Pametno parkiralište korištenjem Interneta stvari i Wi-Fi tehnologije

Pavelić, Andrej

VISOKO UČILIŠTE ALGEBRA

ZAVRŠNI RAD

Pametno parkiralište korištenjem Interneta stvari i Wi-Fi tehnologije

Andrej Pavelić

Zagreb, kolovoz 2017.

„Pod punom odgovornošću pismeno potvrđujem da je ovo moj autorski rad čiji niti jedan dio nije nastao kopiranjem ili plagiranjem tuđeg sadržaja. Prilikom izrade rada koristio sam tuđe materijale navedene u popisu literature, ali nisam kopirao niti jedan njihov dio, osim citata za koje sam naveo autora i izvor, te ih jasno označio znakovima navodnika. U slučaju da se u bilo kojem trenutku dokaže suprotno, spreman sam snositi sve posljedice uključivo i poništenje javne isprave stečene dijelom i na temelju ovoga rada“.

U Zagrebu, datum.

Predgovor

Zahvaljujem se svome mentoru dr.sc. Goranu Đambiću na pomoći i vodstvu tijekom izrade završnog rada.

Zahvaljujem se svim profesorima Visokog učilišta Algebra na prenesenom znanju i motivaciji tijekom studija.

Zahvaljujem mojoj djevojci koja mi je pomogla pri izradi ovog rada i pružila podršku tijekom studija.

Posebno se želim zahvaliti mojim roditeljima koji su mi uvijek bili podrška i poticali me na odlične rezultate prilikom cijelog školovanja.

Prilikom uvezivanja rada, Umjesto ove stranice ne zaboravite umetnuti original potvrde o prihvaćanju teme završnog rada kojeg ste preuzeli u studentskoj referadi

Sažetak

Tijekom izrade rada bit će razvijen sustav koji će pomoći ljudima u pronalasku slobodnog parkirnog mjesta te pritom smanjiti gužve na cestama. Pametno parkiralište korištenjem Interneta stvari i Wi-Fi tehnologije nije najoptimalnije rješenje zbog kompleksnosti instalacije, integracije i visokih troškova. Ideja samog sustava je korisna i potrebna za građane, ali potrebno je pronaći alternativno rješenje koje će biti jednostavnije za uvođenje, a jednako korisno za stanovništvo.

Ključne riječi: internet stvari, pametno parkiralište, raspberry pi, senzori, wi-fi, lorawan

Summary

In this thesis, smart parking system will be design which enables users to find the nearest parking area and automatically reduce traffic jams. Smart parking using Internet of Things and Wi-Fi technology is not the most optimal solution due to the complexity of installation, integration and high cost, but the idea of the smart parking itself it is useful and necessary for population. It is necessary to find an alternative solution that will be easier to implement and equally useful.

Keywords: internet of things, smart parking, raspberry pi, senosores, wi-fi, lorawan

Sadržaj

1. Uvod

Kako urbano stanovništvo sve više raste, tako raste i gospodarski razvoj, a s time i gušća urbana mobilnost što ima utjecaja na promet. Vozila koja traže slobodno parkirno mjesto uzrokuju 30% gužvi i zastoja u prometu (Shoup D.C., 2006), što predstavlja veliki problem za urbanu mobilnost. Veliki postotak vozača koji uzrokuju zastoje su turisti. Turisti najčešće nisu upoznati s gradom u kojeg putuju. Oni mogu tražiti slobodno parkirno mjesto jedino očima i ograničenim informacijama poput prometnih znakova koji označavaju parkiralište.

Kada bi vozači koristili usluge pametnog parkirališta putem svojih pametnih telefona ili tableta, mogli bi bez uzaludnog gubitka vremena pronaći parkirno mjesto, što bi rezultiralo smanjenjem potrošnje goriva, onečišćavanja okoliša i gužvi u prometu. S obzirom da bi vozači unaprijed imali informaciju o zauzeću parkirnih mjesta na željenoj lokaciji, mogli bi unaprijed organizirati način putovanja.

Pomoću integriranih senzora na svakom parkirnom mjestu, postojala bi mogućnost detekcije automobila koji su nepropisno parkirani ili su parkirani na parkirnim mjestima namijenjenim za osobe s invaliditetom, te bi im se naplatila odgovarajuća kazna.

U radu će biti kreiran model prototipa parkirališta i web aplikacije koji će komunicirati pomoću Raspberry Pi-a, Wi-Fi tehnologije i senzora. U radu je zadana hipoteza koja će se kroz sam rad testirati. U svrhu dokazivanja postavljene hipoteze u radu je korištena metoda komparacije. Na taj način će se usporediti pametno parkiralište putem Wi-Fi tehnologije s alternativnim rješenjima. Podaci su prikupljeni iz domaćih i stranih izvora, te relevantnih Internet stranica kao što su službene stranice Raspberry Pi-a.

Rad je podijeljen u pet dijelova. U uvodu je obrazložena problematika, svrha rada, korištene znanstvene metode te je objašnjena struktura rada. U drugom dijelu je projektni zadatak u kojem je zadana hipoteza i definiran je cilj istraživanja. U trećem dijelu pojašnjen je pojam Internet stvari, razvoj koncepta Internet stvari, komponente od kojih se Internet stvari sastoje, načini komunikacije između Internet stvari te gdje se u praksi koriste. U četvrtom dijelu objašnjena je sama izrada makete pametnog parkirališta kroz sve aplikacijske slojeve. Također su detaljno pojašnjeni i načini komuniciranja između Raspberry Pi uređaja, senzora, Web servera i Web aplikacije. Peti dio se sastoji od analize alternativnih rješenja pametnog parkiralište i od analize ostvarenosti zadane hipoteze. U posljednjem dijelu rada, zaključku, prikazane su činjenice završnog rada.

2. Projektni zadatak

U nastavku će biti zadana hipoteza koja će se pomoću metode komparacije i izradom prototipa pametnog parkirališta potvrditi ili opovrgnuti. Biti će pojašnjeni ciljevi istraživanja rada te će se opisati na koji način pametni sustavi mogu doprinijeti određenom području te na koji način bi pridonijeli Hrvatskoj.

2.1. Hipoteza rada

Hipoteza je prihvaćanjem, pretpostavka na kojoj se temelji neki zaključak, koja služi napretku istraživanja i objašnjavanja, a da nije dokazana iz drugih načela te da nije potvrđena (verificirana) iskustvom. Prema kritičkomu Popperovom racionalizmu, sve empirijsko – znanstvene teorije sa svojom "općenitom" valjanošću imaju samo hipotetički karakter. One se, pak, ne mogu nikada konačno verificirati jer se ne mogu promatrati svi pojedinačni slučajevi, ali se one (prema kritičkom racionalizmu) ipak mogu u danom slučaju promatranjem opovrgnuti (falsificirati) (Belak, 2006).

Hipoteza: Integracija i instalacija pametnog parkirališta korištenjem Interneta stvari i Wi-Fi tehnologije je isplativa.

Hipoteza će se analizirati pomoću makete pametnog parkirališta koje će koristiti ultrazvučne senzore i Raspberry Pi, te putem Wi-Fi tehnologije slati stanje parkirnih mjesta na Web aplikaciju. Također će se napraviti analiza alternativnih rješenja implementacije i integracije sustava pametnog parkiralište te će se na temelju toga izvršiti metoda komparacije.

2.2. Cilj istraživanja

Cilj istraživanja je potvrditi ranije navedene hipoteze. Pomoću hipoteza doći će se do zaključka da li je pametno parkiralište pomoću Interneta stvari i Wi-Fi tehnologije iskoristivo za potrebe gradskog prijevoza. Okosnica će biti istraživanje na konkretnom primjeru parkirališta (maketi).

U radu će se koristiti metoda komparacije kako bi se usporedilo rješenje pametnog parkiralište pomoću Interneta stvari i Wi-Fi tehnologije s alternativnim rješenjima što će pomoći pri dokazivanju hipoteze rada. Uspoređivati će se kompleksnost implementacije i integracije, kompleksnost razvoja sustava te isplativost uvođenja sustava u rad.

2.3. Doprinos pametnih sustava

Sve je češća pojava pametnih i inteligentnih sustava u suvremenim organizacijama koji osiguravaju veću kvalitetu i raznovrsnost podataka, s ciljem širenja poslovanja i stjecanja šireg pogleda na svoje klijente. Uvođenje pametnog parkirališta u rad bi osiguralo kvalitetniji život u gradovima, smanjenje nervoze među vozačima, a time i veću udobnost putovanja. Neki gradovi su već počeli raditi na tome da postanu „pametni“. Za primjer se može uzeti Barcelona koja je još 2012. godine započela s implementacijom pametnih sustava. Neki od tih sustava su javni prijevoz, ulična rasvjeta, parking i upravljanje otpadom (http://datasmart.ash.harvard.edu/news/article/how-smart-city-barcelona-brought-the-internet-of-things-to-life-789).

Ovaj rad bi mogao doprinijeti hrvatskim gradovima i Hrvatskoj u tehnološkom razvoju, ali i u rastu hrvatskog turizma, čime bi se uvođenjem pametnog parkirališta i općenito pametnih sustava poboljšala kvaliteta života. Pametni sustavi bi se mogli iskorištavati i u svrhu atrakcije koja bi privukla turiste iz svih dijelova svijeta, što bi pospješilo prihodu za gradski i državni proračun.

3. Internet stvari

Internet stvari (engl. *Internet of Things*) se odnose na fizičke i virtualne objekte koji imaju jedinstvene identitete i imaju mogućnost povezivanja na Internet kako bi se postigla inteligencija na određenim aplikacijama. Takve aplikacije bi mogle „pametnije“ iskorištavati energiju, bilo bi jednostavnije primjenjivati logistiku, industrijsku kontrolu, maloprodaju, poljoprivredu i mnoge druge domene. Internet stvari su nova revolucija interneta koja jako brzo raste i sve je popularnija, a pomoću njih napreduje senzorska tehnologija, mobilni uređaji, bežična komunikacija, umrežavanje i tehnologija u oblaku (Bahga, 2010; Atzoria et al., 2010). Predviđa se da će u 2020. godini biti više od 50 milijardi uređaja/stvari povezanih s Internetom (Evans et al., 2011).

3.1. Komponente Internet stvari

Internet stvari se sastoje od određenih komponenata te će se u nastavku kratko opisati svaka komponenta:

- **Uređaj** – uređaj Internet stvari omogućuje identifikaciju, daljinsko upravljanje i pokretanje, te mogućnost praćenja,
- **Resursi** – softverske komponente na Internet stvari koje služe za pristup, obradu i pohranu podataka sa senzora. Resursi također uključuju i komponente koje omogućuju mrežni pristup,
- **Kontroler** – izvorna usluga koja se pokreće na uređaju i koristi Web usluge. Šalje podatke s uređaja na Web uslugu i prima naredbe iz aplikacije, te se na taj način vrši kontrola nad uređajem,
- **Baza podataka** – može biti instalirana lokalno ili u oblaku, te pohranjuje generirane podatke,
- **Web usluge** – veza između uređaja koji je povezan na Internet, aplikacije, baze podataka i analize podataka. Web usluge mogu biti implementirane korištenje HTTP i REST principa ili uporabom protokola spojne točke (engl. *socket),*
- **Analiza komponenata** – podaci dobiveni iz Internet stvari se analiziraju te se generiraju rezultati u obliku koji je razumljiv korisniku. Analiza se može izvoditi i pohranjivati u baze podataka lokalno i u oblaku,

- **Aplikacija** – aplikacije za Internet stvari pružaju sučelje koje korisnici mogu koristiti. Najčešće se koristi za kontrolu i praćenje različitih aspekta sustava, kao i za prikaz i vizualizaciju obrađenih podataka.

3.2. Razvoj koncepta

Razvojem koncepta, Internet stvari se počinju primjenjivati u raznim područjima djelatnosti. Najveći interes i najveća primjena pokazala se upravo u sektoru transporta i prometa, tj. u odjelima logistike i prometa. Ideja koncepta Internet stvari je povezivanje stvari sa bežičnim sustavima kao što su mobilni uređaji, senzori i slični uređaji koji su u stanju komunicirati međusobno te surađivati s drugim uređajima kako bi se postigao neki određeni cilj. Glavna zadaća i moć Internet stvari je utjecaj na aspekte svakodnevnog života i ponašanje potencijalnih korisnika (Bahga, 2010; Atzoria et al., 2010).

Bitnu stavku u razvoju Internet stvari zauzela je i interoperabilnost među uređajima koja pruža uređajima mogućnost prilagodbe i samostalnosti te osigurava sigurnost, integritet i privatnost.

3.3. Primjena Internet stvari u praksi

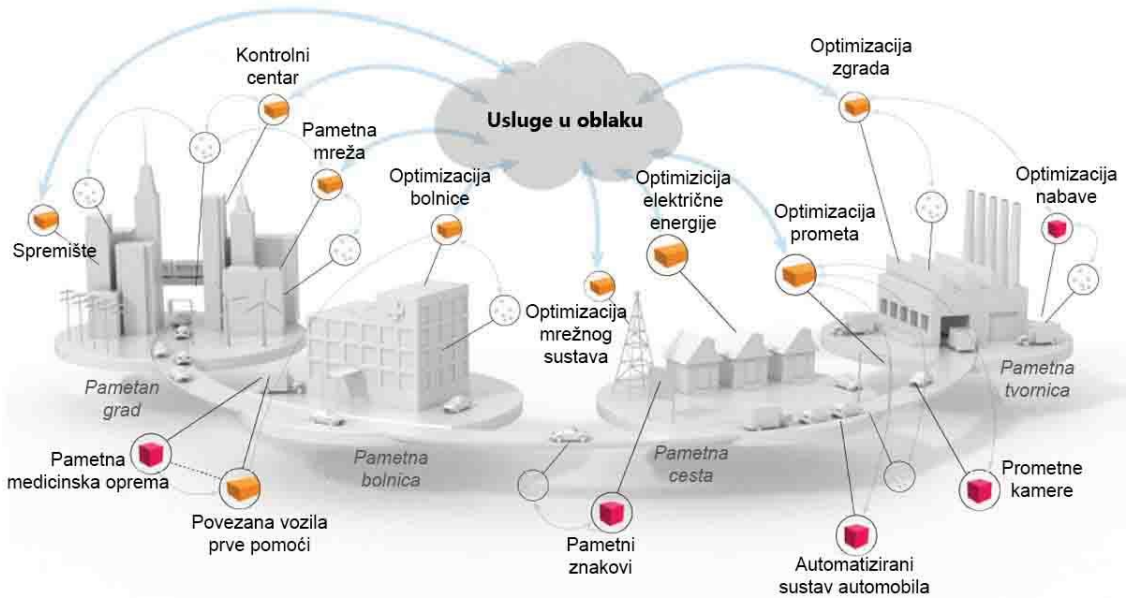
Internet stvari se u sve većem broju koriste te postaju svakodnevnica. U nastavku će biti opisani neki od primjena Internet stvari u praksi:

- **Sustavi za nadgledanje struktura i građevina**

Sustavi za nadgledanje struktura i građevina su sustavi koji prate strukturno „zdravlje“ zgrada i mostova (Slika 3.1). U navedenom sustavu kompletna platforma je u stvarnom vremenu te funkcionira uz pomoć Pro-Trinket[1] mikrokontrolera, NRF modula[2], WI-FI modula i Raspberry Pi-a. Nadzor se vrši pomoću senzora koji kontinuirano prikupljaju podatke, te se uz pomoć matematičkih algoritama otkriva „zdravlje“ određene strukture. Ukoliko dođe do štete, Pro-Trinket mikrokontroler određuje mjesto i veličinu štete i šalje ih putem NRF modula na Raspberry Pi koji služi kao središte za sve prikupljene podatke iz više različitih NRF modula. Nakon obrade podataka uz pomoć Raspberry Pi-a, podaci se

[1] Pro-Trinket je mala mikrokontrolerska pločica velike snage koja ima mogućnost spajanja s drugim miktrokontrolerima kao što su Raspberry Pi i Arduino
[2] NRF modul je modul koji omogućava bežičnu komunikaciju između dva uređaja

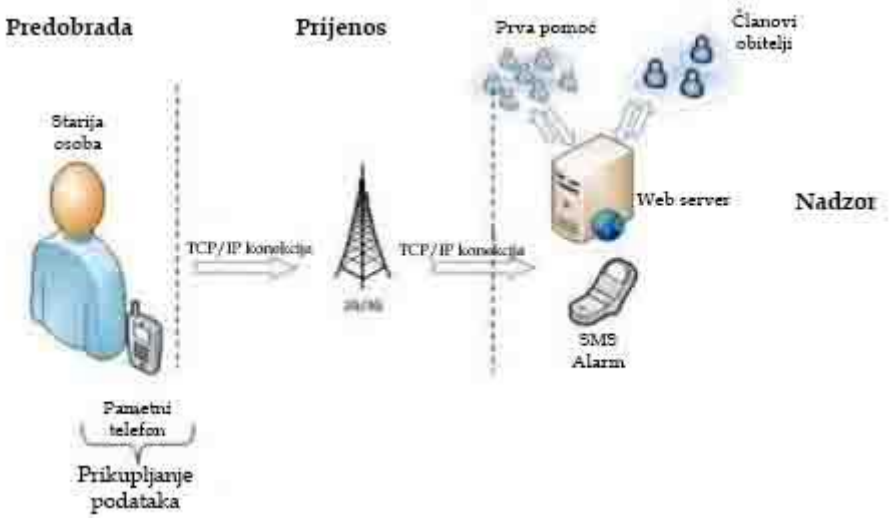
spremaju u oblaku putem Wi-Fi modula te dalje mogu biti analizirani i kontrolirani (Abdelgawad el al., 2017).



Slika 3.1 Način funkcioniranja sustava za nadgledanje struktura i građevina (Botta, 2016)

- **Detekcija pada putem Interneta stvari**

Primarni cilj sustava za detekciju pada putem Interneta stvari je detekcija pada starijih osoba i pravovremeno pružanje pomoći. Sustav funkcionira uz pomoć senzora, poveznika (engl. *gateway*), te pozadinskog dijela aplikacije koji ima tri osnovne komponente: 3D senzor, mikrokontroler i bežični komunikacijski modul. Podaci dolaze iz 3D senzora putem komunikacijskog sučelja korištenjem algoritma za detekciju pada. Kada algoritam otkrije pad aktivira se *push* opcija koja obavještava skrbnika u stvarnom vremenu. Na taj način se unesrećenoj osobi može brzo pružiti prva pomoć (Nguyen Gia at al., 2016). Način funkcioniranja sustava detekcije pada može se vidjeti na Slika 3.2.



Slika 3.2 Način funkcioniranja sustava detekcije pada

- **Zelene kuće**

Sustav Zelena kuća (Slika 3.3) je napredan sustav koji služi u svrhu praćenja vremenskih uvjeta u stakleniku. Ovakav sustav funkcionira uz pomoć Arduina[3], senzora i Wi-Fi tehnologije. Senzori (temperaturni senzori i senzori vlage) u stvarnom vremenu prikupljaju podatke koji se pomoću analognih i digitalnih ulaza šalju na Arduino. Putem Wi-Fi tehnologije podaci se prosljeđuju na Web ili mobilnu aplikaciju, te se vizualiziraju u obliku grafova (Dongarsane et al., 2017).

Slika 3.3 Način funkcioniranja sustava Zelene kuće[4]

[3] Arduino je mikrokontrolerska pločica koja se ponaša kao platforma za stvaranje Internet stvari
[4] MQTT (engl. *Message Queuing Telemetry Transport*) je pojednostavljeni protokol za komunikaciju između uređaja. Detaljno je opisan u poglavlju 3.4.4

3.4. Web usluge i komunikacijski protokoli

Web usluga je programski sustav dizajniran na način da podržava interoperabilne interakcije putem mreže. Ona sadrži sučelja koja su opisana u obliku koji je moguće obraditi u računalu. Drugi sustavi mogu komunicirati sa uslugama Weba na način koji je opisan u njenom opisu korištenjem SOAP poruka, najčešće korištenjem protokola HTTP sa XML serijalizacijom u suradnji sa ostalim Web-orijentiranim normama (Ljubi et al., 2013).

Komunikacijski protokoli definiraju pravila komunikacije na mreži. Jednako kao što ljudi imaju pravila u komunikaciji, tako i uređaji u mreži definiraju pravila na temelju kojih izmjenjuju pakete. Komunikacijski protokoli su osnova rada svake mreže i bez njih bi bilo kakva komunikacija bila nemoguća (Grgić et al., 2013).

Osnovne uloge protokola u računalnim mrežama su (Grgić et al., 2013):

- Definicija oblika poruka koje se prenose mrežom,
- Definicija pravila ponašanja na mreži (tko, kada i na koji način smije komunicirati),
- Definicija veličine i semantike polja unutar paketa koji se prenosi mrežom,
- Definicija sustava koji su potrebni za uspješnu komunikaciju.

3.4.1. REST usluge

REST (*Representational State Transfer*) usluge su usluge bazirane na HTTP protokolu i imaju vrlo veliku skalabilnost. U osnovi se oslanjaju na jednostavne i standardizirane elemente, a za identifikaciju resursa koriste URI pomoću kojeg se rješava i problem globalnog adresiranja prostora kao i otkrivanja usluga i resursa. U trenutku pronalaska nekog željenog resursa, njime se može manipulirati korištenjem operacija za ubacivanje, uređivanje, dohvat i brisanje (PUT, POST, GET, DELETE).

Prevladavajuće je mišljenje da su REST usluge Weba općenito jednostavnije za programiranje i korištenje jer se njihovom dizajnu koriste samo W3C/IETF standardi (poput HTML-a, XML-a ili URI-ja). Uz veliku količinu standardnih klijenata koji ih podržavaj razvijen je i velik broj jednostavnih alata, čime je sigurno jednostavno usvajanje za sve koji ih žele koristiti. REST usluge Weba može koristiti razmjerno velik broj klijenata (Ljubi et al., 2013).

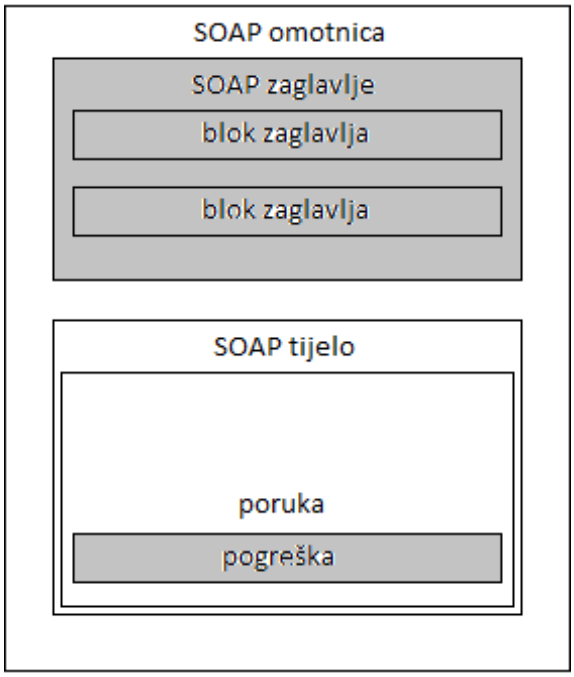
3.4.2. SOAP protokol

SOAP (*Simple – Object Access Protocol*) je komunikacijski protokol zasnovan na XML jeziku koji služi za razmjenu informaciju među različitim entitetima u mreži. Najčešće korišteni protokol je HTTP. Komunikacija u SOAP protokolu najčešće je bazirana na XML-u i njegovim normama, poput XML Scheme i XML Namespace. Poruke se prenose pomoću SOAP poruka koje u XML-u čuvaju zapis u tekstualnom obliku koji jedan od entiteta želi prenijeti drugome (Ljubi et al, 2013).

3.4.2.1 SOAP Poruke

SOAP poruka (Slika 3.4) je XML dokument koji se sastoji od (Ljubi et al., 2013):

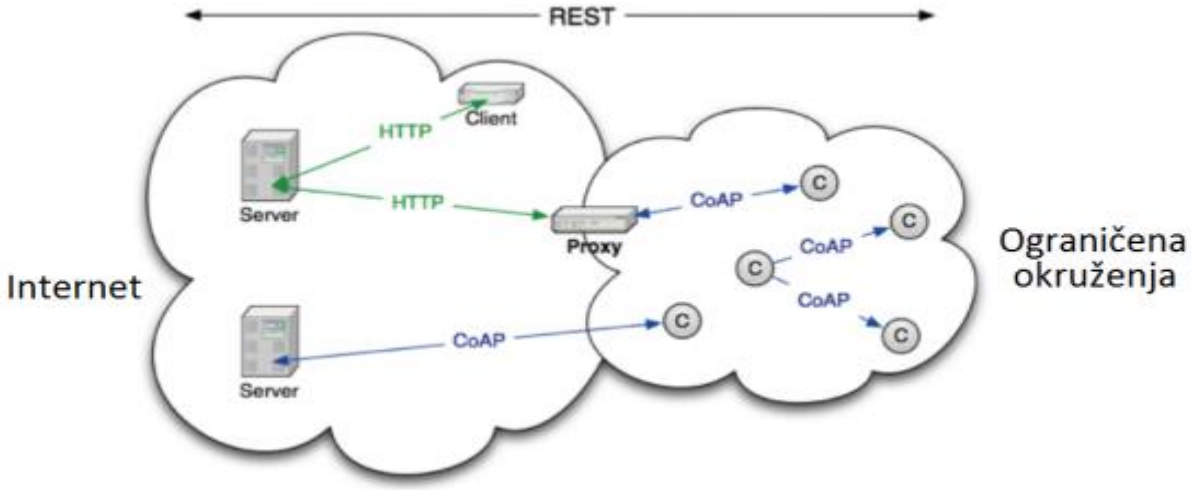
- Omotnice, pomoću koje definiramo da je XML dokument zapravo SOAP poruka,
- Zaglavlja, u kojem se definira na koji će način informacije iz poruke biti procesuirane,
- Tijela poruke, gdje se nalazi sama poruka koja se prenosi,
- Pogreške, koja sadržava informaciju o pogrešci koja se mogla dogoditi tijekom, prijenosa i obrade informacija.



Slika 3.4 Primjer SOAP poruke

3.4.3. CoAP protokol

Protokol CoAP (*Constrained Application Protocol*), koji je vidljiv na Slika 3.5, je softverski protokol koji pomoću mrežnih čvorova omogućava komunikaciju preko mreža (Interneta). Ovaj protokol namijenjen je za komunikaciju jednostavnih elektroničkih uređaja (kao što su senzori, prekidači, ventili i brojni drugi uređaji), koji imaju malu procesnu snagu, malu potrošnju energije te najčešće su povezani na mreže s niskom propusnošću i čestim ispadanjima i gubitcima, a potrebno ih je nadzirati ili upravljati daljinski, putem mreže. Protokol je razvijen od strane organizacije otvorenih standarda, koja se volonterski bavi razvojem i promicanjem Internet standarda, pod nazivom IETF[5], odnosno od strane radne grupe koja se bavi ograničenim RESTful okruženjima, pod imenom CoRE[6]. Smatra se da će u budućnosti, kompletiranjem protokola CoAP, milijuni uređaja u raznim domenama aplikacije koristiti navedeni protokol.



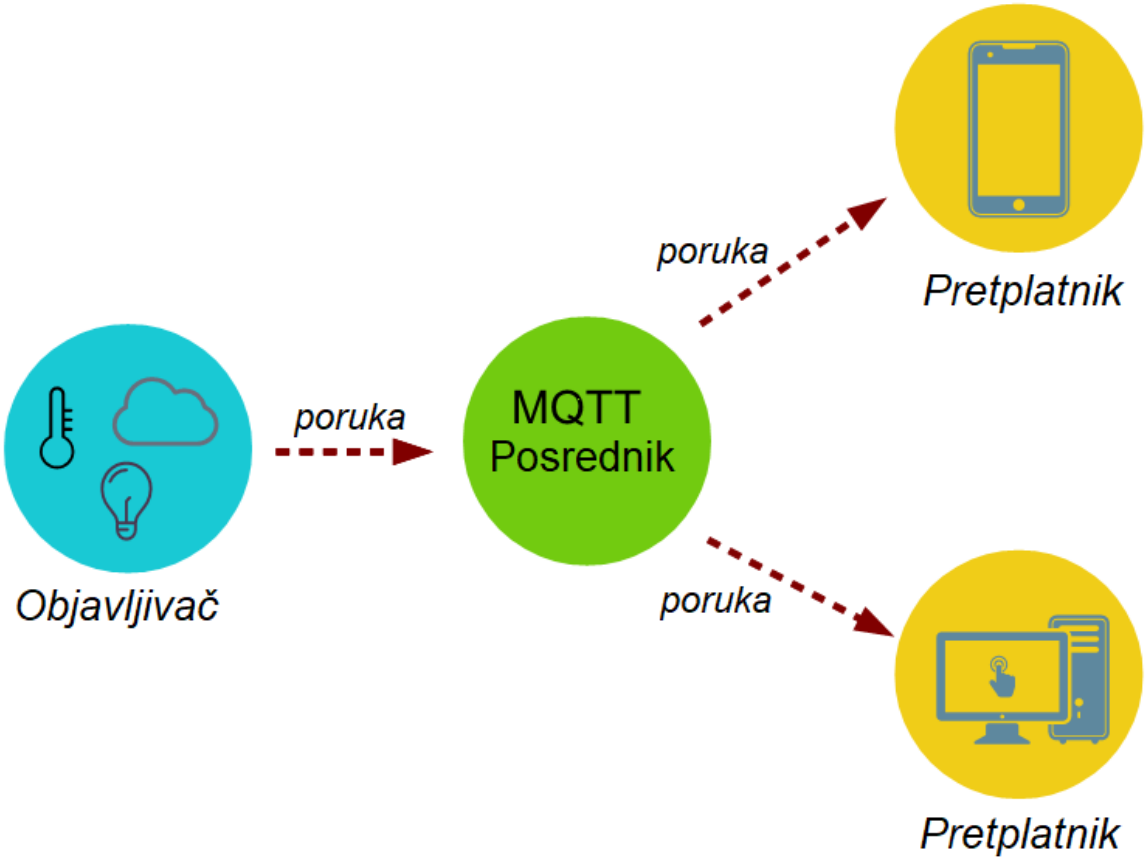
Slika 3.5 Mreža zasnovana na protokolu CoAP[7]

3.4.4. MQTT protokol

MQTT (*Message Queuing Telemetry Transport*) je pojednostavljeni protokol za komunikaciju između uređaja koji je inicijalno razvio IBM, no danas je otvoreni protokol. MQTT je baziran na principu objavi – pretplati, a koristi TCP/IP mrežni protokol (Slika 3.6). Kreiran je da bude lagan i jednostavan za implementaciju. Zbog ograničenog okruženja te

[5] IETF (*Internet Engineering Task Force*) je organizacija posvećena razvoju i promoviranju internetskih standarda i protokola
[6] https://datatracker.ietf.org/wg/core/charter/
[7] Izvor: https://nvisium.com/blog/2015/05/27/implementing-coap-secure-way-part-i/

velikih kašnjenja smanjuje zahtjeve na propusnost mreže što se pokazalo kao dobra funkcionalnost kod nestabilnih mrežnih sustava gdje prevladavaju male brzine prijenosa i gdje postoji mogućnost od gubitka podataka.



Slika 3.6 Prikaz komunikacije korištenjem MQTT protokola[8]

Objavi – pretplati je obrazac koji omogućuje razmjenu poruka na način da objavljivači ne šalju poruke unaprijed poznatom primatelju, već ih spremaju u klase, koristeći teme. Nakon toga pretplatnici se pretplaćuju na određene, njima zanimljive teme i na taj način primaju sve poruke koje objavljivači objavljuju za pojedinu temu.

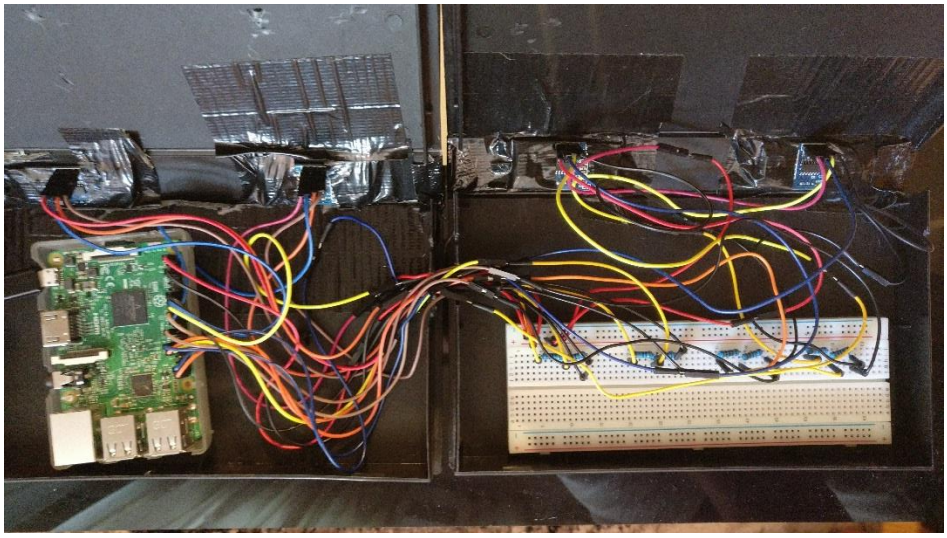
[8] Izvor: https://www.hackster.io/bucknalla/mqtt-micropython-044e77

4. Izrada modela pametnog parkirališta

Praktični rad se sastoji od četiri osnovna dijela:

- **Model parkirališta**

Model pametnog parkirališta izrađen je u obliku makete. Pomoću senzora i Raspberry Pi uređaja prikupljaju se podaci s parkirnih mjesta. Podaci se obrađuju i šalju na centralni Web server. Najveći dio posla odrađuje Raspberry Pi na kojem je pokrenuta Python skripta pomoću koje se vrši detekcija zauzeća parkirnog mjesta i slanje podataka prema serveru. Na Slika 4.1 je prikaz spoja Raspberry Pi uređaja i ultrazvučnih senzora.



Slika 4.1 Spoj Raspberry Pi s ultrazvučnim senzorima

Svaki senzor ima četiri vlastita pina koja služe za povezivanje s određenim uređajem. Jedan pin je pin uzemljenja, koji je u radu spojen s pinom uzemljenja na Raspberry Pi uređaju, jedan je pin napajanja, te *trig* i *echo* pinovi koji su objašnjeni u poglavlju 4.1 Eksperimentalna ploča služi za brzo sklapanje sklopova. Raspberry Pi napaja eksperimentalnu ploču, te ploča napaja svaki od senzora. GPIO pinovi (Slika 4.8) služe kako bi se povezali *trig* i *echo* pinovi sa senzora. Spojeni su na eksperimentalnu ploču pomoću spojnih žica i kondenzatora.

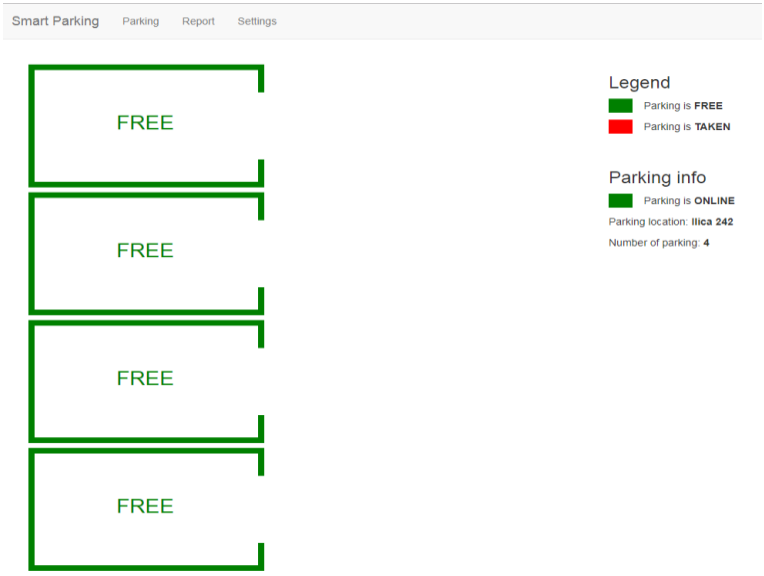
- **Web server**

Web server je podatkovni sloj razvijen u programskom jeziku Java. Namijenjen je da pristupa bazi podataka. Sastoji se od dva modula. Jedan modul komunicira s Raspberry Pi

uređajem, dok je drugi modul REST usluga kojoj može pristupiti bilo tko, a ponaša se kao poslovni sloj cjelokupnog sustava.

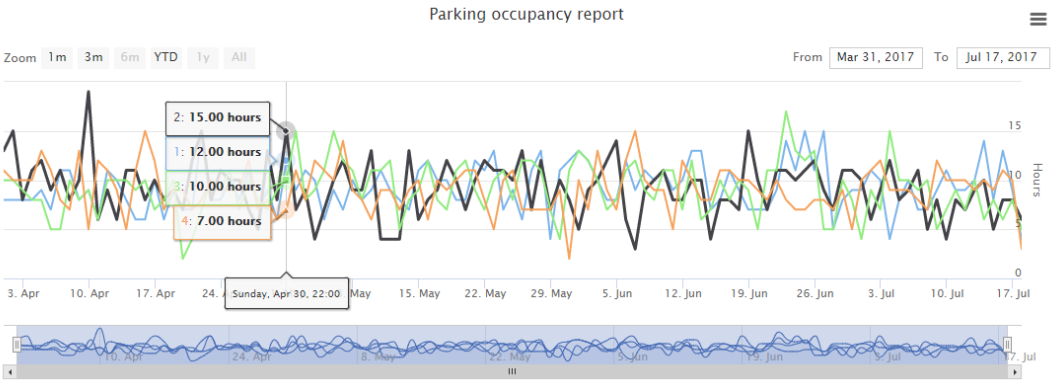
- **Web aplikacija**

Web aplikacija je prezentacijski sloj koji vizualizira podatke dobivene s makete parkirališta. Prikaz je u stvarnom vremenu. Kada automobil dođe ili ode s parkirnog mjesta, to je vidljivo u istom trenutku i na Web aplikaciji. Aplikacija je razvijena u programskog jeziku C# u MVC okruženju. Prikaz slobodnog parkirališta prikazan je na Slika 4.2.



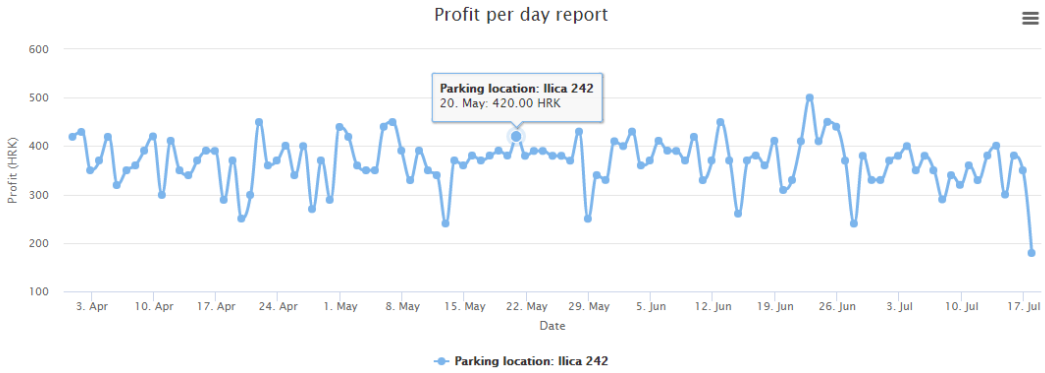
Slika 4.2 Prikaz slobodnog parkirališta

Web aplikacija ima funkcionalnost prikazivanja izvještaja. Prvi izvještaj je izvještaj zauzeća parkirnih mjesta na dnevnoj razini. Pomoću njega se može provjeriti koliko sati je pojedino parkirno mjesto bilo zauzeto na određeni dan (Slika 4.3)



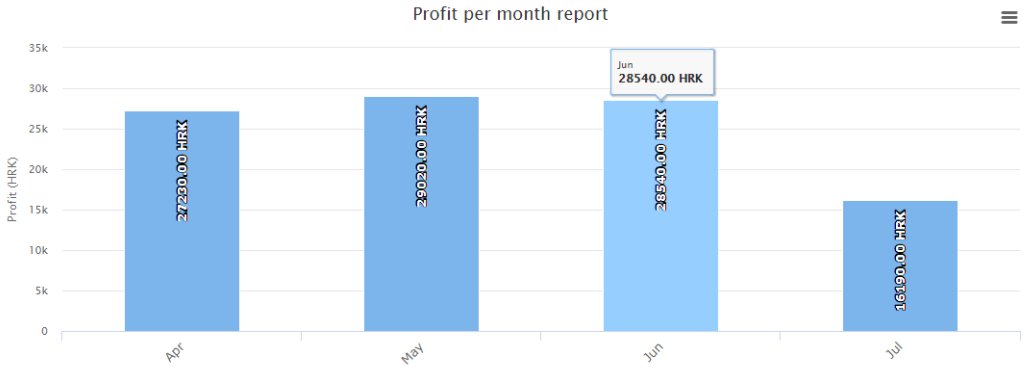
Slika 4.3 Izvještaj zauzeća parkirnih mjesta na dnevnoj razini

Drugi izvještaj pokazuje profit. Na postavkama Web aplikacije implementirana je cijena parkirnog mjesta po satu. Ovisno o cijeni izračunava se ostvareni profit na dnevnoj razini. Na Slika 4.4 može se vidjeti grafički prikaz profita na dan 20. svibnja.



Slika 4.4 Izvještaj profita po danu za određenu lokaciju parkirališta

Treći, posljednji izvještaj je izvještaj profita na mjesečnoj razini (Slika 4.5)



Slika 4.5 Izvještaj profita na mjesečnoj razini

- **Baza podataka**

Baza podataka je mjesto gdje su pohranjeni svi podaci sustava. Od trenutnog stanja pojedinog parkirnog mjesta do svake promijene na pojedinom parkirnom mjestu. Platforma baze podataka je MongoDB koja se temelji na dokumentima te nije klasična relacijska baza podataka već je NoSQL[9] baza podataka koja se pokazala kao dobar izbor zbog jednostavne instalacije, dobre dokumentacije te je besplatna za korištenje.

[9] NoSQL je naziv za neracionalne baze podataka koja se temelji na principu ključ - vrijednost

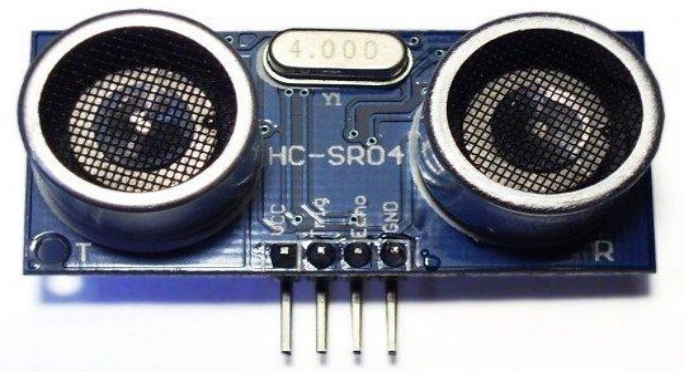
4.1. Senzori

Senzori su uređaji koji detektiraju ili reagiraju na neku pojavu iz fizičkog okruženja. Mogu reagirati na svjetlost, kretanje, vlagu, temperaturu, distancu, pritisak ili bilo koju drugu pojavu. Izlaz senzora je signal koji se pretvara u podatak koji je čitljiv čovjeku. Podaci dobiveni iz senzora mogu biti analizirani i pomoću njih se mogu dobiti razne korisne informacije i znanja.

Ultrazvučni senzor HC-SR04 zbog svoje kompleksnosti ima svoj poseban mikroprocesor pomoću kojeg vraća izmjerenu duljinu u apsolutnim jedinicama. Omogućava bez kontaktno mjerenje u opsegu od 2 cm do 200 cm uz preciznost od oko 3 mm. Preciznost senzora ovisi o veličini objekta i o materijalu objekta. Što je objekt veći, jači se zvuk od njega odbija što govori da je preciznost bolja. Isto tako kod materijala, na primjer, veća preciznost će biti kod metalnih objekata, nego kod kartonskih ili spužvastih objekata. Efektivni kut je 15 stupnjeva. Napaja se naponom od 5 V, a potrošnja struje iznosi manje od 2 mA. Ultrazvučni senzori koristi ultrazvučne valove i pomoću toga određuje udaljenost nekog predmeta. Senzor ima dva osnovna djela: *trig* i *echo*. Mikrokontrolerom se šalju 5 V signali na *trig* pin u trajanju od minimalno 10 mikrosekundi i tako se aktivira ultrazvučni transduktor[10] koji odašilje osam impulsa od 40 Khz te određuje da li postoji povratni signal, tj. čeka njihovu refleksiju. Kada senzor registrira reflektirani impuls šalje podatke natrag mikrokontroleru preko *echo* pina. Vrijeme povratka signala je mjera udaljenosti objekta od senzora. Ukoliko je signalu potrebno dulje od 35 milisekundi da se vrati, senzor registrira da je predmet izvan dosega. *Echo* će uvijek izlaziti nizak (0), osim ako nije pokrenut, a u tom slučaju izlazi s 5 V. Ultrazvučni senzor, pored mikroprocesora, sastoji se i od mikrofona i od zvučnika (Carullo, 2001).

Ultrazvučni senzori (Slika 4.6) u radu služe za detekciju vozila na pametnom parkiralištu te vrše komunikaciju s Raspberry Pi 3 uređajem.

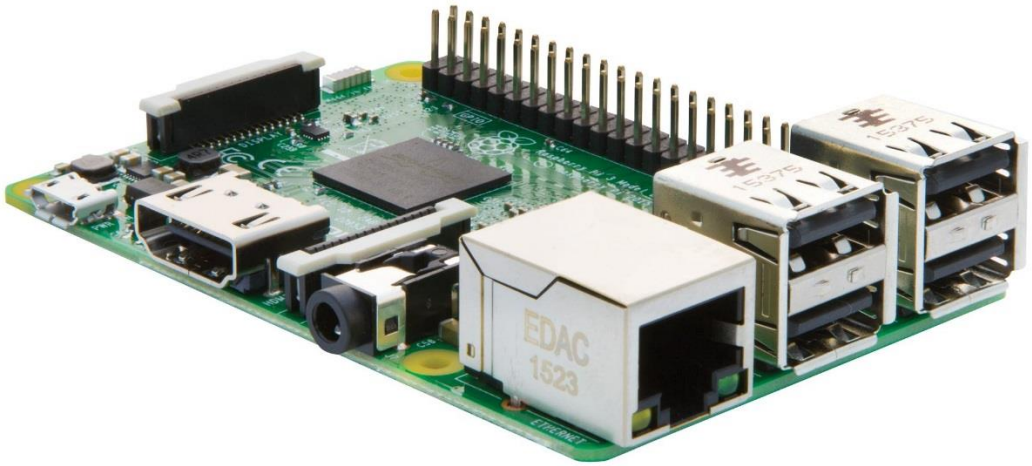
[10] Transduktor – uređaj koji konvertira jedan tip energije u drugi. Konverzija može biti u / iz električne, elektromehaničke, elektromagnetske, fotonske ili nekog drugog oblika energije. Pojam *transduktor* najčešće se podrazumijeva za uporabu sa značenjem senzor/detektor.



Slika 4.6 Ultrazvučni senzor

4.2. Raspberry Pi

Raspberry Pi (Slika 4.7) je mikrokontrolerska pločica veličine kreditne kartice te je sposoban pokretati aplikacije jednako kao i standardno stolno ili prijenosno računalo. Prvobitna ideja Raspberry Pi računala bila je povećanje interesa programiranja i općenito programskog inženjerstva, no nedugo nakon, Raspberry Pi je postao prihvaćen kao univerzalna programibilna kontrola koja je danas integrirana u mnogobrojne strojeve i M2M[11] aplikacije (Wallace at al, 2016; https://www.raspberrypi.org).



Slika 4.7 Raspberry Pi 3

U praktičnom djelu rada Raspberry Pi služi kao centralno mjesto podataka koji dolaze iz ultrazvučnih senzora. Programski se podaci obrađuju kako bi se dobila udaljenost objekta.

[11] M2M (*Machine to Machine*) označava izravnu komunikaciju između uređaja pomoću bilo kojeg komunikacijskog kanala, uključujući žičnu i bežičnu komunikaciju

Obrađeni podaci se šalju na Web aplikaciju koja prikazuje podatke. Način komunikacije i izvedba biti će detaljnije objašnjeni u nastavku rada.

4.2.1. Komponente

Komponente i specifikacija Raspberry Pi 3 modela prikaza je u Tablica 4.1. Pojedinosti komponenata objašnjene su u nastavku (Wallace at al., 2016; https://www.raspberrypi.org).

Tablica 4.1 Specifikacija Raspberry Pi 3 uređaja

Procesor	Quad Core 1.2GHz Broadcom BCM2837 64bit CPU
Radna memorija	1GB RAM
Memorija	Micro SD
USB	4 x USB 2
Pinovi	40-pin GPIO
Bluetooth	Bluetooth Low Energy (BLE)
HDMI	Full size HDMI
Mrežna kartica	BCM43438 wireless LAN
Napajanje	Micro USB do 2.5A
Proširenja	- 4 Pole stereo output - CSI Camera port - DSI Display port - Ethernet port

- **Napajanje**

Napajanje je ploča s 5 V mikro USB priključkom. Potrošnja energije ovisi o vanjskim periferijskim uređajima koji su priključeni na uređaj. Za novije generacije Raspberry Pi modela potrebno je nešto više snage (od barem 2100 mA) budući da imaju naprednije procesore s četiri jezgre. Najčešće se kao napajanje koriste punjači mobilnih telefona s mikro USB priključkom, međutim, za stabilniju radnju, gdje je potrebna veća snaga, potrebna su jača napajanja.

U maketi pametnog parkirališta koristi se napajanje od 4000 mA iz razloga što su na uređaj priključena četiri ultrazvučna senzora, što zahtjeva više snage.

- **HDMI pristupnik**

HDMI pristupnik je veza između uređaja i ekrana. Pruža visoku rezoluciju (engl. *High Definition*) slike i koristi se da bi se olakšala konfiguracija i instalacija Raspberry Pi uređaja. Svi modeli pružaju rezoluciju od 640x350 do 1920x1200.

- **CSI priključak**

Pomoću CSI priključka na Raspberry Pi se može priključiti kamera koja je posebno dizajnirana za Raspberry Pi. Takva kamera pruža visoku rezoluciju za izradu fotografija, kao i za snimanje videa.

- **Audio priključak**

Izlaz kroz 3.5 mm priključnicu za slušalice koji također podržava analogni video izlaz za novije modele Raspberry Pi uređaja

- **Mrežni priključak**

Pruža brzinu od 10/100 Mbit/s putem kabela RJ45 lokalne mreže (LAN). Dostupan je u svim uređajima.

- **Bežična mreža**

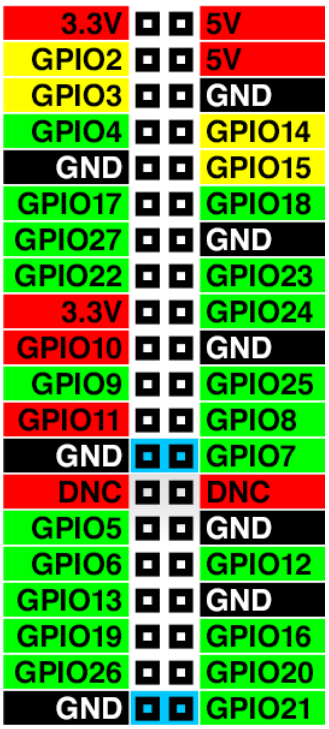
Na Raspberry Pi 3 modelu ugrađen je čip koji omogućava bežično povezivanje uređaj na Internet.

- **USB pristupnik**

Podržava sve USB uređaje kao što su miš, tipkovnica, web kamera, prijenosni tvrdi diskovi i slično. USB verzija priključka je 2.0, te postoji 4 ulaza na Raspberry Pi 3 modelu.

- **GPIO pinovi**

Skup univerzalnih igala ulaznog i izlaznog konektora za opće namjene. Najčešće se koriste za povezivanje eksperimentalne ploče pomoću koje je moguće povezati i druge vanjske uređaje kao što su primjerice senzori.



Slika 4.8 Prikaz GPIO pinova na Raspberry Pi 3 modelu[12]

- **DSI priključak**

Služi za povezivanje Raspberry Pi uređaja i zaslona koji je osjetljiv na dodir.

- **Mikro SD utor**

Služi za umetanje Micro SD kartice koja služi za spremanje željenog operacijskog sustava ili programa i alata potrebnih za izvršavanje određenih radnji.

4.2.2. Operacijski sustav

Raspberry Pi radi na posebnim izvedbama Linux operacijskog sustava. Osnovna distribucija uređaja zove se NOOBS koji sadrži šest različitih operacijskih sustava: Rasbian, RaspBMC, RISC OS, Arch, OpenELEC i Pidora. Pomoću NOOBS distribucije korisnici mogu brzo i jednostavno odabrati željeni operacijski sustav i instalirati ga na Raspberry Pi. Najčešće korišten i najrazvijeniji je Raspbian operacijski sustav. To je sustav koji je posebno razvijen za Raspberry Pi s ciljem iskorištavanja hardvera na što je bolje mogući način, a temelji se na Debian Wheezy operacijskom sustavu (https://www.raspberrypi.org).

[12] Izvor: https://learn.adafruit.com/assets/31833

4.3. Komunikacija

Komunikacija se dijeli na tri dijela. Prvi dio je komunikacija između Raspberry Pi i ultrazvučnih senzora, drugi dio je komunikacija Raspberry Pi uređaja sa serverom, dok je treći dio komunikacija server i Web aplikacije. U nastavku će detaljno biti objašnjena sva tri dijela komunikacije.

4.3.1. Komunikacija ultrazvučnog senzora i Raspberry Pi

Kako je prethodno objašnjeno u 4. poglavlju, podaci između senzora i Raspberry Pi-a putuju pomoću pinova Raspberry Pi, pinova ultrazvučnog senzora, spojnih žica, kondenzatora i eksperimentalne ploče. Podaci dobiveni iz ultrazvučnih senzora su mikrosekunde koje označavaju trajanje puta zvuka od predajnika do prepreke i natrag, stoga se moraju preračunati u udaljenost (u radu udaljenost je izražena u centimetrima).

```
resolver = dns.resolver.Resolver()
dns = resolver.query("Andrej", "A")

MAX_DISTANCE = 5
PORT = 1234
HOST = str(dns[0])
```

Kôd 4.1 Dohvat IP adrese po imenu računala

U Kôd 4.1 pomoću DNS Resolver biblioteke dohvaćaju se sve IP adrese koje imaju ime računala Andrej. Kasnije, nekoliko linija ispod dohvaća se prva spremljena adresa i sprema se u varijablu `HOST`, te se definira `PORT` koji mora biti jednak onom koji se nalazi na Web serveru.

Kako bi GPIO pinovi ispravno radili i čitali podatke sa senzora, potrebno ih je konfigurirati na način da se kaže koji pinovi su input, a koji output. Kod konfiguracije također je potrebno definirati i BCM opciju koja označava Broadcom SOC kanal[13].

```
GPIO.setmode(GPIO.BCM)
GPIO.setup(trig,GPIO.OUT)
GPIO.setup(echo,GPIO.IN)
```

[13] Brojevi GPIO pinova u zelenim pravokutnicima na Raspberry Pi-u koji služe za identifikaciju *trig* i *echo* pinova

Nakon što su senzori konfigurirani potrebno je mjeriti distancu od određenog objekta koja je prikazana u Kôd 4.2.

```
while GPIO.input(echo)==0:
      pulse_start = time.time()
while GPIO.input(echo)==1:
      pulse_end = time.time()

pulse_duration = pulse_end - pulse_start
distance = pulse_duration * 17150
distance = round(distance, 2)
GPIO.cleanup()
```

Kôd 4.2 Izračun udaljenosti senzora i objekta

U Kôd 4.2 sluša se ulazni pin koji je povezan s *echo* pinom. Senzor postavlja *echo* na visinu vremena koje je potrebno da signal ode i vrati se natrag, što bi značilo da je potrebno mjeriti vrijeme koje *echo* provede visoko. Zbog toga se koristi *while* petlja – njome se osigurava da je svaki vremenski zapis signala spremljen u pravilnom redoslijedu.

Dakle, prvi korak je detekcija vremena kada je *echo* posljednji put bio nisko (`puls_start`). Kada se dobije taj signal, vrijednost se mijenja od niskog (0) do visokog (1), a signal će ostati visok sve dok traje *echo* impuls. Stoga je potrebno i posljednje vrijeme kada je signal bio visoko (`pulse_end`).

Kada se dobije početak i kraj impulsa, izračunava se njihova razlika te se time dobije vrijeme trajanja impulsa (`pulse_duration`). Nakon toga se računa udaljenost objekta pomoću formule za izračun brzine koja kaže da je brzina jednaka udaljenosti podijeljenoj s vremenom. U našem slučaju, vrijeme je potrebno podijeliti s dva, pošto signal mora prijeći put do objekta i natrag. Brzina koju ovdje trebamo dobiti je brzina zvuka što na sobnoj temperaturi iznosi 343 m/s. Trenutno imamo formulu: 343 = udaljenost / (vrijeme / 2), što možemo prikazati kao udaljenost = 343 * vrijeme / 2. Kada brzinu zvuka pretvorimo u centimetre i podijelimo s dva dobijemo formulu vidljivu u Kôd 4.2 (`distance = pulse_duration * 17150`). Ukoliko je udaljenost između objekta manja od pet centimetara to znači da je parkirno mjesto zauzeto, u suprotnom, parkirno mjesto je slobodno.

4.3.2. Komunikacija Web servera i Raspberry Pi

Nakon što se dobiju podaci iz ultrazvučnih senzora i detektira se da li je parkirno mjesto slobodno ili ne, podaci se šalju na Web server. Komunikacija između servera i Raspberry Pi-a je Wi-Fi tehnologija, a način prijenosa poruke je putem spojne točke. U nastavku će biti prikazana implementacija slanja podataka s Raspberry Pi, te implementacija dohvata podataka u programskom jeziku Java.

```
def send(sensorId, status):
  sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
  sock.connect((HOST, PORT))
  sock.send(str(sensorId) + "\n")
  sock.send(status + "\n")
  sock.close()
```

Kôd 4.3 Slanje podataka putem spojne točke

Na početku se otvara spojna točka (engl. *socket*), te se definira na kojoj adresi će se spojiti i kojim portom će putovati poruka. Nakon toga šalju se podaci u definiranoj spojnoj točci. Raspberry Pi šalje identifikacijsku oznaku senzora, koja može biti od jedan do četiri, te šalje trenutni status parkirnog mjesta koje može biti „FREE“ ili „NOTFREE“.

Na Web serveru, prilikom njegova pokretanja kreira se instanca klase `ServerSocket` te se u konstruktor prosljeđuje `PORT` koji je jednak onom na Raspberry Pi

```
try(ServerSocket socket = new ServerSocket(PORT)) {
      socket.accept();
      smartParkingIsOnline = true;
      while(true) {
            try {
                  Socket client = socket.accept();
                  ExecutorService es
                  Executors.newFixedThreadPool(4);
                 es.execute(new ClientThread(client,server));
            } catch (IOException e) {
                  e.printStackTrace();
            }
} catch (IOException e) {
      e.printStackTrace();
}
```

Kôd 4.4 Dohvat podataka pomoću serverske spojne točke

Prvi `socket.accept()` označava da je maketa pametnog parkirališta na mreži i da sve funkcionira. Podatak o statusu parkirališta se šalje odmah prilikom pokretanja Python skripte na Raspberry Pi, te se tako označava da je maketa pokrenuta i da je parkiralište spremno za rad. Nakon toga server ulazi u beskonačnu petlju. Na taj način se osigurava da će server svaki puta čekati da dobije pristupnu točku s Raspberry Pi, te kada ju dobije, ponovo će biti spreman i čekati na novu. Kako bi server nesmetano radio, svaki puta kada dođe pristupna točka, kreira se nova nit u koju se prosljeđuje pristupna točka. Nakon što je nit kreirana, ona se pokreće, te se na taj način podaci čitaju iz pristupne točke (Kôd 4.5) i spremaju se u bazu podataka.

```
BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));

String parkingId = in.readLine();
String state = in.readLine();
```

Kôd 4.5 Čitanje podataka iz pristupne točke

4.3.3. Komunikacija Web servera i Web aplikacije

Web server i Web aplikacija u potpunosti su razdvojeni. Time se postiže da aplikacija može imati više klijenata (druge Web aplikacija, mobilne aplikacije, desktop aplikacije i slično), odnosno osigurava se interoperabilnost sustava. Komunikacija je izvršena pomoću Wi-Fi tehnologije i REST usluge koja prethodno objašnjena u poglavlju 3.4.1. Implementacija REST usluge je na Web serveru te ima metode za dohvaćanje trenutnog statusa parkinga, dohvaćanja povijesti promjena parkirnih mjesta za određeno vremensko razdoblje, dohvaćanje cijene pojedinog parkirnog mjesta po satu, ažuriranje cijene pojedinog parkirnog mjesta po satu i dohvaćanje statusa parkirališta koji govori da li je parkiralište spojeno na mrežu (Wi-Fi).

```
@GET
@Produces(MediaType.APPLICATION_JSON)
public String get() {
      List<Parking> parking = server.getParking();
      return gson.toJson(parking);
}
```

Kôd 4.6 HTTP GET metoda za dohvat trenutnog stastanja parkirališta

U Kôd 4.6 može se vidjeti primjer koji je korišten za dohvat trenutnog stanja na parkiralištu. Kako bi server znao koju metodu mora pozvati kada dođe zahtjev od strane klijenta, mora se definirati putanja koju klijent poziva i metoda koja mora biti okinuta. O tome koja će metoda biti pozvana definira se u zaglavlju zahtjeva (GET, POST, DELETE ili PUT). U ovom slučaju poziva se putanja `/parking`, server odlazi u klasu `ParkingService`, koja ima anotaciju `@Path("/parking")`, te je u zaglavlju definirana metoda GET, koja govori koja će se metoda pozvati. Prikaz zaglavlja može se vidjeti na Slika 4.9.

Slika 4.9 Prikaz zaglavlja prilikom zahtjeva na Web server

Kao što se može vidjeti na Slika 4.9, tip podataka je JSON, koji se prethodno definirao u kôdu dodavanjem anotacije (`@Produces(MediaType.APPLICATION_JSON)`) na metodu. JSON je tip podataka koji je razumljiv svim programskim jezicima, a ima izgled običnog teksta koji je napisan po određenim pravilima. Na Web serveru koji je razvijen u programskom jeziku Java, za serijalizaciju i deserijalizaciju JSON objekta koristi se Google-ova biblioteka Gson, dok na Web aplikaciji koji je razvijen u C# programskom jeziku se koristi biblioteka Newtonsoft.

```
RestClient client = new RestClient(HOST);
RestRequest request = new RestRequest("parking", Method.GET);

IRestResponse response = client.Execute(request);

string jsonContent = response.Content;
IEnumerable<ParkingViewModel> model =
JsonConvert.DeserializeObject<IEnumerable<ParkingViewModel>>(
jsonContent);
```

Kôd 4.7 Dohvat podataka sa servera na Web aplikaciju

U Kôd 4.7 prikazan je način dohvaćanja podataka s Web servera. Za slanje zahtjeva i primanje odgovora sa servera koristi se biblioteka RestSharp. Da bi se zahtjev mogao poslati potrebno je kreirati klijenta koji je u ovom slučaju Web server, odnosno korijen (engl. *root)* REST usluge na Web serveru. Tom klijentu potrebno je proslijediti u konstruktor putanju do korijena REST usluge koje je u ovom slučaju `http://localhost:8080/Rest`. Kada je kreiran REST klijent, potrebno je kreirati zahtjev kojim će se zatražiti željeni podaci. Kod kreiranja instance zahtjeva u konstruktor se prosljeđuju dva parametra. Kao što je vidljivo u Kôd 4.7 prvi parametar je putanja do određenog servisa, a drugi parametar je akcija koja se mora okinuti. Na temelju te akcije, Web server će znati koju metodu mora pozvati i koje podatke mora vratiti. Nakon što je zahtjev definiran može se izvršiti (`client.Execute(request)`). Kada se metoda izvršavanja završi, kao povratnu vrijednost vraća odgovor (engl. *Response*) kao objekt koji u sebi ima sadržaj. Sadržaj tog odgovora su željeni podaci dobiveni u JSON objektu. Nakon deserijalizacije tog objekta, podaci su spremni za vizualizaciju.

5. Pametno parkiralište – Analiza implementacije

U nastavku će biti opisana alternativna rješenja implementacije pametnog parkirališta te će biti napravljena metoda komparacije izrađene implementacije i svake opisane alternativne implementacije sustava pametnog parkirališta. Također će biti analizirana ostvarenost hipoteze u poglavlju 5.2.

5.1. Alternativne implementacije

- **Pametno parkiralište korištenjem Raspberry Pi kamera, Raspberry Pi uređaja i Wi-Fi tehnologije**

Implementacija pametnog parkirališta korištenjem kamera moguća je na način da se na svako parkiralište postavi nekoliko stupova te na svaki stup kamera. Supovi moraju biti dovoljno visoki da bi kamera mogla imati u kadru cijelo parkiralište. Razlog većeg broja kamera na jednom parkiralištu je bolja točnost i veća preciznost određivanja objekta na parkirnom mjestu. Moguće je na više načina detektirati da li je parkirno mjesto slobodno ili zauzeto.

Jedan od načina je detekcija bridova objekta sa slike. Taj način je vrlo kompleksan. Kamera bi svakih nekoliko sekundi morala slikati sliku i slati je na Raspberry Pi koji bi unutar jedne sekunde trebao odrediti da li je parkirno mjesto zauzeto ili ne. Kompleksnost ovog načina implementacije je brzina detekcije zauzeća parkirnog mjesta. Prvo je potrebno razviti algoritam koji će iz slike detektirati stanje svih parkirnih mjesta na slici te za svako parkirno mjesto odlučiti da li je objekt automobil ili ne. Nakon toga je potreban algoritam koji će uspoređivati podatke dobivene sa svih kamera i generirati konačno stanje parkinga.

Drugi način je detekcija određenog znaka na parkirnom mjestu. Takva vrsta implementacije pratila bi znakove na samim parkirnim mjestima. Primjerice, svako parkirno mjesto imalo bi veliko slovo P ispisano na samom asfaltu. To slovo bilo bi određene boje (na primjer narančaste) koja bi se lako mogla detektirati sa slike kamere. Dakle, kamera bi slikala svakih nekoliko sekundi te bi sliku slala na Raspberry Pi koji bi provjerio koje parkirno mjesto ima vidljiv znak, a koje nema. Nakon obrade podataka i usporedbe sa svim drugim kamerama, dobilo bi se konačno stanje parkirališta.

U usporedbi s implementacijom pametnog parkirališta korištenjem Interneta stvari i Wi-Fi tehnologije, kod opisane alternative je puno veća kompleksnost razvoja sustava. U prvom

načinu detekcije bridova na slici, potrebna je velika brzina rada algoritama, te mogućnosti krive procjene. U oba slučaja veliku ulogu igraju vremenske neprilike. Na primjer magla bi onemogućila kamerama da snime čistu sliku, te bi algoritam mogao pogriješiti kod detekcije slobodnog/zauzetog parkirnog mjesta. Također problem predstavljaju stupovi (koji bi morali biti instalirani ukoliko ne postoje) koji bi morali imati pristup električnoj energiji kako bi Raspberry Pi i kamera mogli funkcionirati. Kod drugog načina detekcije određenog znaka na parkirnom mjestu, veliki nedostatak je snijeg koji prekrije čitav znak i algoritam detektira zauzeto parkirno mjesto iako je slobodno. U drugu stranu, implementacija pametnog parkirališta korištenjem Interneta stvari i Wi-Fi tehnologije morala bi na svakom parkirnom mjestu imati instalirane senzore koji bi putem žice morali biti spojeni do Raspberry Pi uređaja što je puno žica za jedno parkiralište. Nakon analize može se doći do zaključka da je veća isplativost i korisnost pametnog parkirališta korištenjem Interneta stvari i Wi-Fi tehnologije nego opisana alternativa.

- **Pametno parkiralište korištenjem Interneta stvari, Bluetooth tehnologije i Wi-Fi tehnologije**

Implementacija pametnog parkirališta korištenjem Bluetooth tehnologije bila bi izvedena na način da se na svako parkirno mjesto ugradi senzor koji detektira da li je parkirno mjesto slobodno ili zauzeto te šalje podatke putem Bluetooth-a direktno na uređaj. Raspberry Pi uređaj bi bio smješten na određenom mjestu na parkiralištu, gdje bi bio u dometu senzora. Pošto Raspberry Pi podržava Bluetooth – komunikacija bi tekla bez smetnji. Kada bi senzor poslao stanje parkirnog mjesta, Raspberry Pi bi podatke obradio, te poslao putem Wi-Fi tehnologije na server.

U usporedbi s implementacijom pametnog parkirališta korištenjem Interneta stvari i Wi-Fi tehnologije, opisana alternativa i njena kompleksnost razvoja je vrlo slična implementiranoj implementaciji. Jedini problem kod ove alternative predstavlja izvor električne energije za svaki senzor, dok se puno više isplati nego implementacija pametnog parkirališta korištenjem Interneta stvari i Wi-Fi tehnologije iz razloga što je puno jednostavnija za instalaciju i integraciju. Žice se skoro ne koriste. Ukoliko bi se implementacija odrađena u praktičnom radu instalirala na parkiralište od primjerice 200 parkirnih mjesta, uz veliki broj žica i senzora, postojala bi potreba i za više uređaja Raspberry Pi iz razloga što jedan uređaj ne može napojiti električnom energijom sve senzore (ovisno je o napajanju). S druge strane, opisana alternativa bi također imala potrebu za više uređaja, kod nje bi ovisilo o dometu, odnosno udaljenosti između uređaja i pojedinog parkirnog mjesta.

- **Pametno parkiralište korištenjem LoRaWAN tehnologije i Internet stvari**

LoRa je fizički sloj koji stvara komunikaciju na vrlo velikom dometu. Komunikacija se temelji na mreži širokog područja (engl. *Wide Area Network*) te služi u svrhu komunikacije za M2M uređaje ili za Internet stvari, odnosno za uređaje male potrošnje (Henriksson, 2016).

LoRaWAN mreža ima mogućnost da s jednom antenom pokrije veliko područje, pa čak i na nepristupačnim lokacijama i lokacijama koje imaju loše izvore električne energije. Pored velikog dometa i niske potrošnje električne energije, LoRaWAN ima i duboku pokrivenost unutar objekata, ali i vrlo je velik stupanj sigurnosti podataka (Henriksson, 2016).

Implementacija pametnog parkirališta pomoću LoRaWAN tehnologije bila bi izvedena na način da bi svako parkirno mjesto imalo ugrađeno LoRa senzor koji podržava LoRa mrežu. Senzor bi napajala baterija. Zbog niske potrošnje električne energije senzora, baterija bi imala vijek trajanja do čak 15 godina. Senzori bi bili ugrađeni u asfaltu parkirnog mjesta te bi detektirali stanje tog mjesta (da li je iznad senzora automobil ili ne). Podatke bi pomoću LoRa poveznika (engl. *gateway*) senzor proslijedio na Raspberry Pi koji bi podatke obradio i slao prema serveru. Prikaz načina komunikacije i izgleda parkirališta korištenjem LoRaWAN tehnologije može se vidjeti na Slika 5.1.

Slika 5.1 Komunikacija pametnog parkirališta korištenjem LoRaWAN tehnologije

Posljednja alternativa je najisplativija alternativa. Parkirna mjesta ne moraju imati nikakav izbor električne energije zbog njihovih LoRa senzora niske potrošnje koje može napojiti baterija. Kada bi uzeli za primjer parkirna mjesta jednog grada, recimo Opatije. Zbog velike

pokrivenosti LoRaWAN mreže, samo Jedan LoRa poveznik bio bi dovoljan da sva parkirna mjesta mogu komunicirati sa samo jednim Raspberry Pi uređajem. Što je bolja opcija u odnosu na implementaciju koja je obrađena na praktičnom radu kod koje bi svako parkiralište u Opatiji moralo imati barem nekoliko (najmanje jedan) Wi-Fi pristupnik i barem nekoliko (najmanje jedan) Raspberry Pi uređaja, te dodatno razne instalacije i provodnike za žice koje bi spajale senzore i Raspberry Pi. Također, svako parkiralište moralo bi imat izvor električne energije i pristup toj električnoj energiji kako i se uređaji mogli spojiti na tu energiju, te je proslijediti do senzora.

5.2. Analiza ostvarenosti hipoteze

Hipoteza: Integracija i instalacija pametnog parkirališta korištenjem Interneta stvari i Wi-Fi tehnologije je isplativa.

Temeljem rada i metoda komparacije možemo zaključiti da je postavljena hipoteza opovrgnuta. Zbog kompleksnosti i vrlo visokih troškova, instalacija i integracija pametnog parkirališta korištenjem Interneta stvari i Wi-Fi tehnologije nije isplativa.

Za integraciju i instalaciju parkirališta koje ima kapacitet za 200 parkirnih mjesta bilo bi potrebno minimalno 10 Wi-Fi poveznika, kako bi uređaji nesmetano komunicirali. Potrebno je minimalno 67 Raspberry Pi uređaja (jedan Raspberry Pi na 3 parkirna mjesta) zbog ne mogućnosti napajanja svih senzora. Svako parkirno mjesto moralo bi imati minimalno četiri ultrazvučna senzora, što bi ukupno bilo 800 ultrazvučnih senzora, te bi poveznica između Raspberry Pi uređaja i senzora bila žica, što bi značilo da su potrebne dodatne instalacije za žice. Također, potrebno je sigurno kućište za svaki senzor, ali i za Raspberry Pi uređaj, koji mora imati pristup električnoj energiji. Potrebno je osiguranje električne energije i u slučaju nestanka električne energije, kako bi sustav mogao funkcionirati u svim uvjetima.

Stoga, možemo izračunati da 67 Raspberry Pi uređaja iznosi 17.940,00 HRK, 800 senzora iznosi 5.040,00 HRK. Žice potrebne za spojit 800 senzora iznose 5.000,00 HRK. Pojedino sigurno kućište za Raspberry Pi iznosi 80,00 HRK, što je ukupno 5.360,00 HRK, te iznos kućišta za 800 senzora iznosi 8.000,00 HRK. Postavljanje instalacija iznosi oko 5.000,00 HRK. Ukupna cijena same instalacije jednog parkirališta iznosi minimalno 46.340,00 HRK. Dodatni troškovi mogu nastati ukoliko parkiralište nema izvor električne energije ili ukoliko nije moguće pristupiti Wi-Fi mreži te su potrebni dodatni radovi kako bi se problem riješio. U ukupnu cijenu iznad nisu navedeni mjesečni troškovi potrošenog Internet prometa,

troškovi električne energije, troškovi postavljanja sustava kao ni troškovi održavanja sustava. Troškovi mogu narasti vrlo visoko, ali isto tako postoji rizik da je neko od parkirališta na nepristupačnoj lokaciji ili lokaciji koja ima slab izvor električne energije.

Alternativa implementacije pametnog parkirališta korištenjem LoRaWAN tehnologije i Internet stvar se pokazala kao dobra opcija. Za grad veličine Opatije potrebno je LoRaWAN senzora i baterija koji će napajati senzore u onom broju koliko Opatija ima parkirnih mjesta, jedan LoRaWAN poveznik i jedan Raspberry Pi uređaj. Senzori su ugrađeni u asfalt samog parkirnog mjesta, stoga ne trebaju nikakva dodatna kućišta. Poveznik može biti na nekom centralnom mjestu Opatije, gdje će biti smješten u posebno sigurno kućište, u kojem će biti i Raspberry Pi uređaj. Raspberry Pi može i ne mora biti povezan na Wi-Fi mrežu. Ukoliko se koristi LoRaWAN oblak, a u oblaku je server – Wi-Fi neće biti potreban, u suprotnom, potreban je jedan Wi-Fi poveznik ili USB Wi-Fi Internet koji je dostupan vrlo jeftino.

Senzori koji podržavaju LoRaWAN tehnologiju mogu biti vrlo skupi, što je nedostatak ove alternative. Razlog tomu je taj što je LoRa relativno nova tehnologija i još uvijek ne postoji velik broj proizvođača senzora takve vrste.

Alternativa implementacije pametnog parkirališta Bluetooth tehnologije, Internet stvari i Wi-Fi tehnologije se također pokazala kao dobra opcija. Razvoj sustava bi bio vrlo sličan implementaciji obrađenoj u radu, odnosno razvoj bi bio relativno jednostavan. Za razliku od implementacije pametnog parkirališta implementirane na praktičnom radu, ne bi postojala potreba za dodatnim instalacijama pošto ne postoji potreba za žicama, što bi rezultiralo manjim troškovima.

Zaključak

Prema gore navedenoj problematici vozači imaju potrebu za sustavom koji će im omogućiti informacije o stanju na parkiralištu u stvarnom vremenu. Time bi mogli optimizirati svoj raspored putovanja bez uzaludnog traženja parkirnog mjesta. Pametno parkiralište omogućilo bi svim vozačima da bez problema pronađu slobodno parkirno mjesto na bilo kojoj njima odgovarajućoj lokaciji putem informacijske i komunikacijske tehnologije. Pomoću pametnih telefona korisnici bi se mogli povezivati na Internet i na taj način dobiti potrebne informacije. Svrha implementacije pametnog parkirališta je smanjiti gužve, koristiti javni prijevoz i detektirati nepropisno parkiranje.

Pošto je rezultat hipoteze opovrgnut, a postoji potreba za sustavom pametnog parkirališta do izražaja dolaze alternative kao što su pametno parkiralište korištenjem Interneta stvari, Bluetooth tehnologije i Wi-Fi tehnologije i pametno parkiralište korištenjem LoRaWAN tehnologije i Internet stvari.

Odabir između te dvije alternativne implementacije ovisio bi o dugoročnoj isplativosti i jednostavnosti održavanja i nadogradnje sustava. Prilikom uvođenja sustava, LoRaWAN tehnologija bi bila najskuplji odabir uz najjednostavnije i najbrže uvođenje sustava u rad, međutim, ta implementacija bi kroz dugoročno vrijeme bila najjeftinija opcija i daleko najjednostavnija za održavanje i nadogradnju sustava. Kvarovi bi se vrlo lako detektirali te vi se brzo i jednostavno ti kvarovi uklanjali.

Stoga, dolazimo do zaključka da je najbolja opcija za uvođenje ovakvog sustava korištenje LoRaWAN tehnologije i Internet stvari, te da su druge opcije moguće za realizaciju ali ne i dovoljno kvalitetne da bi bile implementirane u cijelom gradu ili državi.

Popis kratica

HTTP	*Hypertext Transfer Protocol*	protokol prijenosa informacija
3D	*Three-dimensional*	trodimenzionalno
XML	*Extensible Markup Language*	označni jezik
URI	*Uniform Resource Identifier*	jedinstveni identifikator resursa
W3C	*World Wide Web Consortium*	organizacija koja se bavi standardizacijom tehnologija korištenih na Webu
IETF	*Internet Engineering Task Force*	organizacija posvećena razvoju i promoviranju internetskih standarda i protokola
TCP	*Transmission Control Protocol*	protokol kontrole prijenosa
GPIO	*General – Purpose Input/Output*	opće – namjenski ulaz/izlaz
MVC	*Model View Controller*	model prikaz kontroler
NoSQL	*Non Structured Query Language*	ne strukturirani jezik upita
M2M	*Machnine to Machine*	stroj na stroj
USB	*Universal Serial Bus*	univerzalna serijska sabirnica
HDMI	*High-Definition Multimedia Interface*	multimedijalni međusklop visoke definicije
CSI	*Camera Serial Interface*	serijsko sučelje kamere
LAN	*Local Area Network*	lokalna mreža
DSI	*Display Serial Interface*	serijsko sučelje za prikaz
NOOBS	*New Out Of Box Software*	programski startni paket za početnike
DNS	*Domain Name System*	domenski sustav imena
JSON	*JavaScript Object Notation*	oznaka JavaScript objekta
LoRa	*Low Power*	niska potrošnja električne energije
WAN	*Wide Area Network*	mreža širokog područja

Popis slika

Popis tablica

Popis kodova

Literatura

[1] Abdelgawad A., Yelamarthi K. Internet of Things (IoT) Platform for Structure Health Monitoring, *Communication and Mobile Computing, 1* (2017), 10. stranica, http://ieeexplore.ieee.org/document/7870118/?reload=true, kolovoz 2017.

[2] Atzoria L., Iera, A., Morabito, G. Internet of Things: A survey, *Computer Networks, 54*, 15, (2010), 2787 – 2805.

[3] Bahga A., Madisetti V. *Internet of Things: A Hands-On Approach*. Vijay Madisetti, 2016.

[4] Basavaraju R. Automatic Smart Parking System using Internet of Things (IOT), *International Juornal of Scientific and Reseatch Publications, 5,* 12 (2015), 629 – 632. http://www.ijsrp.org/research-paper-1215/ijsrp-p4898.pdf, kolovoz 2017.

[5] Belak, S. *Uvod u znanosti*. Šibenik: Visoka škola za turistički menadžment u Šibeniku, 2006.

[6] Botta A., et al. Integration of Cloud computing and Internet of Things: A survey. *Future Generation Computer Systems, 5,* (2016), 684 – 700.

[7] Carullo A., Parvis M. An ultrasonic sensor for distance measurement in automotive applications, *IEEE Sensors Journal, 1,* 2 (2001), 143 – 147. https://www.researchgate.net/publication/3430936_An_ultrasonic_sensor_for_distance_measurement_in_automotive_applications, kolovoz 2017.

[8] Dongarsane C.R. et al. Green House Automation Using IoT, *International Research Journal of Engineering and Technology, 4,* 1 (2017), 1485 – 1490.

[9] Evans D. The Internet of Things: How the Next Evolution of the Internet Is Changing Everything. *White Paper*, Cisco Internet Business Solutions Group, 2011., https://www.cisco.com/c/dam/en_us/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf, kolovoz 2017.

[10] Grgić S., Kunštek Z. *Uvod u računalne mreže: priručnik,* Zagreb, Algebra, 2010

[11] Henriksson R. Indoor positioning in LoRaWAN networks. *Diplomski rad.* Chalmers University of Technology, 2016., http://publications.lib.chalmers.se/records/fulltext/241190/241190.pdf, kolovoz 2017.

[12] Jeffrey J. Smart Parking System using Wireless Sensor Networks. *The Sixth International Conference on Sensor Technologies and Applications,* Rome, 2012.

[13] Ljubi I., Mihaljević B. *Interoperabilnost informacijskih sustava: priručnik*. Zagreb, Algebra, 2013.

[14] Nguyen Gia T. et al. IoT – Based Fall Detection System with Energy Efficient Sensor Nodes. *Nordic Circuits and Systems Conference,* Copenhagen, (2016).

[15] Shoup D.C. Crusing for parking, *Transport Policy, 13,* 6 (2006), 479 – 486.

[16] Wallace S., Richardson M. *Getting Started With Raspberry Pi: An Introduction to the Fastest-Selling Computer in the World*. Sebastopol: Maker Media, 2016.

[17] http://datasmart.ash.harvard.edu/news/article/how-smart-city-barcelona-brought-the-internet-of-things-to-life-789, kolovoz 2017.

[18] https://learn.adafruit.com/assets/31833, kolovoz 2017.

[19] https://nvisium.com/blog/2015/05/27/implementing-coap-secure-way-part-i/, kolovoz 2017.

[20] https://www.hackster.io/bucknalla/mqtt-micropython-044e77, kolovoz 2017.

[21] https://www.raspberrypi.org, kolovoz 2017.