Sustav za praćenje

Dominović, Mislav

Master's thesis / Diplomski rad

2018

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:096931

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-03-05**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni studij

SUSTAV ZA PRAĆENJE

Diplomski rad

Mislav Dominović

Osijek, 2018.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac D1: Obrazac za imenovanje Povjerenstva za obranu diplomskog rada

Osijek, 24.09.2018.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za obranu diplomskog rada

Ime i prezime studenta:	Mislav Dominović
Studij, smjer:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D 872 R, 27.09.2017.
OIB studenta:	84887107550
Mentor:	Doc.dr.sc. Ivan Aleksi
Sumentor:	
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	Izv. prof. dr. sc. Krešimir Nenadić
Član Povjerenstva:	Doc.dr.sc. Tomislav Keser
Naslov diplomskog rada:	Sustav za praćenje
Znanstvena grana rada:	**Automatika (zn. polje temeljne tehničke znanosti)**
Zadatak diplomskog rada:	(rezervirao Mislav Dominović) Sustav za praćenje putem internet aplikacije. Sastoji se od Arduina s GSM i GPS modulom koji šalje podatke o lokaciji korisnika na internet aplikaciju i prikazuje na karti gdje se osoba nalazi. Ukoliko se osoba udalji 300 metara od zadane lokacije internet aplikacija šalje poruku/poziva jednog od 3 zadana kontakta na internet aplikaciji. Mentor u tvrtki UHP Digital: Matej Dragun, mag. ing. comp.
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene mentora:	24.09.2018.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 25.09.2018.

Ime i prezime studenta:	Mislav Dominović
Studij:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D 872 R, 27.09.2017.
Ephorus podudaranje [%]:	3

Ovom izjavom izjavljujem da je rad pod nazivom**: Sustav za praćenje**

izrađen pod vodstvom mentora Doc.dr.sc. Ivan Aleksi

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD

Praćenje lokacije osobe ili nekog predmeta je u današnje doba sve izraženije i potrebnije, pa s time raste i želja za stalnim znanjem o njihovoj lokaciji. Čovjek se svakodnevno susreće sa strahom od gubitka dragih predmeta ili vremena u potrazi za kućnim ljubimcem ili osobama s određenim zdravstvenim i psihičkim problemima (npr. s demencijom). Ovakav sustav pomaže pri sastajanju s navedenim problemima, ali vjerojatno i s još mnogima. Dio sustava koji je zadužen za alarmiranje skrbnika ili vlasnika predmeta također omogućuje lakše opuštanje i ne razmišljanje hoće li netko ili nešto nestati, jer će osoba odmah dobiti obavijest o udaljavanju uređaja iz dopuštenih granica.

U drugom poglavlju se nalazi opis svih potrebnih tehnologija u izradi sustava. Treće poglavlje opisuje razvoj uređaja koji se bazira na prikupljanju podataka o lokaciji i spremanju podataka u bazu, te razvoj internet aplikacije koja služi za pristup podatcima koje je uređaj spremio u bazu, prikazivanju trenutne lokacije na karti i obavještavanju korisnika u slučaju napuštanja zadanog područja. Na kraju poglavlja se nalazi primjer rada sustava. Četvrto poglavlje ukazuje na slabosti sustava i njihova moguća rješenja.

2. OPIS KORIŠTENIH TEHNOLOGIJA

Temeljne tehnologije koje su korištene za izradu ovog diplomskog rada su programski jezik PHP, Laravel programski okvir, Arduino Uno [1] i A7 Ai-Thinker GPS/GPRS/GSM modul [2]. Internet aplikacija je izrađena uz pomoć PHP programskog jezika i trenutno najpopularnijeg okvira za izradu internet aplikacija - Laravel. Uređaj se sastoji od Arduino Uno-a, A7 modula i baterije. Za prikazivanje trenutne lokacije uređaja korišteno je nekoliko usluga iz Google Maps servisa.

2.1. PHP

PHP: Hypertext Preprocessor je skriptni jezik na strani poslužitelja koji je dizajniran za razvoj internet stranica, ali se također koristi kao programski jezik opće namjene. Napravio ga je Rasmus Lerdorf 1994. godine. PHP-ovu referentnu implementaciju sada proizvodi *The PHP Group.* PHP je izvorno značilo osobna stranica (engl. *Personal Home Page),* no sada se radi o rekurzivnom akronimu *PHP: Hypertext Preprocessor*.

2.2. Laravel

Laravel je besplatan PHP programski okvir otvorenog koda koji je napravio Taylor Otwell i namijenjen je razvoju internet aplikacija koji koristi arhitektonski uzorak model-pogled-upravljač (engl. *MVC – Model-View-Controller*) i temelji se na Symfony programskom okviru. Neke od značajki Laravel-a su modularni sustav pakiranja s namjenskim upraviteljem paketima, različitim načinima za pristup relacijskim bazama podataka, uslužnim programima koji pomažu u primjeni i održavanju aplikacija te usmjeravanju prema većoj čitljivosti koda. Izvorni kod Laravel-a moguće je pronaći na GitHub-u [3] i licenciran je pod uvjetima MIT (engl. *Massachusetts Institute of Technology*) licence.

2.3. Arduino

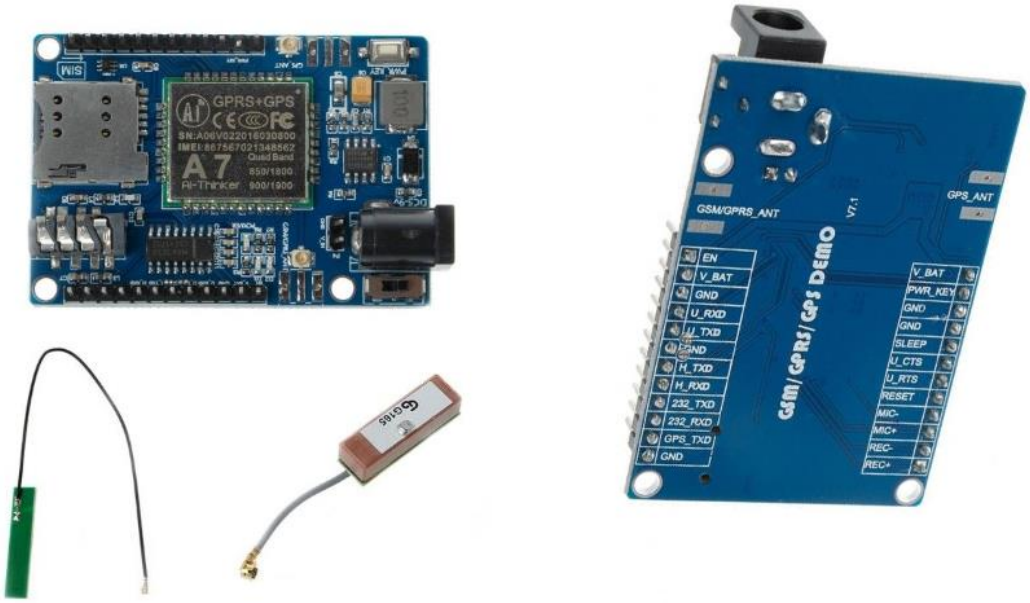
Arduino je kompanija koja proizvodi mikroprocesorske pločice koje je moguće programirati putem njihove programske podrške otvorenog koda koja je pristupačna svima na njihovoj stranici. Postoji više inačica Arduino pločica, a u ovom diplomskom radu se koristi Arduino Uno koji koristi ATmega328P mikroprocesor [4]. Kodovi za Arduino su pisani u prilagođenom C++ jeziku. Njegova glavna zadaća je povezivanje računalnog i fizičkog svijeta kroz konstruiranje radova, učenje programiranja i spajanja komponenti u cjelinu. Uz pomoć raznih senzora, motora i ostalih elektroničkih komponenti, Arduino može primati podražaje iz okoline i sam utjecati na okolinu.

Na samoj pločici se nalazi nekolicina ulaznih i izlaznih pinova. Za praćenje rada i komunikaciju Arduino-a koristi se *Serial Monitor*, na kojem je moguće prikazivati vrijednosti senzora, trenutno stanje programa ili nešto drugo bitno za praćenje tijeka i razvoja programa. U slučaju potrebe više *Serial Monitor-a* postoje biblioteke koje ga simuliraju, kao što je *Software Serial* [5] koji je korišten u ovom radu [6] . Na slici 2.1. se nalazi upravljačka jedinica Arduino Uno korištena u ovom diplomskom radu.

Slika 2.1. - Arduino Uno [1]

2.4. A7 Ai-Thinker

A7 čip je proizveden od strane Ai-Thinker tvrtke koja iza sebe ima nekoliko sličnih proizvedenih čipova. Sastoji se od GSM (engl. *Global System for Mobile communication*), GPRS (engl. *General Packet Radio Service*) i GPS (engl. *Global Positioning System*) modula. GSM je digitalni sustav mobilne telefonije koji se koristi u cijelom svijetu [7]. GPRS je standard bežične komunikacije dostupan korisnicima druge i treće generacije mobilnih uređaja koji koriste sustav GSM. Brzinom i pouzdanošću omogućuje jači razvoj korisničkih aplikacija raznovrsnog sadržaja [8]. GPS je detaljnije opisan u sljedećem potpoglavlju. Zbog male cijene, tj. jeftine proizvodnje, modul nije poslan s dokumentacijom. Dokumentacija modula [9] i popis AT komandi [10] su jako sažeti bez detaljnog objašnjenja. Na slici 2.2. se nalazi modul korišten u ovom diplomskom radu.



Slika 2.2. *A7 Ai-Thinker GSM/GPRS/GPS modul sa antenama [2]*

2.5. GPS

Globalni sustav pozicioniranja (engl. *GPS – Global Positioning System*) je uslužni program u vlasništvu SAD-a koji pruža korisnicima usluge pozicioniranja, navigacije i vremena (eng. *PNT – Position, navigation, time*). Ovaj se sustav sastoji od tri segmenta: segmenta prostora, kontrolnog segmenta i segmenta korisnika. [11]

2.5. NMEA rečenice

NMEA (engl. *National Marine Electronics Association)* je standardni format podataka za komunikaciju GPS uređaja. Podatci su formirani u rečenice čija je prva riječ vrsta podataka i definira tumačenje ostatka rečenice. Svaka vrsta podataka ima svoje jedinstveno tumačenje i definirana je u NMEA standardu. Ostale rečenice mogu ponavljati neke iste podatke, ali će također dati nove podatke. Bilo koji uređaj ili program koji čita podatke može uzimati u obzir podatkovnom rečenice za koje je zainteresiran i jednostavno ignorirati ostale rečenice. U NMEA standardu nema naredbi koji bi ukazivale na to da bi GPS trebao učiniti nešto drugačije. Svaki prijemnik šalje sve podatke i očekuje da će mnogo toga biti zanemareno. Neki prijemnici imaju naredbe unutar jedinice koje mogu odabrati podskup svih rečenica ili u nekim slučajevima pojedine rečenice za slanje. Nema načina da se javi pošiljatelju ili potvrdi da li je rečenica ispravno

primljena ili da se zatraži ponovno slanje nekih podataka koji nisu dobiveni. Jedinica za primanje sama provjerava kontrolni zbroj i zanemaruje podatke ako je kontrolni zbroj loš i očekuje da će se podatci ponovno poslati kasnije. [12]

2.6. Tehnologije za upravljanje izgledom internet aplikacije

Temeljne tehnologije za upravljanje izgledom internet stranica i aplikacija su HTML (engl. *Hypertext Markup Language*), CSS (engl. *Cascading Style Sheets*) i JavaScript programski jezik. HTML je označiteljski jezik koji opisuje internet stranicu, čiji elementi grade stranicu i sastoje se od oznaka otvaranja i zatvaranja koji označavaju početak i kraj elementa. Elementi se ne prikazuju unutar preglednika, nego preglednik vraća samo njihov sadržaj. CSS opisuje kako bi HTML elementi trebali biti prikazani unutar preglednika.

JavaScript je visoko tumačeni programski jezik koji je karakteriziran kao dinamičan, prototipan i više paradigmski. Uz tehnologije za upravljanje dizajnom internet stranica kao što su HTML i CSS, JavaScript je jedna od tri osnovne tehnologije *World Wide Web-a*. Omogućuje interaktivne internet stranice i stoga je bitan dio internet aplikacija. Većina internet stranica ga koriste, te svi glavni preglednici imaju podršku JavaScript programskog jezika. Kao više paradigmski jezik, JavaScript podržava stilove programiranja temeljene na događajima, funkcijama i imperativnim stilovima programiranja.

Unutar Laravel-a se koristi jQuery JavaScript biblioteka. To je brza, mala i bogata JavaScript biblioteka, koja čini stvari poput prijenosa HTML-a i manipulacije dokumentima, rukovanja događajima, animacije i AJAX-a (engl. *Asynchronous JavaScript and XML*) mnogo jednostavnijim pomoću jednostavne usluge koja funkcionira u mnoštvu preglednika. S kombinacijom svestranosti i proširivosti, jQuery je promijenio način na koji milijuni ljudi pišu JavaScript. [13]

2.7. Google Maps

Google Maps je Google-ova usluga digitalnih karata. Prikazuje satelitske snimke, karte ulica, panoramski pogled ulice (engl. *Street View*), prometne uvjete u stvarnom vremenu i planiranje ruta te se sastoji još od nekolicine usluga. Uz pomoć njihovih JavaScript rješenja, Google-ove usluge se jednostavno implementiraju u aplikacije.

Pristup pogodnostima Google Maps-a omogućeno je putem njihovih API-ja (engl. *Application programming interface*) koji također omogućuju jednostavnu implementaciju na internetske

stranice i aplikacije. API je termin koji se koristi za uslugu kojoj predajemo određene podatke i očekujemo čiste podatke kao odgovor bez dodatnog prezentacijskog sučelja kao što su internetske stranice. Obično vraćaju podatke u JSON (engl. *JavaScript Object Notation)* zapisu. JSON je format zapisa za razmjenu podataka. Ljudima ga je lako čitati i pisati, a i strojevima analizirati i generirati. Izgrađen je na dvije strukture:

- skup parova imena i vrijednosti – realizirano u različitim jezicima kao objekt, zapis, struktura, rječnik, *hash* tablica, lista s ključevima i vrijednostima ili asocijativno polje podataka,
- poredana lista vrijednosti – realizirano kao polje, vektor, lista ili slijed. [14]

2.7.1. Geocoding API

Usluga koja omogućuje proces geokodiranja, a to predstavlja pretvaranja adresa (npr. "*1600 Amphitheatre Parkway, Mountain View, CA*") u zemljopisne koordinate (zemljopisne širine 37.423021 i dužine -122.083739), koje se mogu koristiti za postavljanje oznaka na karti ili položaj karte. [15]

2.7.2. Places API

Usluga koja vraća informacije o mjestima koristeći HTTP zahtjeve. Mjesta su definirana unutar ovog API-ja kao ustanove, geografske lokacije ili istaknute točke interesa. Omogućuje odabiru mjesta unutar formi uz dodatak automatskog dovršavanja imena upisanog mjesta. [16]

2.7.3. Distance Matrix API

Usluga koja pruža izračunavanje udaljenost između dvije točke. API vraća informacije na temelju rute između početnih i završnih točaka izračunatog pomoću Google Maps API-ja i sastoji se od podataka trajanja i vrijednosti udaljenosti za svaki par. [17]

2.8. Blade sustav predložaka

Blade je jednostavan, ali moćan sustav predložaka koji se isporučuje sa Laravel-om. Za razliku od drugih popularnih PHP sustava predložaka, *Blade* ne ograničava od korištenje običnog PHP koda u pogledima. Svi *Blade-ovi* pogledi se prevode u običan PHP kod i spremaju se dok ne budu izmijenjeni. [18]

2.9. Git

Git je besplatan i distribuirani sustav kontrole verzija otvorenog koda dizajniran za obradu malih do vrlo velikih projekata brzo i učinkovito. Alat za verzioniranje programskog koda radi lakšeg rada u timovima i otklanjanje grešaka. Koristi se granama (engl. *branch*) za lakše odvajanje od programskog koda glavnog dijela aplikacije zbog programiranja novih dijelova. [19]

2.10. Heroku

Heroku je „platforma u oblaku“ temeljena na upravljanom sustavu kontejnera s integriranim podatkovnim uslugama i snažnim ekosustavom za implementaciju i pokretanje modernih aplikacija. Heroku programeri iskušavaju pristup prilagođen za isporuku aplikacija, integriran s najpopularnijim alatima za razvoj i tijekove rada. [20]

2.11. HTTP protokol

HTTP (engl. *Hypertext Transfer Protocol*) je aplikacijski protokol za distribuirane, suradničke, hipermedijske informacijske sustave. To je generički protokol bez spremanja stanja koji se može koristiti za mnoge zadatke izvan upotrebe za hipertekst, kao što su nazivni poslužitelji i distribuirani sustavi za upravljanje objektima, kroz proširenje svojih metoda zahtjeva, kodova grešaka i zaglavlja. Značajka HTTP-a je tipizacija i pregovaranje o prikazu podataka, što omogućava izgradnju sustava neovisno o prenesenim podacima. HTTP je u upotrebi od 1990. [21]

HTTP protokol koristi nekoliko metoda zahtjeva za upravljanje resursima putem interneta. Tablica 2.1. prikazuje popis osnovnih metoda s objašnjenjima njihovih funkcija.

Tablica 2.1. *Osnovne metode zahtjeva HTTP protokola*

Ime metode zahtjeva	Opis
HEAD	Zahtjev za vraćanje zaglavlja dokumenta
GET	Zahtjev za vraćanje dokumenta
PUT	Zahtjev za spremanje dokumenta
POST	Zahtjev za pružanje podataka koji trebaju biti dodani u dokument
DELETE	Zahtjev za brisanje dokumenta

2.12. MySQL

MySQL je najpopularnija baza podataka otvorenog koda na svijetu. Bez obzira da li je za brzo rastuće platforme, tehnologije ili velika poduzeća, MySQL može ekonomično pomoći u isporuci skalabilnih aplikacija visokih performansi s bazom podataka. [22]

2.13. PostgreSQL

PostgreSQL je moćan, objektno-relacijski sustav baze podataka otvorenog koda s više od 30 godina aktivnog razvoja koji je stekao snažnu reputaciju za pouzdanost, robusnost i performanse. [23]

3. RAZVOJ VLASTITOG RJEŠENJA

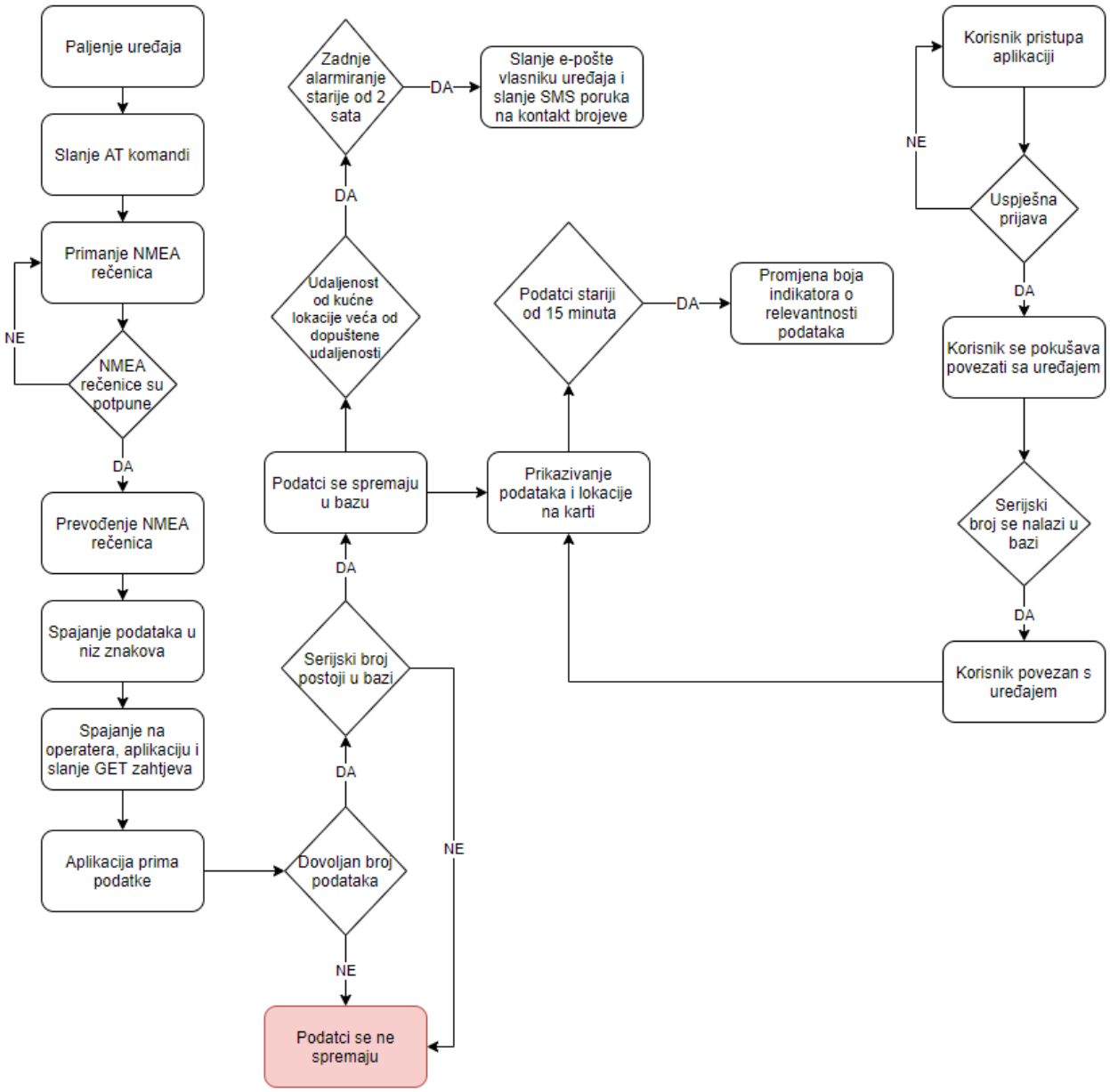
3.1. Opis sustava

Za sustav za praćenje je potreban pristup podatcima o zemljopisnom položaju uređaja, te prikazivanje trenutnog položaja uređaja uz dodatne podatke koji se dobivaju sa satelita uz koordinate. Za ovaj sustav odabrani su Arduino Uno i Ai-Thinker A7 modul i internet aplikacija s integriranim Google Maps API-jem. Arduino Uno je potrebno programirati za iskorištavanje potrebnih funkcionalnosti modula kako bi se dobile zemljopisna dužina i širina, nadmorska visina, brzina kretanja i broj satelita. Kada se dobiju svi potrebni podatci potrebno je ostvariti pristup internet aplikaciji za pohranu podataka u bazu. Ta ista baza podataka je povezana s internet aplikacijom napravljenom u Laravel programskom okviru za izradu internet aplikacija u PHP programskom jeziku. Unutar internet aplikacije prikazuje se lokacija uređaja na karti zajedno sa ostalim podatcima koji se dobivaju uz pomoć modula.

Razvoj rješenja kreće s osposobljavanjem A7 modula te isprobavanja potrebnih funkcionalnosti modula preko terminala ili *Serial Monitor-a*. Nakon testiranja funkcionalnosti modula nastupa implementacija potrebnih funkcija uređaja u prilagođenom C++ jeziku za programiranje Arduino Uno-a. A7 modul radi na principu primanja signala od satelita u obliku NMEA rečenica koje nakon toga Arduino Uno prevodi ako NMEA rečenice sadrže korisne podatke. GPS modulu je potrebna primanje signala s minimalno tri satelita kako bi mogao odrediti svoj položaj u dvodimenzionalnom prostoru, tj. zapravo mu treba d+1 satelita da odredi svoj položaj u d-dimenzionalnom prostoru. Zatim uređaj šalje podatke u bazu podataka preko internetske mreže pristupom internet aplikaciji koja sprema podatke u bazu. Internet aplikacija napravljena u Laravel-u je implementirana na Heroku servisu zajedno sa bazom podataka. Internet aplikacija se sastoji od prijave i registracije korisnika, povezivanja uređaja s korisničkim računom i dodavanjem brojeva telefona na koji će se obavijestiti ako uređaj napusti udaljenost od kućne lokacije uređaja koju je korisnik postavio. Spremanje lokacijskih podataka uređaja u bazu i povezivanje s računom je riješeno spremanjem serijskog broja A7 modula (svaki modul ima drugačiji broj) zajedno s podatcima koje dobije sa satelita, te kada korisnik upiše serijski broj modula povezuje se s uređajem i ima pristup podatcima o lokaciji uređaja koje je njegov modul upisao u bazu.

3.2. Dijagram toka sustava

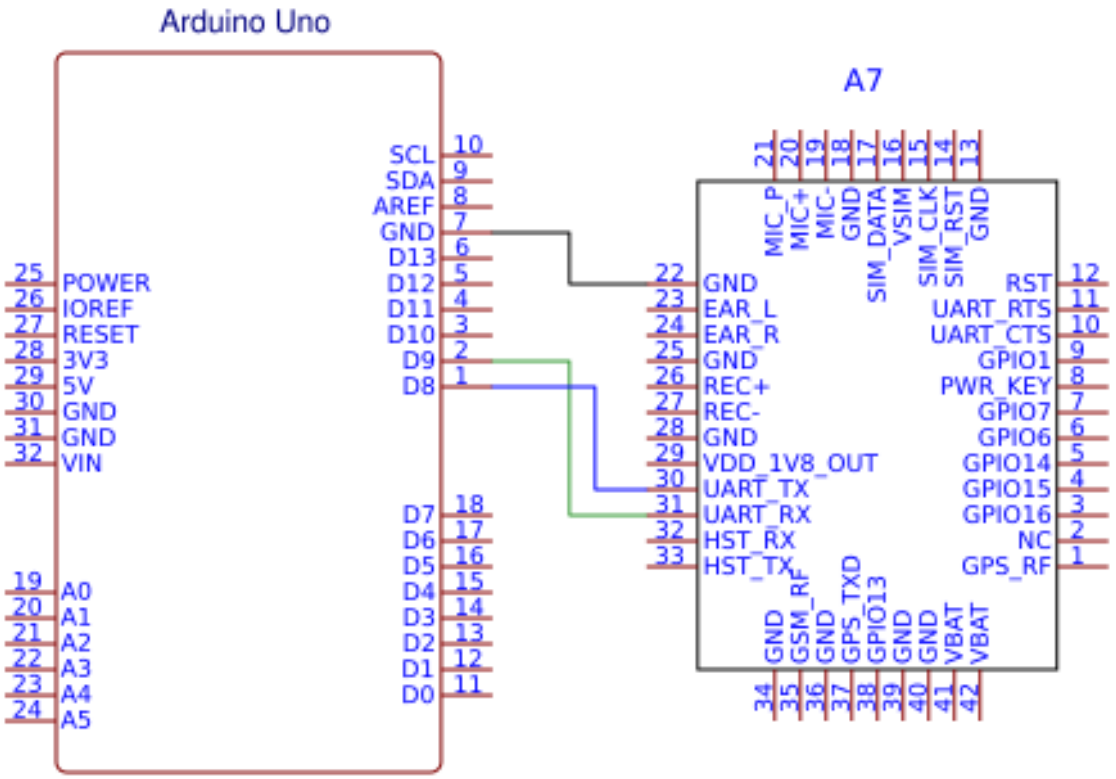
Na slici 3.1. se nalazi dijagram toka sustava, tj. prikaz svih bitnih aktivnosti kroz koje prolazi sustav od paljenja uređaja do prikazivanja lokacije i obavještavanja korisnika. Nakon paljenja uređaja šalju se AT komande za pokretanje primanja NMEA rečenica. Njihova valjanost se provjerava i ukoliko se mogu prevesti u korisne podatke sustav nastavlja s radom i podatke spaja u niz znakova koje šalje u bazu pristupom internet aplikaciji putem HTTP protokola s GET zahtjevom. Tu završava posao upravljačke jedinice, tj. uređaja. Nakon toga aplikacija prima podatke i provjerava da li je dovoljan broj podataka poslan i da li uređaj sa serijskim brojem predanim uz podatke o lokaciji postoji u bazi podataka. Ako prođu te dvije provjere podatci se spremaju u bazu. Zatim se provjerava udaljenost lokacije od kućne lokacije uređaja. Ako je izvan dopuštene udaljenosti, korisnik se obavještava slanjem e-pošte i SMS poruke na brojeve navedene kao kontakti. Pošto su podatci spremljeni u bazi, moguće ih je prikazati na karti i tablici pored karte. Osim dijela komunikacije aplikacije s uređajem, također postoji i drugi dio tijeka aplikacije koji kreće s registracijom korisnika i prijavom u sustav. Tada korisnik prolazi kroz klasične provjere postojanja računa u bazi te usporedbom šifrirane lozinke sa zapisom u bazi. Nakon toga se korisnik spaja na uređaj sa serijskim brojem, te prolazi provjeru da li postoji uređaj s tim serijskim brojem. Ako postoji korisnik je povezan i može pristupati podatcima o lokaciji uređaja.



Slika 3.1. *Dijagram toka sustava*

3.3. Arduino

Arduino Uno i A7 modul su vrlo jednostavno povezani i osposobljeni jer A7 modul se sastoji od tri modula od kojih su za ovaj rad potrebno dva, te manji broj fizičkih modula smanjuje kompleksnost povezivanja s mikroprocesorom. Za iskorištenje svih funkcionalnosti modula potrebno je umetnuti SIM karticu u modul s raspoloživim iznosom domaće valute ili jedinica za pristup internetu. Na slici 3.2. se nalazi shema spoja Arduino Uno-a i A7 modula.



Slika 3.2. *Shema spoja*

Modul radi na principu otvaranja GPS komunikacije, povezivanju sa satelitima, te primanje NMEA rečenica koje je potrebno prevesti u vrijednosti razumljive čovjeku i Google Maps API-u. To je omogućeno s *TinyGPSPlus* bibliotekom [24]. Za očitavanje zemljopisnih koordinata lokacije uređaja potrebno je minimalno tri satelita. Valjanost NMEA rečenica se može provjeriti s besplatnim *Free* NMEA prevoditeljem koji prikazuje NMEA rečenice u obliku čitljivom čovjeku [25] .

U nastavku se nalaze dva primjera NMEA rečenica, te na slikama 3.3. i 3.4. njihov prijevod uz pomoć *Free* NMEA servisa.

```
$GPGGA,134819.000,4533.26135,N,01841.15690,E,1,05,3.9,134.5,M,,M,,0000*4D
$GPRMC,134819.000,A,4533.26135,N,01841.15690,E,0.00,0.00,170918,,,A*6B
$GPVTG,0.00,T,,M,0.00,N,0.00,K,A*3D
```

Type	UTC Time	Position	Course	Speed, kn/kph	Altitude	HDOP,VDOP,PDOP	Satellites	CRC OK?
GGA	2018-09-17T13:48:19Z	45°33'15.68"N, 18°41'9.41"E			134.5	3.9 - -	5	CRC OK
RMC	2018-09-17T13:48:19Z	45°33'15.68"N, 18°41'9.41"E				- -		CRC OK
VTG						- -		CRC OK
Empty line skipped								

Slika 3.3. *Prijevod NMEA rečenice uz pomoć Free NMEA prevoditelja*

```
$GPGGA,134833.000,4533.26120,N,01841.15544,E,1,05,4.0,136.3,M,,M,,0000*41
$GPGSA,A,3,01,03,17,22,23,,,,,,,,5.8,4.0,4.3*39
$GPGSV,3,1,10,01,85,153,28,03,59,283,32,11,57,183,20,17,29,307,28*75
$GPGSV,3,2,10,22,79,336,32,23,28,208,23,32,17,045,,09,01,215,*78
$GPGSV,3,3,10,31,25,092,,08,07,183,*73
$GPRMC,134833.000,A,4533.26120,N,01841.15544,E,0.00,0.00,170918,,,A*6D
$GPVTG,0.00,T,,M,0.00,N,0.00,K,A*3D
```

Type	UTC Time	Position	Course	Speed, kn/kph	Altitude	HDOP,VDOP,PDOP	Satellites	CRC OK?
GGA	2018-09-17T13:48:33Z	45°33'15.67"N, 18°41'9.33"E			136.3	4 - -	5	CRC OK
GSA						4 - 4.3 - 5.8	5	CRC OK
GSV						- -	10	CRC OK
GSV						- -	10	CRC OK
GSV						- -	10	CRC OK
RMC	2018-09-17T13:48:33Z	45°33'15.67"N, 18°41'9.33"E				- -		CRC OK
VTG						- -		CRC OK
Empty line skipped								

Slika 3.4. *Prijevod NMEA rečenice uz pomoć Free NMEA prevoditelja*

Uz pomoć GET zahtjeva modul pristupa internet aplikaciji koji iz krajnje točke uzima niz podataka i rastavlja ih na dijelove koji su odijeljeni znakom „*“ te sprema u bazu podataka. Sastoji se od zemljopisnih koordinata, broja satelita, brzine kretanja, nadmorske visine i serijskog broja modula. Zbog manjka mogućnosti slanja POST zahtjeva s A7 modulom, GET zahtjev ostaje kao jedina opcija predavanja podataka u bazu. Sve operacije koje A7 modul nudi se mogu pronaći u dokumentaciji [9] i popisu AT komandi [10] koje se šalju preko *Serial Monitor-a*. U nastavku se nalazi primjer krajnje točke za prijenos podataka s uređaja na aplikaciju, gdje se inače umjesto uglatih zagrada nalaze vrijednosti koje uređaj očitava.

```
https://calm-plateau-88601.herokuapp.com/api/location/new/[latitude]
*[longitude]*[speed]*[altitude]*[satellites]*[serial_number]
```

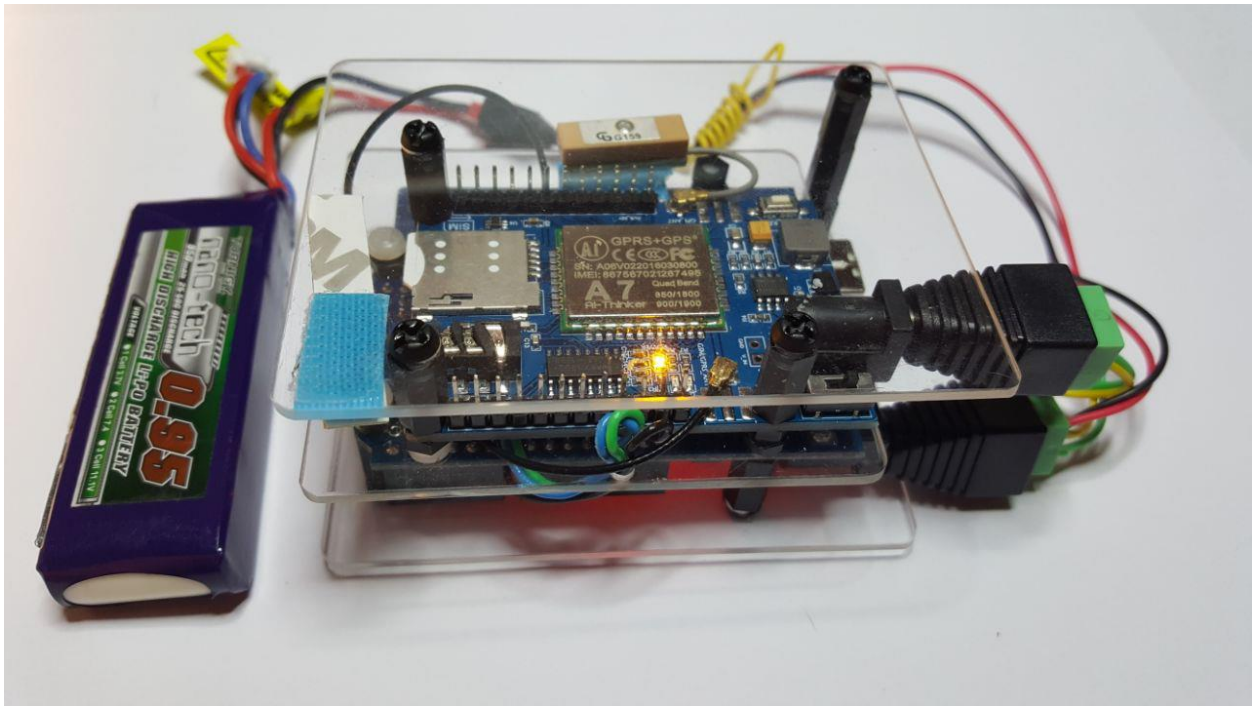
Proces spajanja modula na Arduino Uno počinje sa spajanjem UART_TX (na modulu označen kao U_TXD), UART_RX (na modulu označen kao U_RXD) i GND pinova na A7 modulu na TX, RX (digitalni pinovi 1 i 2) i GND pinove te zatim testiranje AT komandi uz pomoć SSH (engl. *Secure Shell)* klijenta ili uz pomoć *Serial Monitor-a*.

Nakon toga se A7 modul prespaja na druga dva digitalna pina po izboru zbog upotrebe *Software Serial-a* kako bi mogli slati potrebne komande, te na klasični *Serial Monitor* primati NMEA rečenice. Zato što su NMEA rečenice nečitljive u takvom obliku potrebno ih je prevesti. Ranije spomenuta *TinyGPSPlus* biblioteka omogućuje izvlačenje svih mogućih podataka iz potpunih NMEA rečenica. Često zbog slabog signala ili neočitavanja dovoljno satelita NMEA rečenice znaju biti polovične ili nedovršene. U nastavku slijedi primjer skupa rečenica iz koje se ne mogu dobiti nikakvi korisni podatci.

```
$GPGGA,,,,,,0,00,,,M,,M,,0000*66
$GPRMC,,V,,,,,,,,,,N*53
$GPVTG,,T,,M,,N,,K,N*2C
```

Programiranje Arduino Uno-a počinje s postavljanjem oba *Serial Monitor-a* na 9600 baud-a te slanjem AT komandi za pokretanje GPS modula – AT+GPS=1. Zatim se poziva komanda AT+GPSRD=1, što preusmjerava NMEA rečenice sa GPS_TXD pina na UART_TX pin. Zapravo čita NMEA rečenice svake sekunde. Kada se krenu čitati potpune rečenice (rečenice koje se mogu prevesti u korisne vrijednosti) i kada se promjeni barem jedna vrijednost u odnosu na skup prethodnih, podatci se ažuriraju i poziva se funkcija za učitavanje krajnje točke te slanje GET zahtjeva za upisivanje podataka u bazu. To se obavlja tako da se pošalje komanda AT+GPSRD=0, tj. zaustavi se čitanje lokacije, te se uz pomoć *TinyGSM* [26] biblioteke šalju AT komande za pristup internetskoj stranici. Modul se prvo spaja na operatera (engl. *APN - Access point network)*, zatim na poslužitelja (internet aplikaciju) i onda šalje GET zahtjev. Nakon toga se gasi internet veza i ponovno se šalje komanda za čitanje GPS podataka i program nastavlja s radom, tj. ponavlja se cijeli proces. Kod za programiranje Arduino Uno-a se nalazi u prilozima pod P.3.1.

Zbog napajanja iz utičnice, uređaj ima slabu prenosivost. Stoga je potrebno spojiti ga na bateriju i nekako učvrstiti Arduino Uno i A7 modul. Slika 3.4. prikazuje izvedbu uređaja u prenosivom obliku.



Slika 3.4. *Uređaj u prijenosnoj izvedbi*

Modul ispravno radi na napajanju između 5 i 9 V [9] koje se spaja na ulaz jednosmjerne struje na pločici, a Arduino Uno na istom ulazu ispravno radi na preporučenom naponu između 7 i 12 V [1]. Stoga je odabrana raspoloživa baterija od nominalne voltaže, kojoj je nominalna voltaža 7.4 V. Baterija je spojena improviziranim razdjelnikom za spajanje s dva nastavka potrebna za napajanje A7 modula i upravljačke jedinice Arduino Uno. Tehničke specifikacije baterije se nalaze u tablici 3.1.

Tablica 3.1. *Specifikacije baterije [27]*

Kapacitet	Napon	Pražnjenje	Težina	Dimenzije	Priključak za balansiranje	Priključak za pražnjenje
950 mAh	7.4 V	25C konstantno / 50C nalet	46 g	71×25×14 mm	JST-XH	mini-JST

Mjerenje je pokazalo da sustav pri normalnom radu troši struju jakosti 0.06 A, tj. 60 mA. Koristeći

matematički izraz naveden pod (3-1) dobiva se da se sustav može napajati približno 16 sati, koristeći ranije navedenu bateriju.

$$t = \frac{Q}{I} \tag{3-1}$$

gdje je:

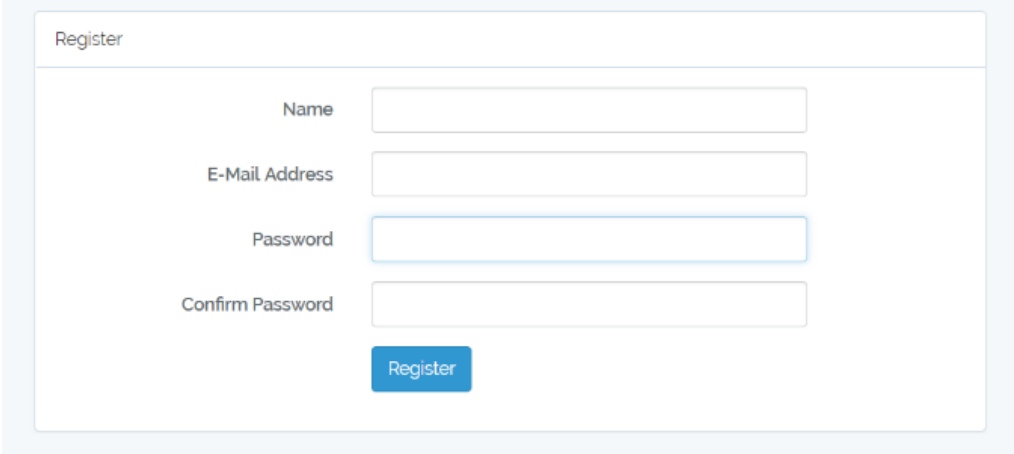
- t – vrijeme koje baterija može izdržati (u satima),
- Q – kapacitet baterije,
- I – jakost el. struje koja se crpi iz baterije.

3.4. Internet aplikacija

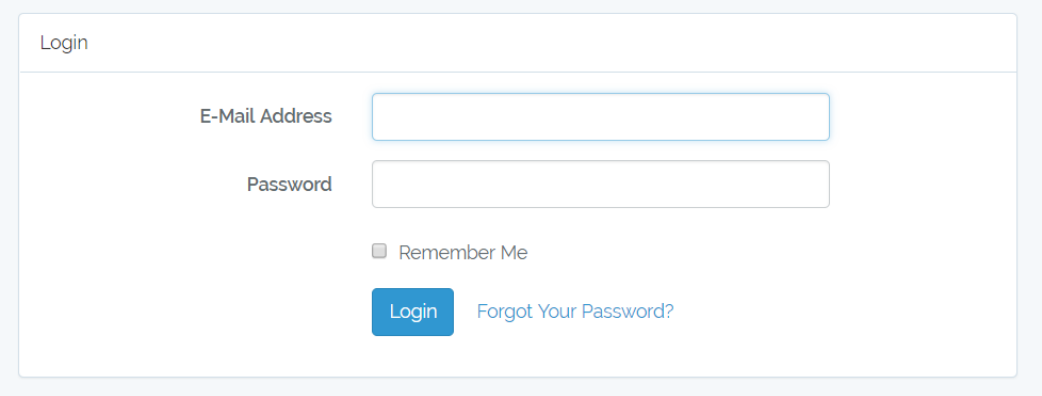
Aplikacija je napravljena uz pomoć Laravel programskog okvira u PHP programskom jeziku. Korisničko sučelje i izgled aplikacije je napravljen uz pomoć *Blade* sustava predložaka koji olakšava uređivanje izgleda korisničkog sučelja, koristeći HTML, CSS, JavaScript i jQuery. Za raspored elemenata na pogledima i promjenu izgleda aplikacije ovisno o veličini ekrana uređaja kojim se pristupa aplikaciji zadužen je *Bootstrap* [28] . Pri ulasku na internet aplikaciju postoji opcija za prijavu i registraciju. Na slici 3.5. se nalazi početni zaslon pri ulasku na aplikaciju.

Slika 3.5. *Početna stranica*

Sustav za prijavu i registraciju je napravljen koristeći komandu „*php artisan make:auth*“. To je automatiziran proces za stvaranje spomenutog sustava koji napravi sve potrebne resurse i odmah je spreman za korištenje. Neki od resursa koje napravi su: tablice u bazi podataka, pogledi potrebni za registraciju, prijavu i zaboravljenu lozinku, korisniče i ostale modele, upravljače, rute, međusloj koji štiti rute od pristupa neprijavljenih korisnika. Na slici 3.6. i 3.7. se nalaze obrasci za prijavu i registraciju.

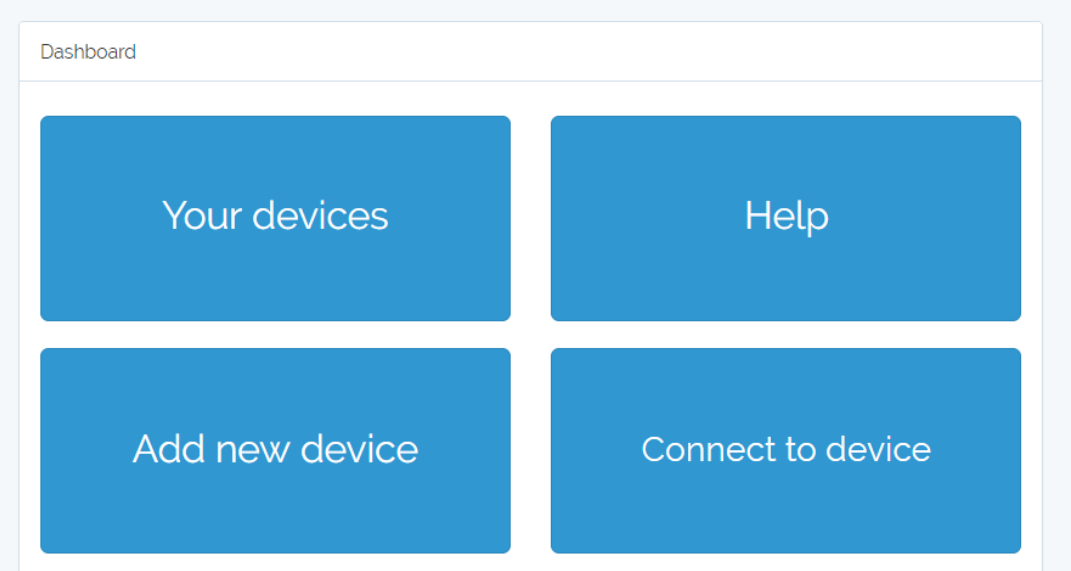


Slika 3.6. *Obrazac za registraciju*



Slika 3.7. *Obrazac za prijavu*

Nakon registracije i ulaska u novootvoreni račun, nalazi se izbornik koji se sastoji od tipke za dodavanje novog uređaja (engl. *add new device*), povezivanje sa već postojećim uređajima (engl. *connect to device*), prikazivanje svih dodanih uređaja (engl. *your devices*) i tipke za pomoć (engl. *help*) ako korisniku nešto nije jasno. Slika 3.8. prikazuje izgled izbornika.

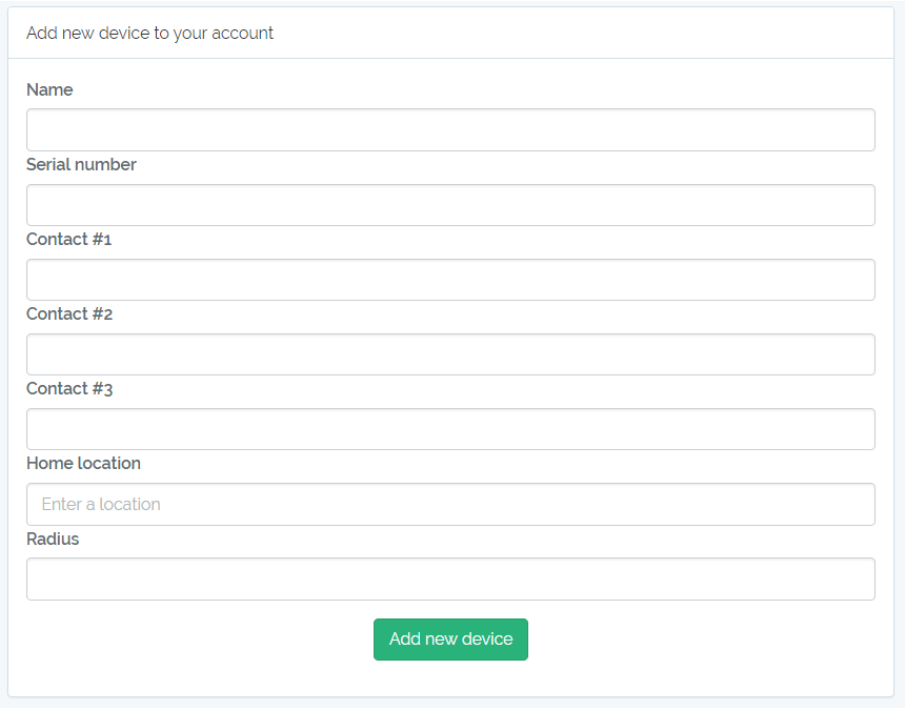


Slika 3.8. *Izbornik*

Uređaji koji se koriste su već dodani u bazu podataka i pri dodavanju novog uređaja se zapravo samo povezuje korisnik s uređajem, postavlja kao vlasnik i zadaju se osnovni podatci uređaja:

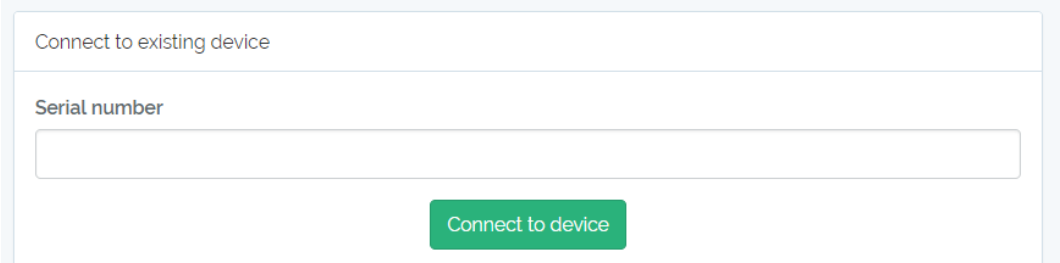
- ime uređaja
- serijski broj - provjerava se da li postoji uređaj s navedenim serijskim brojem u bazi i da li već ima vlasnika
- tri kontakt broja koja se obavještavaju SMS porukom ako uređaj napusti zadane granice - potrebno je navesti minimalno jedan broj
- kućna lokacija uređaja oko koje se zadaje udaljenost granica u kojima bi se uređaj trebao moći kretati bez alarmiranja korisnika
- udaljenost od kućne lokacije do granica kretanja uređaja.

Kućna lokacija uređaja se dodaje upisivanjem adrese ili mjesta koje korisnik želi postaviti. To omogućuje Google *Places* API sa automatskim dovršavanjem teksta koji za vrijeme upisivanja u polje nudi predložak o mjestu. Nakon odabira željenog mjesta, sprema ga u varijablu i nju šalje kao zahtjev *Geocoder* API-ju koji vraća koordinate zemljopisne duljine i širine kao odziv. Za jednostavnije slanje zahtjeva na API, korišten je Guzzle HTTP klijent [29]. Na slici 3.9. je prikazan izgled obrazaca za dodavanje novog uređaja, tj. povezivanja na uređaj kao vlasnik.



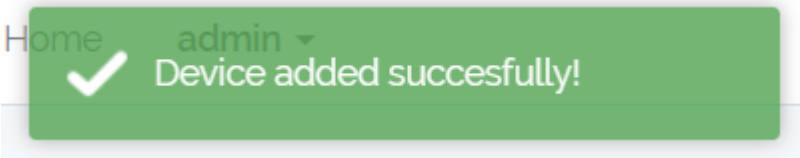
Slika 3.9. *Obrazac za dodavanje novog uređaja*

Slika 3.10. prikazuje obrazac za drugi način povezivanja s uređajem koji omogućuje samo gledanje podataka i lokacije. Sastoji se samo od unosa serijskog broja jer se smatra da korisnik koji će tako dodati uređaj nema namjeru biti vlasnik i mijenjati podatke uređaja.

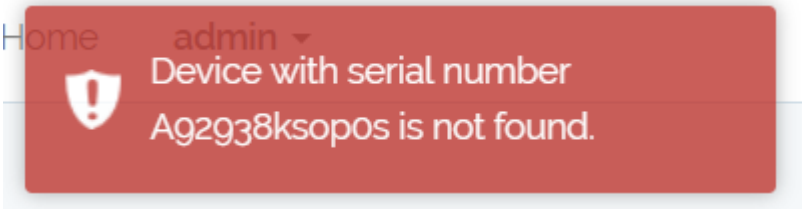


Slika 3.10. *Obrazac za povezivanje sa već postojećim uređajem*

Prilikom dodavanja novog uređaja ili rađenja bilo koje slične aktivnosti na aplikaciji, u desnom gornjem kutu se pojavljuje obavijest o uspješnom ili neuspješnom izvršavanju aktivnosti. Na slici 3.11. i 3.12. je primjer kako izgleda kada se novi uređaj poveže s korisničkim računom i kada se pokuša povezati s uređajem s krivim serijskim brojem. To omogućuje Toastr JavaScript biblioteka za ne blokirajuće obavijesti. [30]



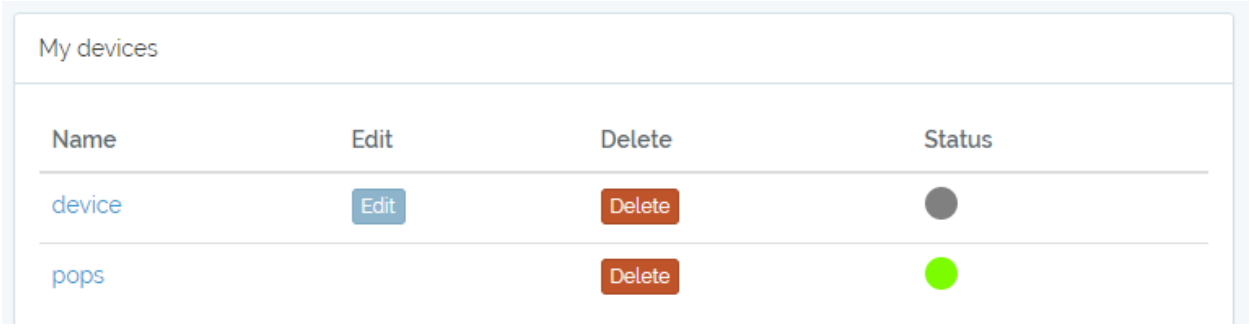
Slika 3.11. *Uspješno povezivanje sa uređajem*



Slika 3.12. *Dodavanje uređaja sa nepostojećim serijskim brojem*

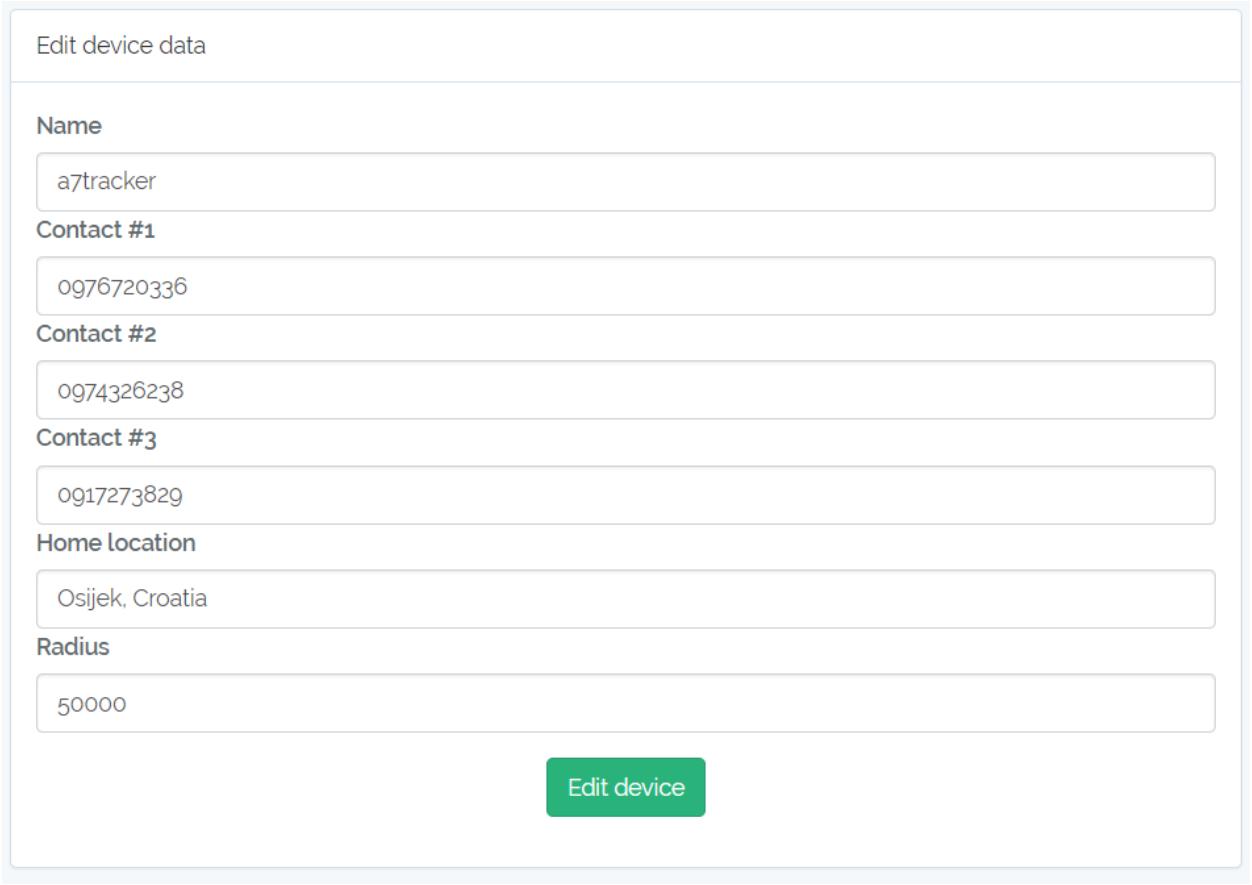
Serijski broj koji je potreban pri povezivanju sa uređajem se nalazi na A7 modulu zajedno sa IMEI-em (engl. *International Mobile Equipment Identity*).

Pritiskom tipke *Your devices* prikazuju se svi uređaji s kojima se korisnik povezao. Ako je korisnik vlasnik uređaja, ima opciju za izmjenu podataka (engl. *edit)* uređaja koje je upisivao pri dodavanju, tj. povezivanju s uređajem. Pored tipke za izmjenu se nalazi tipka za brisanje uređaja i indikator statusa uređaja. Na slici 3.13. se može primijetiti da je korisnik vlasnik prvog uređaja dok drugog nije (nema opciju za izmjenu).



Slika 3.13. *Prikaz uređaja povezanih s korisnikom*

Na slici 3.14. se nalazi obrazac za izmjenu podataka uređaja koji izgleda jako slično obrascu za dodavanje uređaja, samo bez polja za upis serijskog broja, dok su ostala polja ispunjena trenutnim podatcima iz baze.

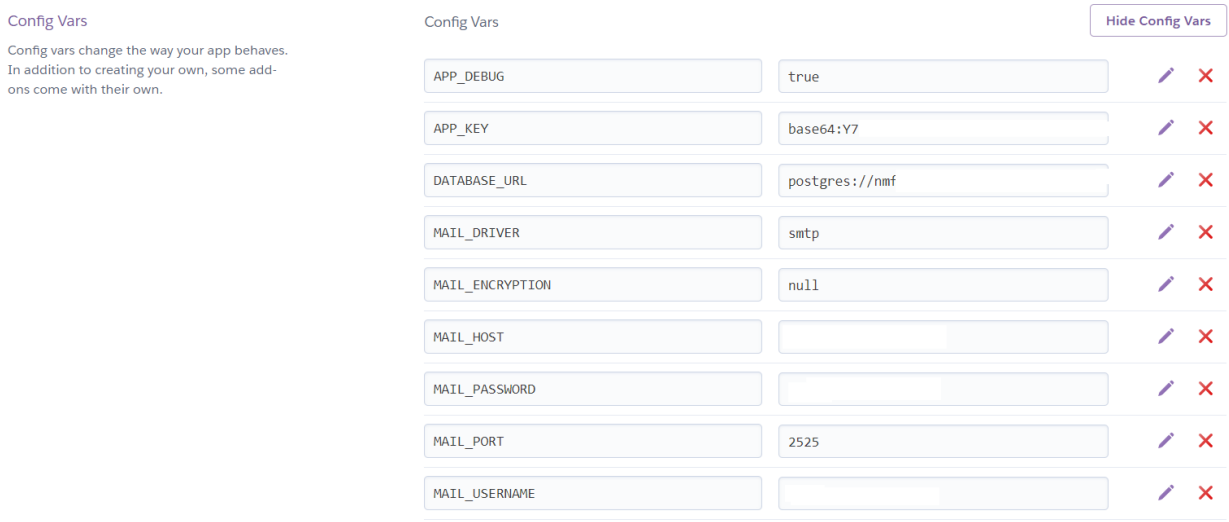


Slika 3.14. *Obrazac za izmjenu podataka o uređaju*

U glavnom izborniku se nalazi i tipka za pomoć (engl. *help)* koja pruža opis osnovnih funkcionalnosti sustava ako korisniku nešto nije jasno.

Uz pomoć Heroku CLI (engl. *Command Line Interface*) [31] i njihovih uputa, aplikacija je uspješno prenesena s lokalnog poslužitelja na Heroku servis, te je implementirana PostgreSQL baza podataka. Kada se napravi nova Heroku aplikacija stvori se novi udaljeni direktorij aplikacije i *git* direktorij, što znači da je za prikazivanje promjena na aplikaciji potrebno samo gurnuti promjene na Heroku *master* granu koja automatski promjeni sadržaj na aplikaciji. Zbog korištenja PostgreSQL baze podataka, izmjene se moraju napraviti zbog podataka za prijavu u bazu podataka na projektu jer je aplikacija razvijana u lokalnom okruženju s MySQL bazom. Heroku korisničko sučelje omogućuje jednostavno dodavanje varijabli koje su potrebne da bi aplikacija radila (podatci koji se inače nalaze u *.env* datoteci varijabli okruženja, a nisu preneseni putem *git-a* zato

što je datoteka navedena u *.gitignore* datoteci što sprječava prenošenje osjetljivih podataka); kao što je *APP_KEY* koji je potreban svakoj Laravel aplikaciji, krajnja točka koja sadrži podatke za obavljanje aktivnosti nad PostgreSQL bazom podataka koja se poziva u aplikaciji. Prikaz varijabli okruženja se nalazi na slici 3.15.



Slika 3.15. *Konfiguracijske varijable Heroku aplikacije*

Migracija baze podataka je učinjena pokretanjem Heroku terminala unutar direktorija aplikacije komandom „*heroku run bash*“ i pokretanjem komande za migraciju – „*php artisan migrate --seed*“. Navedena komanda za migriranje također pokreće *seeder* klase za popunjavanje baze podataka osnovnim podatcima za rad sustava, kao što su korisnik i uređaj.

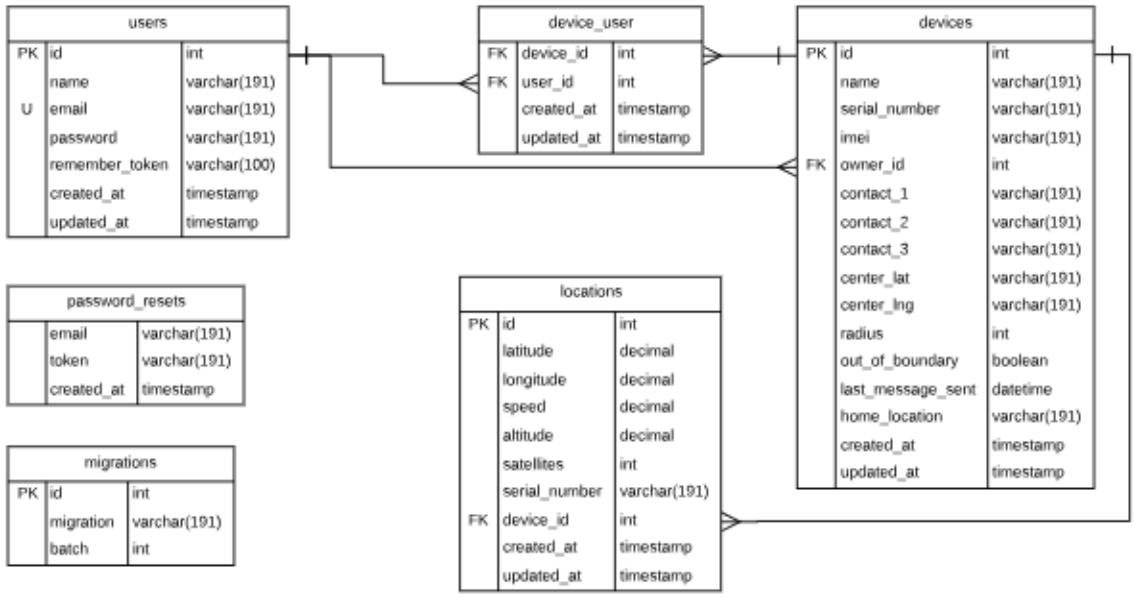
Heroku servis je besplatan za manje projekte zajedno s bazom podataka, ali je ograničeno i nema puno prostora. No to ne stvara nikakve probleme pri razvoju i testiranju aplikacije.

Cijela aplikacija je verzionirana putem *git* servisa i moguće ju je pronaći u prilozima pod P.3.2.

3.4.1. Struktura baze podataka

Baza podataka se sastoji od tablica za korisnike, uređaje za praćenje, lokacije, resetiranja lozinke, migracije i pivot tablice za Više-Na-Više (eng. *Many-To-Many*) relacijskog odnosa između tablice za korisnike i tablice za uređaje za praćenje. Migracijska tablica je namijenjena za praćenje migracija koje su se odvile u aplikaciji, te je sastavni dio Laravel programskog okvira. Tablica za

resetiranje lozinke je dio sustava za autentifikaciju (stvaranje jedinstvenog naloga za svakog korisnika kako bi mogao pristupiti mogućnostima aplikacije). Na slici 3.16. se nalazi strukture baze podataka. Tablica za migracije nema nikakve relacije prema nijednoj tablici kao i tablica za resetiranje lozinke, koja sprema samo adresu e-pošte korisnika na koji se šalje jedinstveni kod potreban za resetiranje lozinke.



Slika 3.16. *Struktura baze podataka*

Tablica *users* i *devices* povezuje ranije navedena Više-Na-Više relacija uz pomoć pivot tablice prema kojoj tablice *users* i d*evices* imaju Jedan-Na-Više relaciju. Te dvije tablice također imaju poveznicu između sebe Jedan-Na-Više između jedinstvenog identifikatora *users* tablice i *owner_id* polja u *devices* tablici. Na taj način se određuje vlasnik uređaja, te svaki uređaj može imati samo jednog vlasnika, dok svaki korisnik može biti vlasnik više uređaja. Slična stvar se događa s tablicom *locations* i *devices*. Svaki zapis o lokaciji uređaja se veže samo uz jedan uređaj, a svaki uređaj može imati više podataka o svojim prošlim lokacijama.

Podatci tablice *users*:

- *Id* – jedinstveni identifikator,
- *Name* – ime korisnika,
- *Email* – adresa e-pošte,
- *Password* – lozinka korisnika,

- *Remember_token* – token pamćenja kako bi korisnik mogao biti prijavljen uz pomoć kolačića,
- *Created_at* – vremenska oznaka stvaranja korisničkog naloga,
- *Updated_at* – vremenska oznaka zadnje promjene korisničkih podataka.

Podatci tablice *Devices*:

- *Id* – jedinstveni identifikator,
- *Name* – ime uređaja koje zadaje vlasnik (korisnik),
- *Serial_number* – serijski broj uređaja (nalazi se na A7 modulu),
- *Imei* – IMEI broj uređaja (nalazi se na A7 modulu),
- *Owner_id* – jedinstveni identifikator korisnika vlasnika,
- *Contact_1* – prvi od tri moguća kontakt broja,
- *Contact_2* – drugi od tri moguća kontakt broja,
- *Contact_3* – treći od tri moguća kontakt broja,
- *Center_lat* – koordinata zemljopisne širine kućne lokacije uređaja,
- *Center_lng* – koordinata zemljopisne dužine kućne lokacije uređaja,
- *Radius* – udaljenost koju uređaj može napustiti bez alarmiranja korisnika i navedenih kontakt brojeva,
- *Out_of_boundary* – označava status uređaja, da li je unutar ili izvan granica,
- *Last_message_sent* – vremenska oznaka kada su korisnik i navedeni brojevi alarmirani o napuštanju granica,
- *Home_location* – tekstualni zapis adrese/mjesta kućne lokacije uređaja,
- *Created_at* – vremenska oznaka stvaranja zapisa,
- *Updated_at* – vremenska oznaka zadnje promjene zapisa.

Podatci tablice *locations*:

- *Id* – jedinstveni identifikator,
- *Latitude* – koordinata zemljopisne širine lokacije,
- *Longitude* – koordinata zemljopisne dužine lokacije,
- *Speed* – brzina kretanja uređaja u trenutku očitavanja podataka o lokaciji,
- *Altitude* – visina uređaja u trenutku očitavanja podataka o lokaciji,
- *Satellites* – broj satelita uz pomoć kojih su dobiveni podatci o lokaciji,
- *Serial_number* – serijski broj uređaja koji predaje podatke o lokaciji,

- *Device_id* – jedinstveni identifikator uređaja koji predaje podatke o lokaciji,
- *Created_at* – vremenska oznaka stvaranja zapisa,
- *Updated_at* – vremenska oznaka zadnje promjene zapisa.

Podatci pivot tablice *device_user*:

- *Device_id* – jedinstveni identifikator uređaja koji se povezuje sa korisnikom,
- *User_id* – jedinstveni identifikator korisnika koji se povezuje sa uređajem,
- *Created_at* – vremenska oznaka stvaranja zapisa,
- *Updated_at* – vremenska oznaka zadnje promjene zapisa.

Podatci tablice p*assword_resets*:

- *Email* – e-pošta korisnika koji je zaboravio lozinku,
- *Token* – nasumično generiran skup znakova potreban za resetiranje lozinke,
- *Created_at* – vremenska oznaka stvaranja zapisa.

Podatci tablice m*igrations*:

- *Id* – jedinstveni identifikator,
- *Migration* – ime migracije,
- *Batch* – serija migracije.

3.4.2. Dodavanje lokacije u bazu podataka

Kada uređaj ostvari GPS komunikaciju s najmanje tri satelita, *TinyGPSPlus* biblioteka prevodi NMEA rečenice i podatci koji se dobiju se šalju GET zahtjevom na poslužitelj (aplikaciju) putem krajnje točke. Aplikacija uzima podatke iz krajnje točke i rastavlja na dijelove koji su odijeljeni znakom „*“ te sprema u polje. Zatim se uzima zadnji element polja koji predstavlja serijski broj uređaja uz pomoć kojega se traži uređaj iz tablice *devices* kako bi se mogao povezati s lokacijom. Nakon toga se svi podatci spremaju u model *location* koji se zatim sprema u bazu podataka. Slijede dva primjera pristupanja krajnjoj točki bez podataka i s podatcima.

```
http://calm-plateau-88601.herokuapp.com/api/location/new/[latitude]
*[longitude]*[speed]*[altitude]*[satellites]*[serial number]
```

```
http://calm-plateau-88601.herokuapp.com/api/location/new/45.554962*18.695514
*13*23*6*A06V022016030800
```

Bitan dio krajnje točke je *„ .../api/location/new/[data]*“. Dio prije „…/*api*/…“ ovisi o poslužitelju, tj. o domaćinu aplikacije, a dio *„[data]“* uvijek ostaje isti kao u primjerima.

U tablici 3.2. se nalazi primjer spremanja vrijednosti iz krajnje točke u varijable prije spremanja u bazu podataka.

Tablica 3.2. *Primjer spremanja podataka iz krajnje točke u varijable*

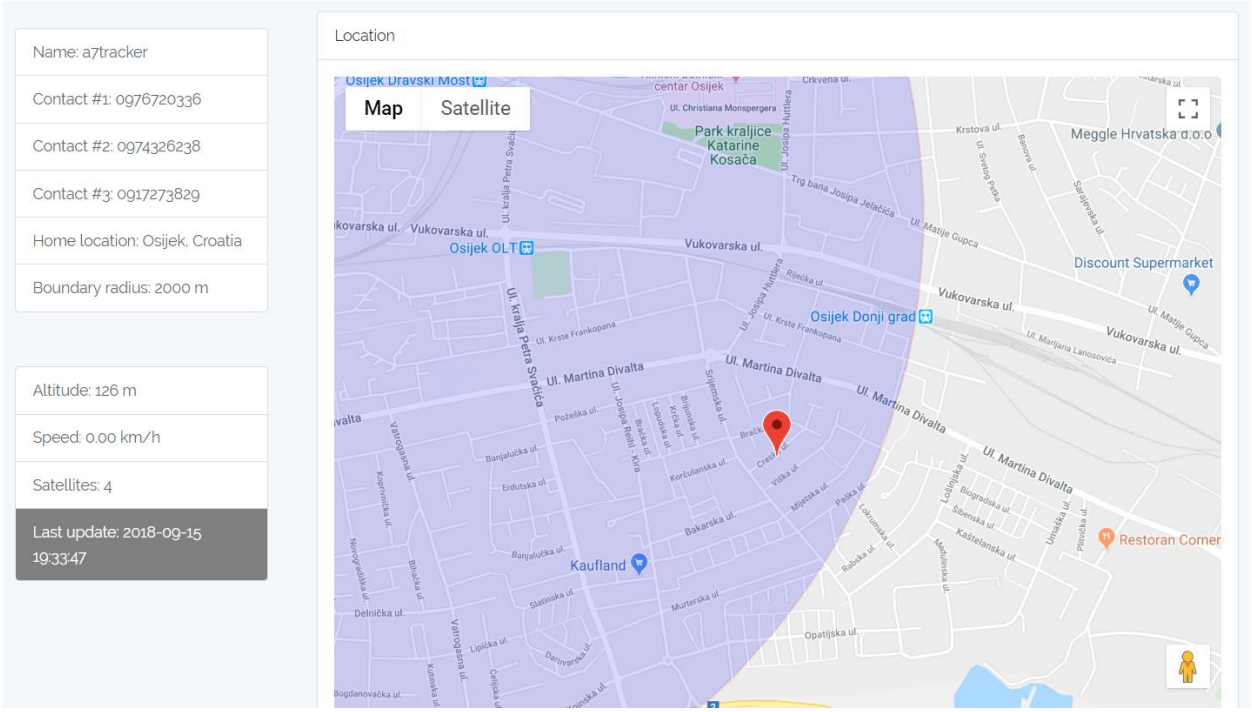
Ime varijable	**Vrijednost**
Latitude	45.554962
Longitude	18.695514
Speed	13
Altitude	23
Satellites	6
Serial_number	A06V022016030800

Nadmorska visina (engl. a*ltitude)* je izražena u metrima, a brzina kretanja (engl. s*peed)* u kilometrima po satu (km/h).

3.4.3. Prikaz lokacije

Za prikaz trenutnog položaja uređaja korišten je GooglMapper Laravel dodatak [32] koji pojednostavljuje korištenje Google Maps JavaScript API-ja. Dodatni podatci (koji se prikupljaju uz koordinate) su prikazani u tablici pored karte, zajedno s podatcima o uređaju.

Središte karte je postavljeno na trenutni položaj uređaja, a prostor u kojem se može kretati je označen plavom bojom. Kada se korisnik kreće s uređajem iscrtava se linija za posljednjih sto upisa lokacije u bazu. Može se zaključiti da na slici 3.17. korisnik stoji na mjestu.



Slika 3.17. *Prikaz lokacije uređaja*

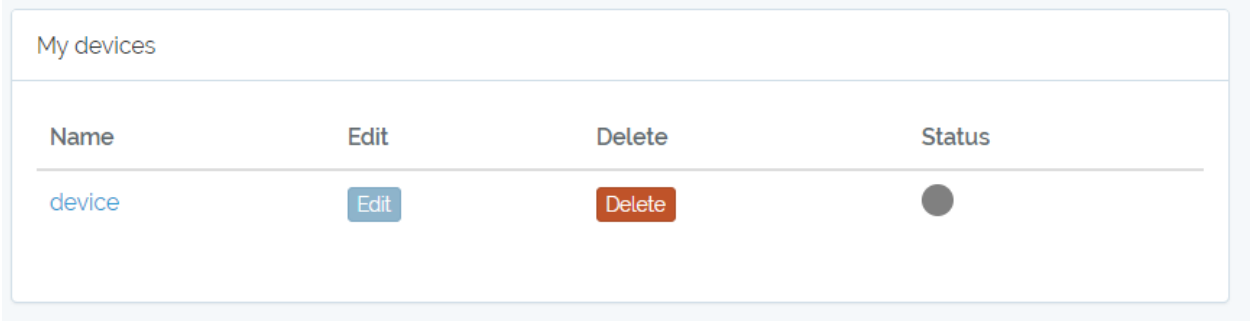
Na slici 3.17. se može primijetiti sivo polje u tablici koje označava da prikazani podatci nisu relevantni jer je prošlo više od 15 minuta od zadnjeg očitanja lokacije uređaja. Na karti je moguće odabrati način prikaza (karta ili satelitski prikaz) i moguć je prijelaz u *Street View* [33] način prikaza.

3.4.4. Obavijest korisnika o napuštanju uređaja iz granica

Nakon spremanja podataka u bazu provjerava se udaljenost uređaja od kućne lokacije koju je zadao korisnik koji je postavljen kao vlasnik uređaja. To omogućuje *Distance Matrix* API kojemu je potrebno poslati koordinate obje lokacije, te zatim API vraća udaljenost kao odziv koja se uspoređuje s postavljenom udaljenosti koju je vlasnik uređaja postavio pri povezivanju s uređajem. Ako je udaljenost koju je API vratio kao odziv veća, šalje se obavijest u obliku e-pošte vlasniku uređaja i SMS poruka brojevima koji su navedeni kao kontakti. Slanje poruka omogućuje Nexmo servis. To je jednostavan i brz API koji omogućuje slanje SMS poruka, uspostavljanje poziva preko interneta te još nekoliko usluga. [34]

Nakon obavještenja svih potrebnih o napuštanju granica uređaja, vrijednost varijable *out_of_boundary* se postavlja na „1“, odnosno istinito (engl. *true*), a u l*ast_message_sent* se zapisuje trenutak u kojemu su se poslale poruke. Te vrijednosti se mijenjaju da bi se mogao

promijeniti indikator statusa uređaja i da se ne zatrpava korisnika s porukama pri svakom unosu podataka u bazu kada je uređaj izvan dopuštene udaljenosti. Vrijeme između slanja poruka je dva sata. Slika 3.18. prikazuje uređaj čiji podatci nisu relevantni jer uređaj nije ažurirao svoju lokaciju više od 15 minuta.

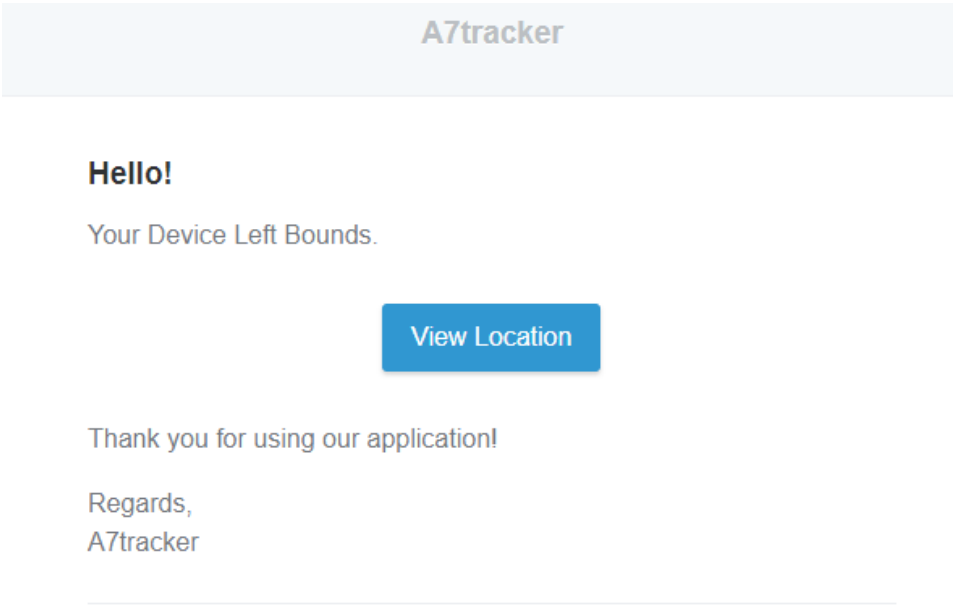


Slika 3.18. *Tipke za izmjenu podataka i brisanje uređaja, te indikator statusa uređaja*

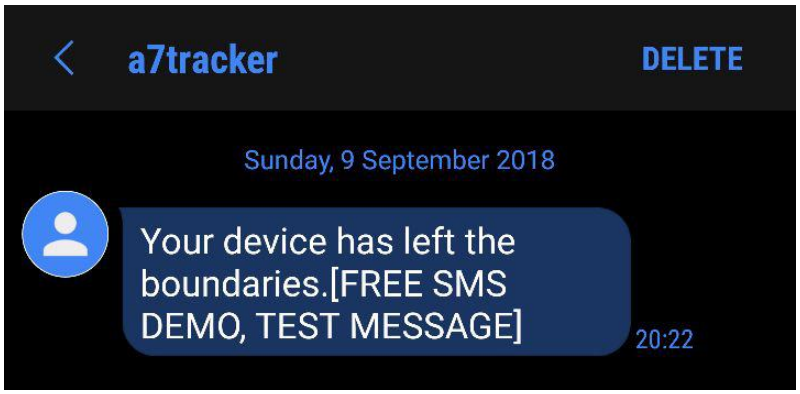
Na slici 3.19. se mogu vidjeti vrijednosti statusa uređaja koja se mijenja ovisno o tome da li je uređaj unutar granica (zelena boja), izvan granica (crvena boja) i da li su podatci o lokaciji vremenski relevantni, tj. da li je unutar 15 minuta očitana lokacija uređaja (siva boja ako nisu relevantni).

Slika 3.19. *Moguće vrijednosti statusa uređaja*

Na slici 3.20. i 3.21. su prikazani primjeri obavještenja korisnika putem e-pošte i SMS poruke.



Slika 3.20. *Primjer e-pošte*

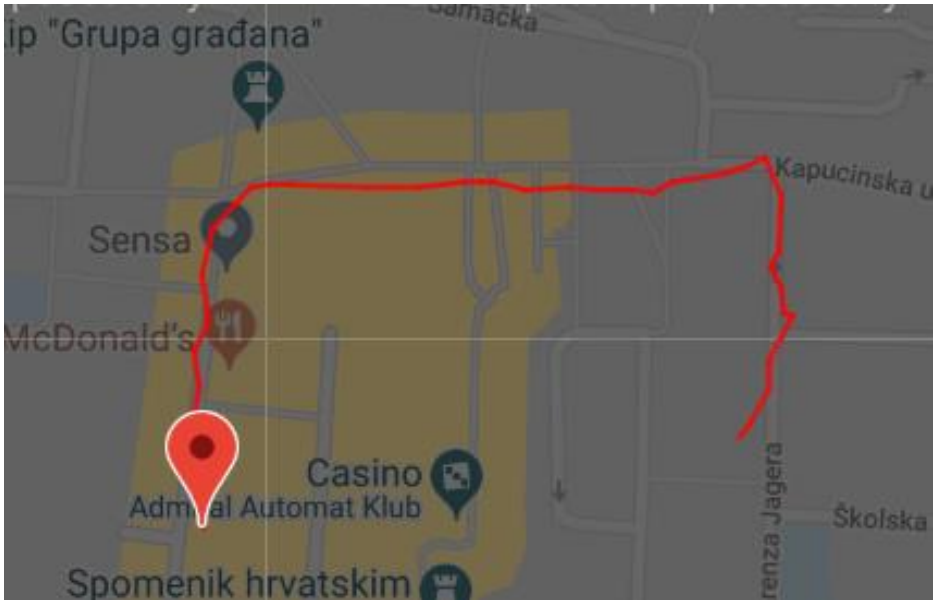


Slika 3.21. *Primjer SMS poruke*

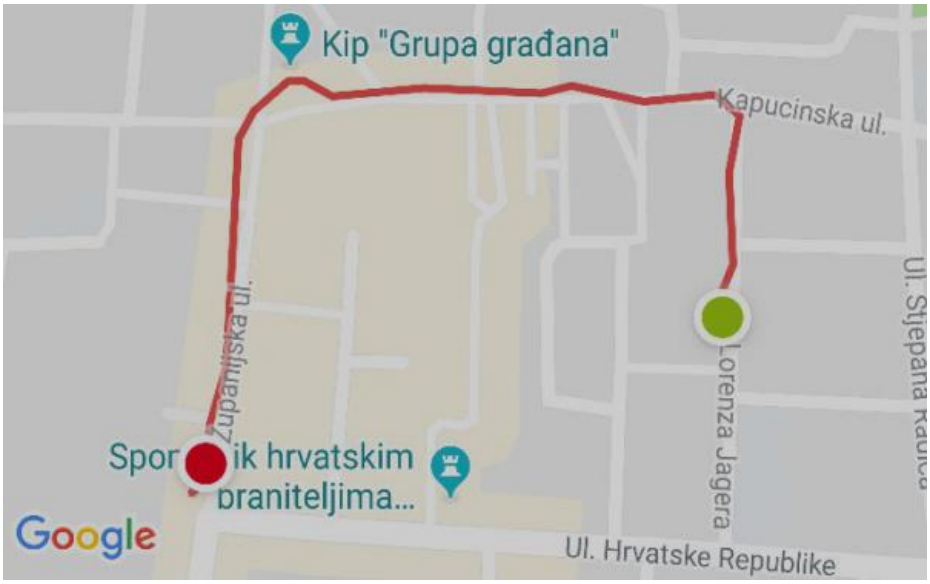
3.4.5. Rad sustava

Nakon uspješne izrade, sustav je testiran hodanjem po gradu i rezultat je uspoređen s mobilnim aplikacijama za praćenje kretanja tijekom rekreacije – *MapMyWalk* [35]. To je aplikacija za mobilne uređaje koja pruža još nekoliko mogućnosti uz funkciju aplikacije koja je potrebna za ovu usporedbu – praćenje kretanja tijekom rekreacije. Prije nego što uređaj počne očitavati lokaciju, baterija se spaja prvo na modul te se tipka za paljenje modula drži duže od dvije sekunde. Tada je modul upaljen i Arduino Uno se može spojiti na bateriju. Nakon toga je potrebno pričekati nekoliko minuta dok modul ne ostvari vezu sa satelitima. Na slici 3.22. se nalazi prikaz iscrtavanja

putanje kretanja na aplikaciji napravljenoj u ovom diplomskog radu koja se nalazi na Heroku servisu, a na slici 3.23. je prikaz putanje kretanja koristeći *MapMyWalk* aplikacije.



Slika 3.22. *Prikaz putanje kretanja na internetskoj aplikaciji na Heroku servisu*



Slika 3.23. *Prikaz putanje kretanja na mobilnoj aplikaciji MapMyWalk*

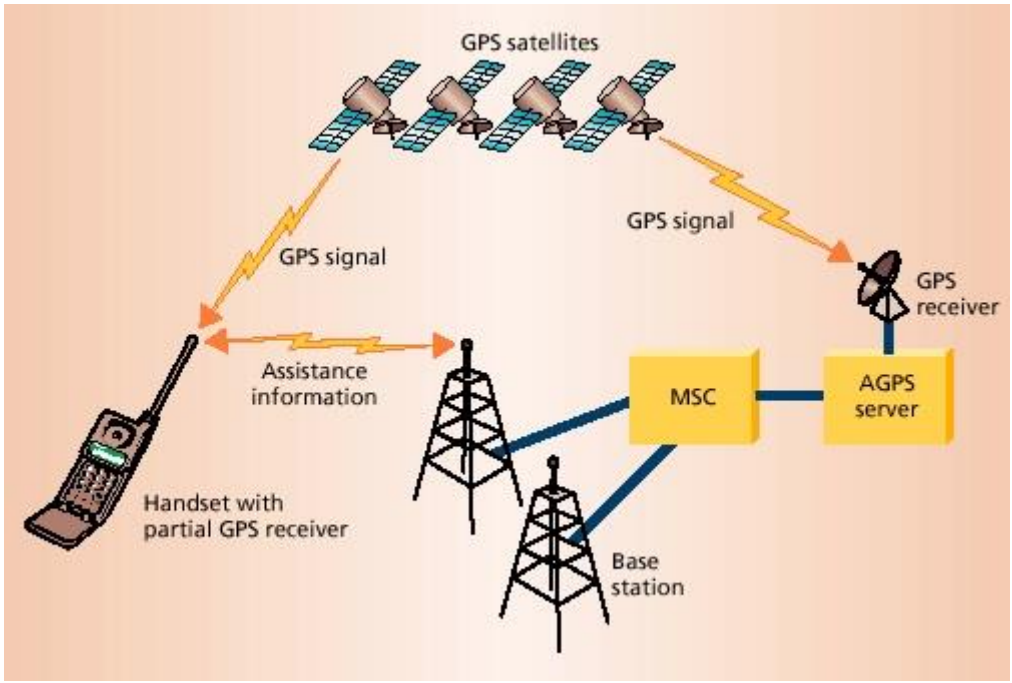
Usporedbom slika 3.22. i 3.23. je vidljivo da sustav napravljen u ovom diplomskog radu ima grešaka u spremanju lokacije, pogotovo na početku kretanja gdje je zabilježeno da se uređaj

nalazio duboko u zgradama dok je zapravo bio na cesti. Svakako treba uzeti u obzir da mobilna aplikacija *MapMyWalk* koristi mobilni internet za točnije očitavanje lokacije.

Potrošnja internetskih jedinica za slanje 49 skupova podataka o lokaciji je 0.4883 MB (engl. *megabyte)*. Što znači da sustav za jedno slanje podataka zahtjeva približno 10 kB (engl. *kilobyte*).

4. SLABOSTI SUSTAVA

Iako ovaj sustav radi i služi svrsi, određena ograničenja ga sprječavaju u ispunjenu njegove zadaće u potpunosti. Kako GPS jedino ne radi u zatvorenom prostoru potrebno je koristiti A-GPS (engl. *assisted global positioning system*) koji omogućuje procjenu položaja na temelju udaljenosti od modula do najbližih odašiljača za pristup mobilnom internetu. Na slici 4.1. se nalazi princip rada A-GPS-a.



Sl. 4.1. *Princip rada Assisted GPS-a [36]*

U popisu AT komandi modula je navedeno da je modul sposoban koristiti A-GPS, no pri pokušaju korištenja modul javlja grešku. Sustav također ima problema s povezivanjem sa satelitima, te je to još jedna stvar u kojoj bi A-GPS pomogao jer omogućuje brže povezivanje zbog korištenja internetske veze. Pri prvom spajanju, tzv. fiksiranju (engl. *TTFF – time to first fix)*, GPS preuzima određenu količinu podataka preko jako spore veze. Taj proces traje nekoliko minuta, ponekad i do deset minuta. Još jedan od problema je što sustav nema mogućnost slanja POST zahtjeva, pa se podatci moraju slati u bazu s GET zahtjevom. Spremanje podataka u bazu nema nikakvu zaštitu osim provjere postojanja uređaja sa serijskim brojem koji se predaje uz podatke lokacije, pa nije naročito sigurno. To je moguće riješiti koristeći HTTPS protokol za komunikaciju između uređaja i aplikacije. Sigurniji je od HTTP protokola jer je šifriran slojem za prijenosnu sigurnost (engl. *TLS – Transport Layer Security).*

5. ZAKLJUČAK

U ovome diplomskom radu je izrađen sustav za praćenje koji se sastoji od uređaja i internet aplikacije. Ovakav sustav može imati koristi u različitim situacijama. Od praćenja osobnih vozila, kofera, torbi ili nekog drugo predmeta, također može biti korišteno za praćenje osoba s određenim fizičkim ili psihičkim oštećenjima, npr. osoba s demencijom. Ako se dovoljno prilagodi veličina i težina, uređaj može biti korišten za praćenje kućnih ljubimaca. Ovakva inačica je prenosiva ali je definitivno prevelika i otvorena. Prije puštanja proizvoda u produkciju bi se smanjile dimenzije i prilagodilo bi se za nošenje u džepu, ušivanje u podstavu jakne ili jednostavno postavljanje u pretinac automobila ili nekog drugog prijevoznog sredstva. Problemi sa slabom komunikacijom A7 modula i satelita u zatvorenim prostorima je moguće riješiti implementacijom jačeg, tj. skupljeg modula te omogućavanje ranije navedenog A-GPS-a. Cijeli sustav se također može unaprijediti pojednostavljenjem svih komponenti, tj. odbacivanjem svega osim neophodnih dijelova upravljačke jedinice i čipa modula što bi smanjilo dimenzije i potrošnju električne energije. Dodatna nadogradnja sustava bi mogla biti implementacija *Socket-a*, što bi omogućilo prikaz lokacije i ostalih podataka u stvarnom vremenu.

LITERATURA

[1] Arduino Uno upravljačka jedinica: https://store.arduino.cc/arduino-uno-rev3 [21.9.2018.]

[2] A7 Ai-Thinker GSM/GPRS/GSM modul: https://www.ebay.com/itm/A7-GSM-GPRS-GPS-3-In-1-Module-Shield-DC-5-9V-STM32-51MCU-Universal-For-Arduino/112777039297?epid=12005554306&hash=item1a42090dc1:g:PegAAOSwDkVaav21 [14.8.2018]

[3] Izvorni kod Laravel okvira: https://github.com/laravel/laravel [15.8.2018.]

[4] Dokumentacija ATmega328P čipa: http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Summary.pdf [21.9.2018.]

[5] *Software Serial* biblioteka:https://www.arduino.cc/en/Reference/SoftwareSerial [21.9.2018.]

[6] Arduino platforma: https://www.arduino.cc/en/Guide/Introduction [21.9.2018.]

[7] Opis GSM: https://searchmobilecomputing.techtarget.com/definition/GSM [21.9.2018.]

[8] Opis GPRS: http://docbook.rasip.fer.hr/ddb/res/45/Ch3.7.html [21.9.2018.]

[9] Dokumentacija A7 modula: https://www.makerfabs.com/desfile/files/A6_A7_A6C_datasheet-EN.pdf [21.9.2018.]

[10] AT komande za A7 modul: https://www.makerfabs.com/desfile/files/A6A7A6CA20_AT_Commends.pdf [21.9.2018.]

[11] Opis GPS: https://www.gps.gov/systems/gps/ [21.9.2018.]

[12] Opis NMEA: https://www.gpsinformation.org/dale/nmea.htm [21.9.2018.]

[13] Opis jQuery-a: https://jquery.com/ [21.9.2018.]

[14] Opis JSON: http://www.json.org/ [21.9.2018.]

[15] Opis *Geocoding* API-ja: https://developers.google.com/maps/documentation/geocoding/intro [21.9.2018.]

[16] Opis *Places* APi-ja: https://developers.google.com/places/web-service/intro [21.9.2018.]

[17] Opis *Distance Matrix* API-ja: https://developers.google.com/maps/documentation/distance-matrix/start [21.9.2018.]

[18] Opis *Blade* sustava predložaka: https://laravel.com/docs/5.7/blade [21.9.2018.]

[19] Opis Git sustava za verzioniranje koda: https://laravel.com/docs/5.7/blade [21.8.2018.]

[20] Opis Heroku sustava: https://www.heroku.com/platform [21.9.2018.]

[21] Opis HTTP protokola: https://www.w3.org/Protocols/HTTP/1.1/rfc2616bis/draft-lafon-rfc2616bis-03.html [21.9.2018.]

[22] Opis MySQL-a: https://www.mysql.com/products/ [21.9.2018.]
[23] Opis PostgreSQL-a: https://www.postgresql.org/ [21.9.2018.]
[24] *TinyGPSPlus* biblioteka: https://github.com/mikalhart/TinyGPSPlus [21.9.2018.]
[25] *Free* NMEA prevoditelj: http://freenmea.net/ [21.9.2018.]
[26] *TinyGSM* biblioteka: https://github.com/vshymanskyy/TinyGSM [21.9.2018.]
[27] Specifikacije baterije: https://hobbyking.com/en_us/turnigy-nano-tech-950mah-2s-25-50c-lipo-pack.html?___store=en_us [21.9.2018.]
[28] Bootstrap: http://getbootstrap.com/ [21.9.2018.]
[29] Guzzle HTTP klijent: https://github.com/guzzle/guzzle [21.9.2018.]
[30] Toastr JavaScript biblioteka za obavijesti na internet aplikaciji: https://cdnjs.com/libraries/toastr.js/latest [21.9.2018.]
[31] Heroku CLI programski dodatak: https://devcenter.heroku.com/articles/heroku-cli [21.9.2018.]
[32] GooglMapper Laravel dodatak: https://github.com/bradcornford/Googlmapper [21.9.2018.]
[33] Google *Street View* usluga: https://www.google.com/streetview/ [21.9.2018.]
[34] Nexmo: https://www.nexmo.com/ [21.9.2018.]
[35] MapMyWalk – aplikacija za praćenje kretanja tijekom rekreacije: https://www.mapmywalk.com/ [21.9.2018.]
[36] Slika principa rada A-GPS-a: https://cellphonetrackers.org/gps-versus-a-gps.html [21.9.2018.]

SAŽETAK

Sustav za praćenje

U ovom diplomskom radu napravljen je sustav za praćenje, koji se sastoji od uređaja za praćenje i internetske aplikacije. Uređaj je izrađen pomoću upravljačke jedinice Arduino Uno i modula A7. Internet aplikacija izrađena je s PHP programskim jezikom pomoću Laravel programskog okvira. Modul se koristi za pristup i slanje GPS podataka na internet aplikaciju, koja ih sprema u bazu podataka i prikazuje trenutnu lokaciju uređaja i obavještava korisnike ako se uređaj previše udalji od kućne lokacije. Korisnik se može povezati s uređajem da samo gleda gdje se nalazi ili kao vlasnik, čime stječe mogućnost postavljanja kućne lokacije uređaja, udaljenost oko kućne lokacije gdje se uređaj smije kretati i kontakt brojeve koji se obavještavaju kada se uređaj previše udaljio od kućne lokacije.

Ključne riječi: GPS, Arduino, A7, Laravel, PHP, baza podataka, sustav za praćenje.

ABSTRACT

Tracking system

In this graduate thesis tracking system was made, which consists of tracking device and web application. The device was created with the Arduino Uno control unit and A7 module. Web application was made with PHP programming language using Laravel Framework. Module is used for accessing and sending GPS data to the web application, which is saving it in database, displaying device's current location and alarming users if device leaves given radius around home location. The user can connect to the device just to watch where it is, or as the owner, thereby gaining the ability to set up the home location of device, radius around home location which device should not leave, and contact numbers which are alarmed when the device leaves the area.

Keywords: GPS, Arduino, A7, Laravel, PHP, database, tracking system.

ŽIVOTOPIS

Mislav Dominović rođen je 5.6.1995. godine u Osijeku. Nakon završene Osnovne Škole Antuna Mihanovića u Osijeku, upisuje Opću gimnaziju u Osijeku te maturira 2013. g. Iste godine upisuje sveučilišni studij na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija, smjer računarstvo. Nakon završenog preddiplomskog studija 2016. godine, na istom fakultetu upisuje diplomski studij – smjer Informacijske i podatkovne znanosti.

Potpis:

_____

PRILOZI

Prilog P.3.1. – Programski kod za programiranje Arduino Uno upravljačke jedinice

```
#include <SoftwareSerial.h>
#include <TinyGsmClient.h>
#include <ArduinoHttpClient.h>
#include <string.h>
#include <TinyGPS++.h>

SoftwareSerial mySerial(8, 9);

#define TINY_GSM_MODEM_A7
#define TINY_GSM_RX_BUFFER 256

#define S1debug true
#define S_debug true

TinyGsm modem(mySerial);

const char apn[]  = "internet.ht.hr";
const char user[] = "";
const char pass[] = "";

char server[] = "calm-plateau-88601.herokuapp.com";
String resource = "/api/location/new/";
String serial_number = "A06V022016030800";
char char_array[] = "";
const int  port = 80;

TinyGPSPlus gps;
TinyGsmClient client(modem);

void setup()
{
  Serial.begin(9600);
  delay(10);

  mySerial.begin(9600);
  delay(3000);

  while (mySerial.available())
  {
    mySerial.read();
  }

  Serial.println(F("Initializing modem..."));
  modem.init();

  String modemInfo = modem.getModemInfo();
  Serial.print(F("Modem: "));
  Serial.println(modemInfo);

  SendAT("AT+GPS=1", 2000, S1debug);

  SendAT("AT+GPSRD=1", 2000, S1debug);
}

void loop()
{
  String buffLng;
  String buffLat;
  String buffSpeed;
```

```
  String buffAlt;
  String buffSat;
  String buff;

  resource = "/api/location/new/";

  mySerial.flush();

  while (mySerial.available())
  {
    gps.encode(mySerial.read());

    if (gps.location.isUpdated())
    {
      buffLat = String (gps.location.lat(), 6);
      buffLng = String (gps.location.lng(), 6);
      buffSpeed = String (gps.speed.kmph(), 1);
      buffAlt = String (gps.altitude.meters(), 1);
      buffSat = String (gps.satellites.value());
      buff = buffLat + "*" + buffLng + "*" + buffSpeed + "*" + buffAlt + "*" + buffSat + "*" + serial_number;
      Serial.println(buff);

      resource += buff;

      SubmitRequest(resource);
    }
  }
}

void SubmitRequest(String resource) {

  SendAT("AT+GPSRD=0", 2000, S1debug);

  Serial.print(F("Waiting for network..."));
  if (!modem.waitForNetwork()) {
    Serial.println(" fail");
    delay(10000);
    return;
  }
  Serial.println(" OK");

        Serial.print(F("Connecting to "));
        Serial.print(apn);
        if (!modem.gprsConnect(apn, user, pass)) {
          Serial.println(" fail");
          delay(10000);
          return;
        }
        Serial.println(" OK");

        Serial.print(F("Connecting to "));
        Serial.print(server);
        if (!client.connect(server, port)) {
          Serial.println(" fail");
          delay(10000);
          return;
        }
        Serial.println(" OK");

        client.print(String("GET ") + resource + " HTTP/1.0\r\n");
        client.print(String("Host: ") + server + "\r\n");
        client.print("Connection: close\r\n\r\n");

        unsigned long timeout = millis();
        while (client.connected() && millis() - timeout < 10000L) {
```

```
125.           while (client.available()) {
126.             char c = client.read();
127.             Serial.print(c);
128.             timeout = millis();
129.           }
130.         }
131.         Serial.println();
132.
133.         client.stop();
134.         Serial.println(F("Server disconnected"));
135.
136.         modem.gprsDisconnect();
137.         Serial.println(F("GPRS disconnected"));
138.
139.         SendAT("AT+GPSRD=1", 2000, S1debug);
140.       }
141.
142.       void SendAT(String command, const int timeout, boolean debug) {
143.         String response = "";
144.         mySerial.println(command);
145.         delay(5);
146.         if (debug) {
147.           long int time = millis();
148.           while ( (time + timeout) > millis()) {
149.             while (mySerial.available()) {
150.               response += char(mySerial.read());
151.             }
152.           }
153.           Serial.print(response);
154.         }
155.       }
```

Prilog P.3.2. – Programski kod internetske aplikacije

Programski kod internet aplikacije napravljene u PHP programskom jeziku i Laravel programskom okviru za sustav za praćenje se nalazi na: https://github.com/mdominovic/a7tracker.

Aplikaciji na Heroku servisu se može pristupiti na: https://calm-plateau-88601.herokuapp.com/ .