algoritmom *Blowfish* prije pohrane u bazu podataka.

3.3. Korištene tehnologije u bazi podataka

PostgreSQL:

PostgreSQL je moćan sustav relacijskih baza podataka otvorenog koda. Podržava napredne značajke kao što su složeni upiti, strani ključevi, okidači i transakcije, što ga čini pouzdanim i fleksibilnim za širok raspon aplikacija. Poznat po svojoj stabilnosti i performansama, PostgreSQL se široko koristi u malim projektima i velikim poslovnim aplikacijama.

pgAdmin:

pgAdmin (verzija 4) je sveobuhvatan alat za upravljanje za PostgreSQL, kompatibilan s PostgreSQL verzijom 15 korištenoj u ovom sustavu. Nudi jednostavno web sučelje za jednostavno upravljanje i administraciju vaših PostgreSQL baza podataka. Uz pgAdmin 4 možete stvarati i mijenjati objekte baze podataka, pokretati SQL upite i nadzirati performanse baze podataka, čineći upravljanje bazom podataka jednostavnim i učinkovitim.

4. Implementacija

4.1. Implementacija aplikacije za sučelje

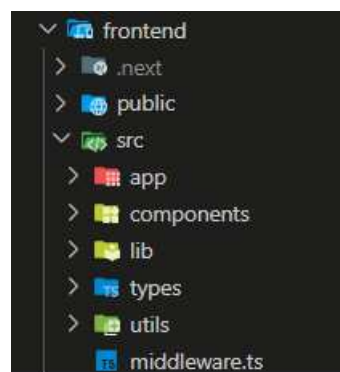
U ovom poglavlju opisujemo implementacijski dio projekta u kontekstu aplikacije za sučelje.

Struktura samog direktorija aplikacije za sučelje je prikazana na slici 4-1.

U ovom radu se ne bavimo prevođenjem/interpretacijom koda stoga nećemo ulaziti u objašnjenje direktorija `.next`.

Direktorij `public` je zadužen za slike, fontove i ostale datoteke korisne klijentskoj strani aplikacije.

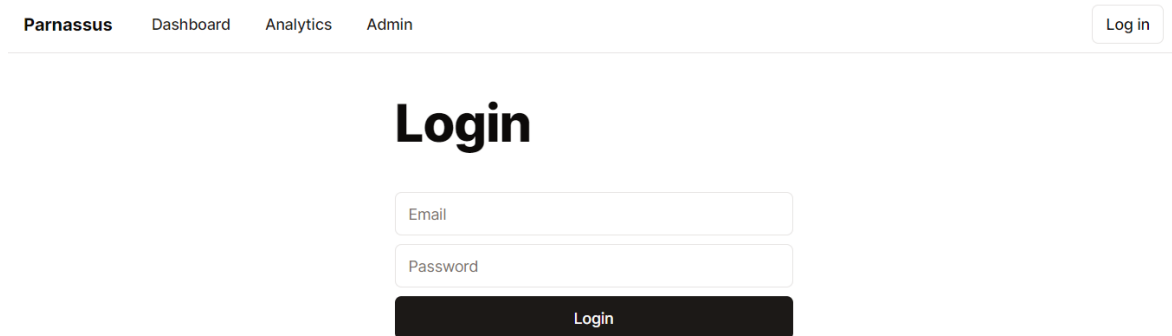
Direktorij `src` je direktorij koji sadrži *source code* aplikacije i, nadalje, ćemo se fokusirati na njegove poddirektorije. Direktorij `app` sadrži kod individualnih web-stranica kao i strukturu tih web-stranica u aplikaciji te `api` direktorij koji sadrži *API endpoint*-ove poslužiteljske strane aplikacije zadužene za komunikaciju između klijentske i poslužiteljske strane aplikacije. Direktorij `components` sadrži komponente (zamislite puzzle) koje višestruko koristimo u web-stranicama, npr. `Table` komponenta koja prikazuje tablicu vojnika. Direktorij `lib` služi za pohranu zajedničkih modula, funkcija i instanci koje se koriste u cijeloj aplikaciji, `types` direktorij sadrži tipove podataka korištenih u aplikaciji, `utils` sadrži pomoćne funkcije i alate te je `middleware.ts` zadužen za provjeru autorizacije pristupa web-stranicama.



Slika 4- 1

4.1.1. Stranica za prijavu

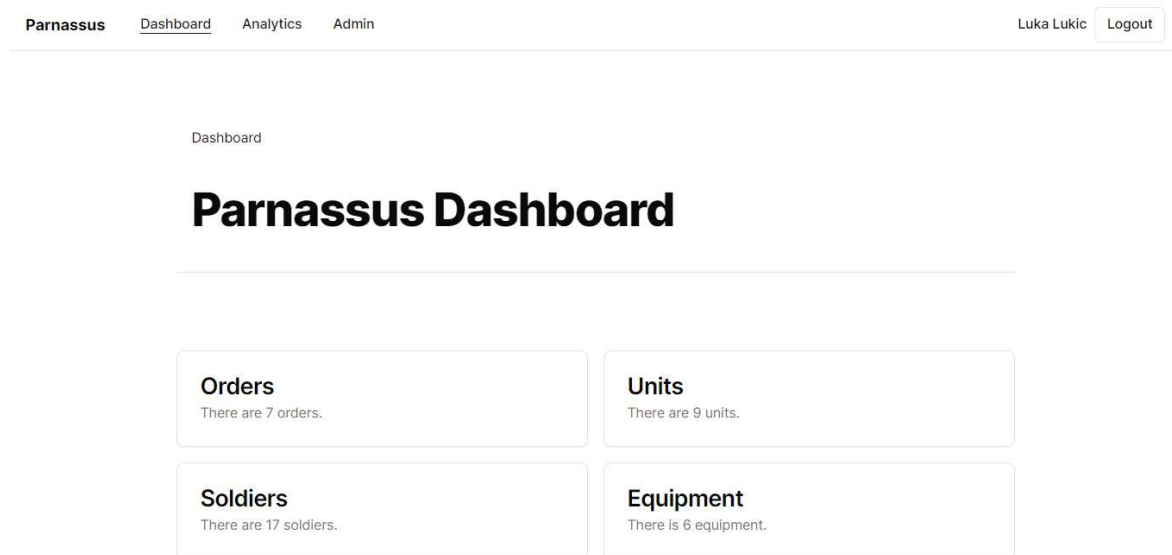
Stranica za prijavu (slika 4-2) je jednostavno napravljena koristeći dva polja za unos i gumb koji u pozadini poziva funkcije biblioteke NextAuth.js za autentifikaciju.



Slika 4- 2

4.1.2. Glavna nadzorna ploča

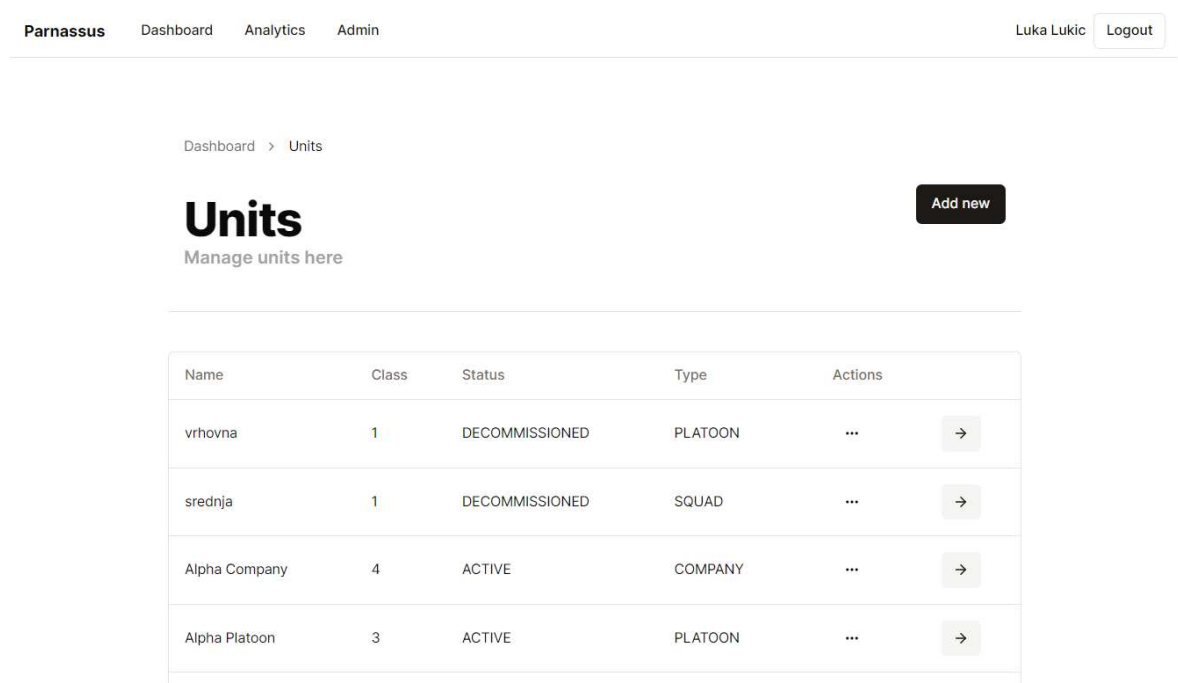
Glavna nadzorna ploča (slika 4-3) je glavno čvorište cijele aplikacije za sučelje. Naslov i individualne “kartice” koje korespondiraju određenom entitetu baze podataka su napravljeni uz pomoć Shadcn/ui biblioteke.



Slika 4- 3

4.1.3. Stranice s pregledom entiteta

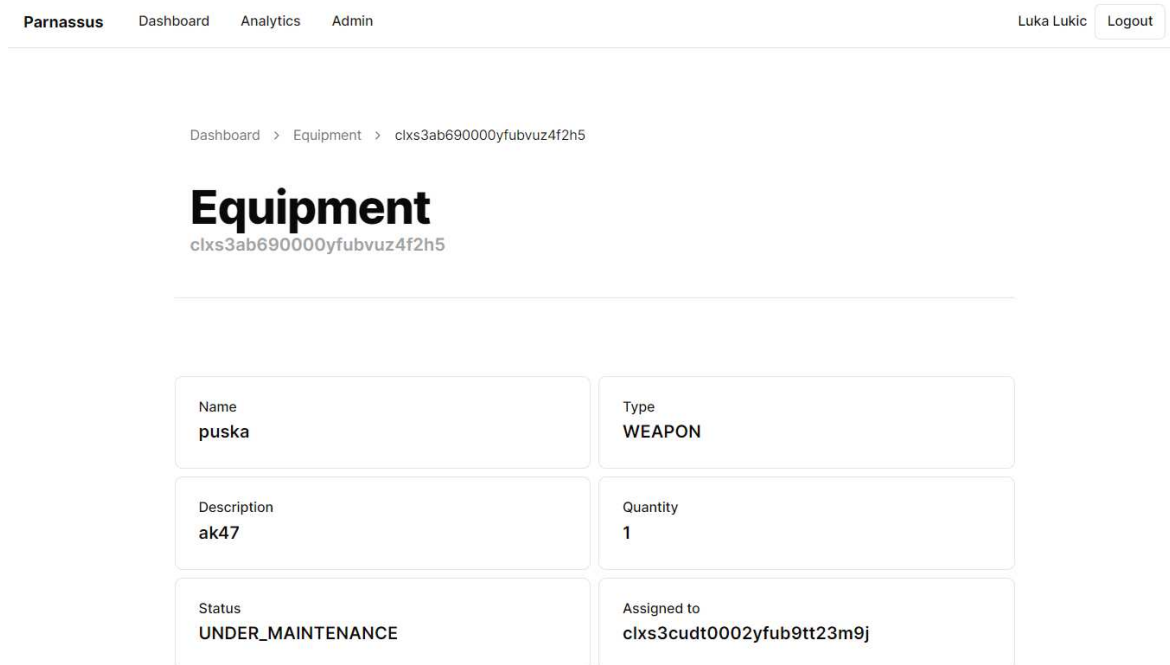
Stranica s pregledom određenog tipa entiteta (slika 4-4) je napravljena pomoću jednostavnog naslova, podnaslova i tablice individualnih entiteta iz Shadcn/ui biblioteke.



Slika 4- 4

4.1.4. Stranice pojedinačnih entiteta

Stranice pojedinačnih entiteta (slika 4-5) su isto tako napravljene pomoću Shadcn/ui biblioteke, koristeći uvedene komponente za naslov, podnaslov i informativne kartice.



Slika 4- 5

4.1.5. Enkripcija i dekripcija podataka

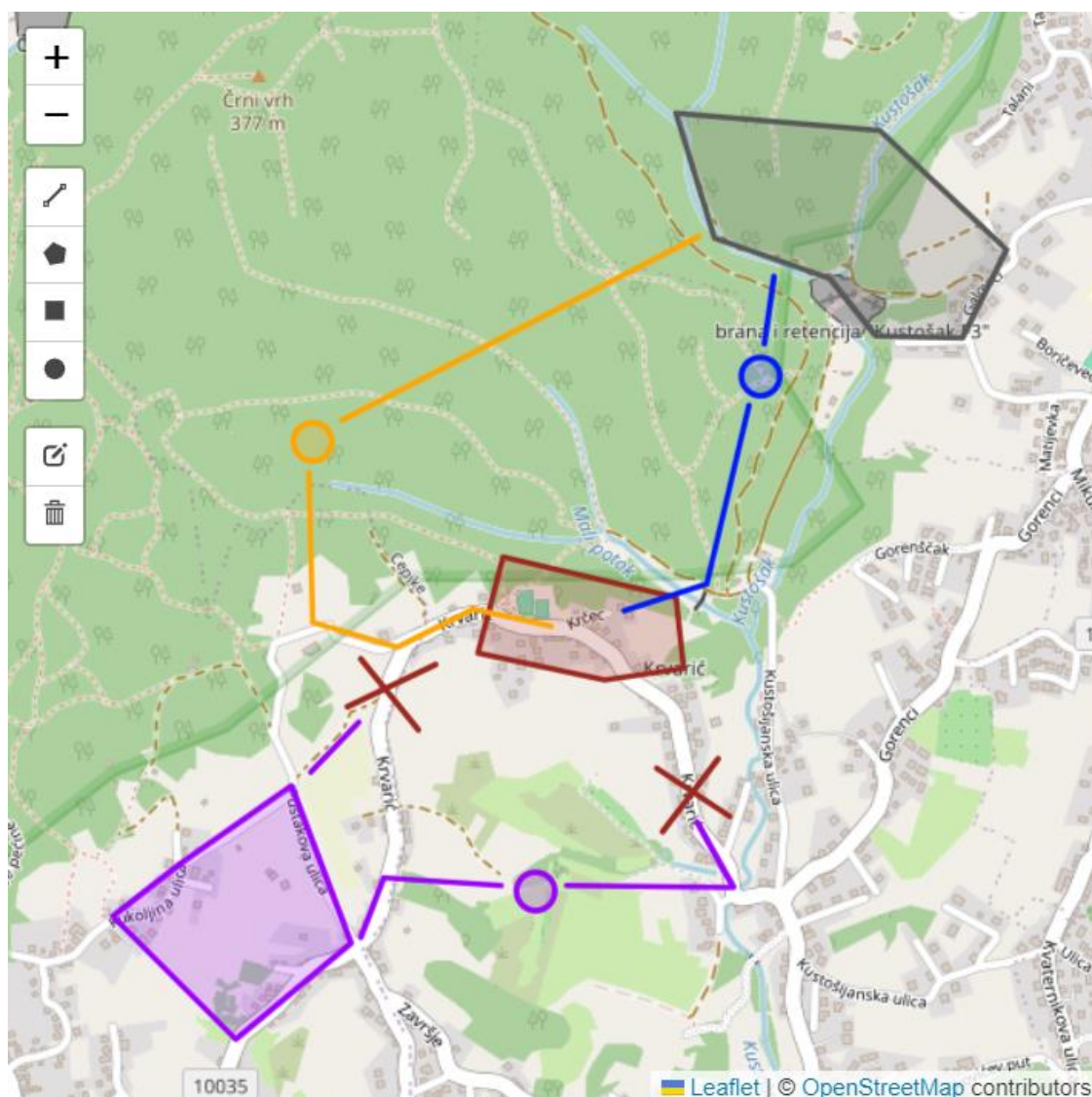
Enkripcija na poslužiteljskoj strani aplikacije se odvija u funkciji `secureFetch` koja služi kao nadogradnja konvencionalnoj ugrađenoj funkciji `fetch` koja pomoću biblioteke `Jsonwebtoken` od slanih podataka stvara *Json Web Token* (token koji u sebi sadrži enkriptirane podatke) koji ističe za 10 sekundi nakon njegovog potpisivanja u svrhu dodatne sigurnosti (ako bi taj zahtjev stigao na poslužiteljsku aplikaciju nakon 10 sekundi, token nebi više bio validan).

U slučaju slanja zahtjeva poslužiteljskoj aplikaciji s klijentske strane aplikacije, zahtjev se, radi sigurnosti, najprije proslijedi poslužiteljskoj strani aplikacije preko `/api/handler` rute kao *proxy*, jer se dohvaćanje tajnog ključa za enkripciju i sama enkripcija odvijaju na poslužiteljskoj, a ne klijentskoj strani aplikacije. U `/api/handler` ruti se, zatim, poziva `secureFetch` funkcija.

4.1.6. Interaktivna karta

Posebno napravljena React komponenta MapWithDraw, koja se nalazi u stranicama individualnih naredbi, stvara interaktivnu kartu koristeći Leaflet biblioteku. Korisnicima omogućuje crtanje oblika na karti i spremanje njihovih položaja i svojstava. Korisnici mogu odabrati različite jedinice, svaku s različitim bojom, te crtati oblike te boje na karti. Karta se inicijalizira s podacima iz *mapData* polja naredbe i lokacijama jedinica ako takvi postoje. Karta i nacrtani oblici spremaju se, pri promjenama, u bazu podataka, osiguravajući postojanost podataka kroz sesije.

Primjer skice na interaktivnoj karti je prikazan na slici 4-6, plava, ljubičasta i narančasta točka su automatski prikazane trenutne lokacije jedinica dohvaćenih iz baze podataka, a sve ostalo je nacrtano.

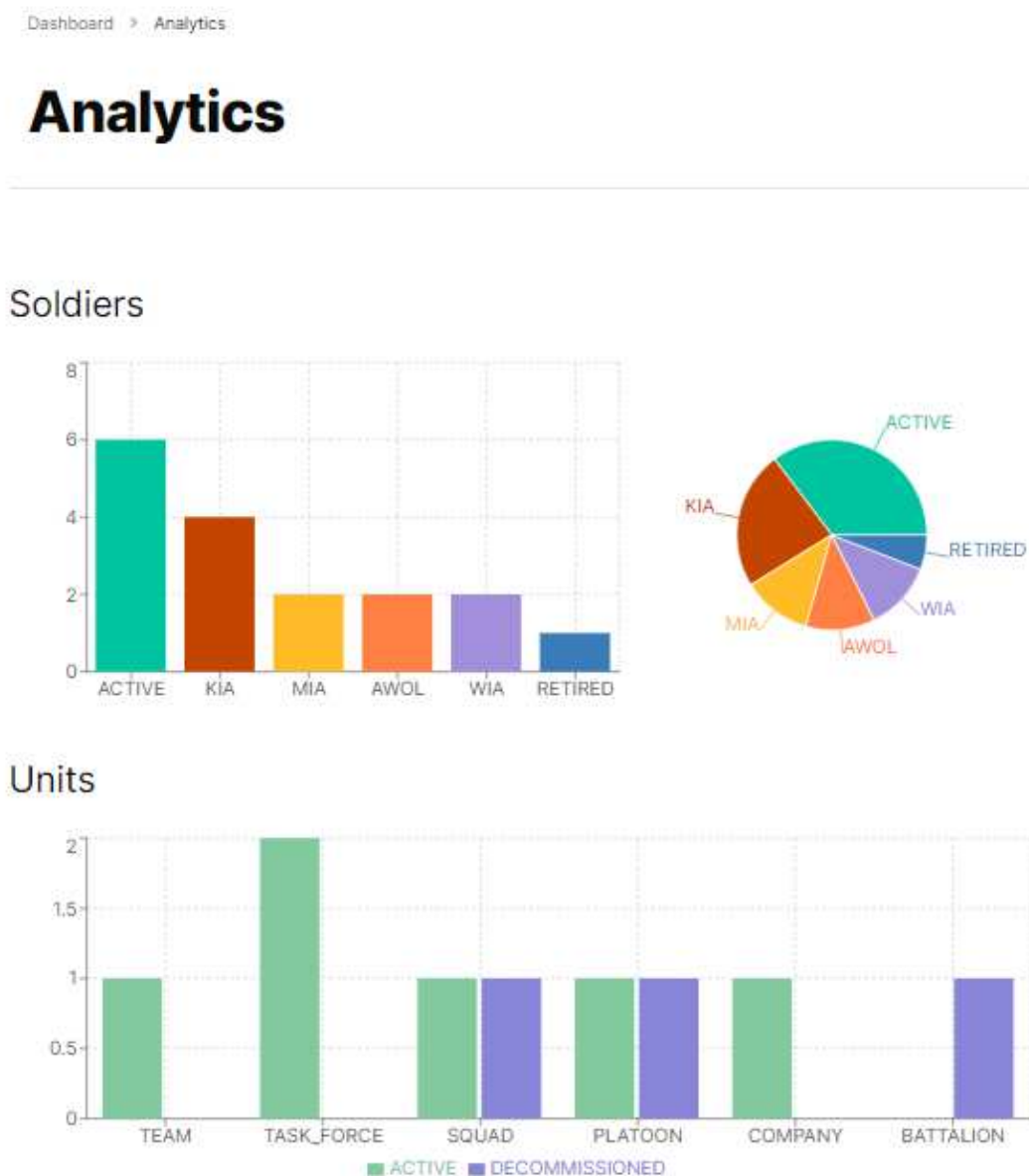


Slika 4- 6

4.1.7. Analitika

Analitika je postignuta s bibliotekom Recharts koja je popularna biblioteka za grafikone napravljena posebno za React, iskorištavajući snagu D3.js ispod haube za stvaranje jednostavnih, prilagodljivih i responzivnih grafikona. Korisnik u /analytics stranici ima mogućnost pregleda grafova statusa svih entiteta sustava.

Slika 4-7 prikazuje stupićasti i kružni graf statusa vojnika te stupićasti graf statusa jedinica grupiranih u tipove.



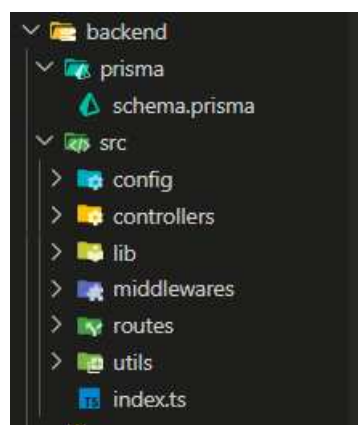
Slika 4- 7

4.2. Implementacija poslužiteljske aplikacije

U ovom poglavlju opisujemo implementacijski dio projekta u kontekstu aplikacije za sučelje.

Struktura implementacije poslužiteljske aplikacije je prikazana na slici 4-8, a osnovni direktoriji i datoteke su ukratko objašnjeni u listi ispod:

- prisma sadrži konfiguracije za Prisma ORM koji koristi bazu podataka
 - schema.prisma definira modele baze podataka i konfiguraciju veze
- src sadrži sav izvorni kod aplikacije
 - controllers sadrži logiku za rukovanje zahtjevima i odgovorima
 - routes sadrži definicije ruta za različite dijelove aplikacije, a index.ts je glavna ruta aplikacije
 - middlewares sadrži middleware funkcije koje procesiraju zahtjeve prije nego što stignu do kontrolera
 - config sadrži konfiguracijske datoteke
 - lib služi za pohranu zajedničkih modula, funkcija i instanci koje se koriste u cijeloj aplikaciji
 - utils sadrži pomoćne funkcije i alate
 - index.ts je ulazna točka aplikacije koja postavlja Express server, primjenjuje middleware, definira rute i pokreće server



Slika 4- 8

4.2.1. Usmjerivači i kontroleri

U Expressu usmjerivači (*routers*) i kontroleri (*controllers*) rade zajedno kako bi obradili HTTP zahtjeve i definirali logiku aplikacije. Usmjerivači se koriste za definiranje ruta, koje mapiraju URL staze do određenih rukovatelja zahtjevima. Oni organiziraju rute primjene u modularne segmente kojima se može upravljati. Kontroleri, s druge strane, sadrže poslovnu logiku za te rute. Kada se uspoređuje ruta, poziva se odgovarajuća funkcija kontrolera za obradu zahtjeva, interakciju s bazom podataka ili drugim uslugama i vraćanje odgovora. Ovo odvajanje zaduženosti pomaže u održavanju koda organiziranim, što olakšava upravljanje i održavanje sustava.

4.2.2. Enkripcija i dekrepcija podataka

Enkripcija poslužiteljske aplikacije koristi istu tehnologiju i način enkripcije kao aplikacija za sučelje, no ta enkripcija se odvija u middleware funkcijama, gdje je `decryptRequestMiddleware` funkcija zadužena za dekrepciju podataka u zahtjevu, `encryptResponseMiddleware` je zadužen za enkripciju podataka u odgovoru prije njegova slanja, a `authCheckMiddleware` za provjeru i dekrepciju autorizacijskog tokena u zaglavlju zahtjeva.

5. Upute za lokalno pokretanje

5.1. Potrebni programi i preduvjeti

Potrebno je skinuti i instalirati Node.js verziju v20.14.0 s Interneta te, skupa s tim, npm (Node Package Manager) verziju 10.7.0.

Korištenjem IDE-a po odabiru, otvoriti direktorij frontend ili backend na kojima se nalaze aplikacija za sučelje te poslužiteljska aplikacija respektivno. Te aplikacije se po idealnom korištenju pokreću na odvojenim računalima no u svrhu uputa se obje mogu pokrenuti na istom računalu.

Komandnom linijom potrebno je navigirati do direktorija frontend i backend te, zasebno za obje, skinuti potrebne biblioteke za određenu aplikaciju koristeći komandu „npm“.

Potrebno je kreirati i postaviti .env datoteku u *root* direktoriju obje aplikacije, zatim u obje dodati tajni ključ za enkripciju podataka (JWT_SECRET) te tajni ključ za enkripciju sjednice (JWT_SECRET_AUTH). Također je potrebno na aplikaciji za sučelje postaviti URL poslužiteljske aplikacije (BACKEND_URL), nasumično odabrani tajni ključ za svrhe autentifikacije i autorizacije sesija (NEXTAUTH_SECRET) te URL same aplikacije za sučelje (NEXTAUTH_URL). U .env datoteku na poslužiteljskoj aplikaciji je još dodatno postaviti URL aplikacije za sučelje (ORIGIN) te URL baze podataka (DATABASE_URL) o kojoj ćemo pričati više kasnije.

5.2. Konfiguracija baze podataka

Potrebno je skinuti i instalirati PostgreSQL i pgAdmin 4 zadužen za bazu podataka (preferabilno na računalu s poslužiteljskom aplikacijom), napraviti novu bazu podataka te konfigurirati URL za bazu podataka te ju postaviti u .env poslužiteljske aplikacije. Oblik tog URL-a je postgres://[KORISNIK]:[LOZINKA]@[DOMENA ili IP-ADRESA]:[PORT, *default* je 5432] /[IME BAZE PODATAKA], ukoliko joj se pristupa lokalno, domena je localhost.

U komandnoj liniji zaduženoj za poslužiteljsku aplikaciju je potrebno upisati komande „npx prisma db push“ za postavljanje modela baze podataka u bazu podataka te „npx prisma generate“ za sinkroniziranje strukture baze podataka s ostatkom koda. Kao provjeru

dostupnosti i provjeru trenutno dostupnih podataka u bazi podataka, uvijek možete upisati komandu „`npx prisma studio`“ koja pokreće zasebno web sučelje za pregled baze podataka ili ju možete direktno provjeriti u pgAdmin aplikaciji.

5.3. Pokretanje aplikacija

Nakon potrebnih i željenih daljnjih konfiguracija aplikacija i baze podataka, navigiranjem do direktorija aplikacija koristeći odvojene komandne linije, potrebno je pokrenuti aplikacije individualno koristeći komandu „`npm run dev`“ u *development* okruženju, ili „`npm run start`“ u *production* okruženju.

Ako ovim aplikacijama nisu pridružene domene, URL-ovi će biti <http://localhost:3000> za aplikaciju za sučelje te <http://localhost:4000> za poslužiteljsku aplikaciju.

Zaključak

Cilj ovog rada bio je razvoj vojnog sustava za upravljanje sadržajem (CMS) koji omogućava učinkovito upravljanje vojnim jedinicama, osobljem, opremom i naredbama. Sustav se oslanja na moderne tehnologije kao što su Next.js i React za frontend te Node.js s Expressom i Prismom za backend, uz PostgreSQL bazu podataka. Implementacija uključuje interaktivnu kartu za strateško planiranje i osnovne analitike resursa.

Razvijeni CMS uspješno pruža funkcionalnosti koje zadovoljavaju postavljene zahtjeve, uključujući sigurnu autentifikaciju i autorizaciju korisnika, raspolaganje resursima, osobljem i vojnim jedinicama, jednostavno korištenje interaktivne karte, enkripciju podataka te mogućnost jednostavne analize resursa.

Sustav je uspješno implementiran i testiran, ali postoje daljnje mogućnosti za unaprjeđenje. Na primjer, uz dodatnu sigurnost i MFA (*Multi-factor authentication*), aplikacija za sučelje bi mogla biti pokrenuta na demilitariziranoj zoni vlasnika sustava, što bi omogućavalo korisnicima korištenje aplikacije za sučelje putem Interneta.

Cijeli izvorni kod rada dostupan je u prilogu završnog rada. U budućnosti bih volio dodati više funkcionalnosti koje će sustav učiniti još korisnijim i prilagodljivijim potrebama korisnika.

Literatura

[1] PostgreSQL (2024)

Poveznica: <https://www.postgresql.org/download/>

Pristupljeno: 1.6.2024.

[2] Node.js (2024)

Poveznica: <https://nodejs.org/en>

Pristupljeno: 1.6.2024.

[3] Informacije o vojnoj strukturi jedinica 1 (2024)

Poveznica: <https://www.thirteen.org/blog-post/u-s-army-units-explained-from-squads-to-brigades-to-corps/>

Pristupljeno: 1.6.2024.

[4] Informacije o vojnoj strukturi jedinica 2 (2024)

Poveznica: <https://www.defense.gov/Resources/Insignia/>

Pristupljeno: 1.6.2024.

[5] Postman (2024)

Poveznica: <https://www.postman.com/>

Pristupljeno: 3.6.2024.

[6] Typescript dokumentacija (2024)

Poveznica: <https://www.typescriptlang.org/docs/>

Pristupljeno: 2.6.2024.

[7] Prisma dokumentacija (2024)

Poveznica: <https://www.prisma.io/docs>

Pristupljeno: 1.6.2024.

[8] Next.js dokumentacija (2024)

Poveznica: <https://nextjs.org/docs>

Pristupljeno: 1.6.2024.

[9] OWASP Security cheat sheet (2024)

Poveznica:

https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

Pristupljeno: 4.6.2024.

[10] NextAuth.js (2024)

Poveznica: <https://next-auth.js.org/getting-started/introduction>

Pristupljeno: 4.6.2024.

[11] jsonwebtoken biblioteka (2024)

Poveznica: <https://www.npmjs.com/package/jsonwebtoken>

Pristupljeno: 5.6.2024.

[12] React dokumentacija (2024)

Poveznica: <https://react.dev/>

Pristupljeno: 5.6.2024.

[13] Axios biblioteka (2024)

Poveznica: <https://axios-http.com/docs/>

Pristupljeno: 6.6.2024.

[14] Diagrams.net za crtanje dijagrama (2024)

Poveznica: <https://app.diagrams.net/>

Pristupljeno: 7.6.2024.

[15] pgAdmin 4 (2024)

Poveznica: <https://www.pgadmin.org/download/pgadmin-4-windows/>

Pristupljeno: 2.6.2024.

[16] Leaflet biblioteka (2024)

Poveznica: <https://leafletjs.com/reference.html>

Pristupljeno: 5.6.2024.

[17] Recharts biblioteka (2024)

Poveznica: <https://recharts.org/en-US/>

Pristupljeno: 4.6.2024.

[18] IMUNES za skiciranje mreže (2024)

Poveznica: https://www.imunes.net/dl/imunes_user_guide.pdf

Pristupljeno: 6.6.2024.

[19] shadcn/ui biblioteka (2024)

Poveznica: <https://ui.shadcn.com/>

Pristupljeno: 3.6.2024.

Sažetak

Web-temeljen sustav za upravljanje osobljem i resursima vojske

Cilj ovog sustava je pružiti učinkovit, korisniku prilagođen sustav za upravljanje vojnim resursima i osobljem. Olakšava stvaranje, ažuriranje i organizaciju podataka koji se odnose na vojnike, jedinice, opremu i zapovijedi. Uključivanjem značajki poput interaktivne karte za strateško planiranje i osnovne analitike za upravljanje resursima, CMS ima za cilj pojednostaviti administrativne zadatke i poboljšati procese donošenja odluka. Izgrađen s modernim tehnologijama, sustav osigurava skalabilnost i sigurnost, što ga čini robusnim alatom za vojnu administraciju.

U prvom poglavlju prolazimo kroz korisničke i funkcionalne zahtjeve ovog sustava, u drugom prolazimo kroz modeliranje i konceptualizaciju ideje sustava, u trećem opisujemo korištene tehnologije, u četvrtom opisujemo implementaciju sustava te u petom opisujemo lokalno pokretanje i konfiguraciju sustava za osobno korištenje.

Ključne riječi: web aplikacija, CMS, typescript, vojska, karta, sigurnost, postgresql

Summary

A web-based content management system for army personnel- and resource-management

The goal of this system is to provide an efficient, user-friendly system for managing military resources and personnel. Makes it easy to create, update, and organize data related to soldiers, units, equipment, and commands. By including features such as an interactive map for strategic planning and basic analytics for resource management, the CMS aims to simplify administrative tasks and improve decision-making processes. Built with modern technologies, the system ensures scalability and security, making it a robust tool for military administration.

In the first chapter we go through the user and functional requirements of this system, in the second we go through the modeling and conceptualization of the system idea, in the third we describe the technologies used, in the fourth we describe the implementation of the system and in the fifth we describe the local startup and configuration of the system for personal use.

Key words: web application, CMS, typescript, military, map, security, postgresql