Primjena IoT uređaja u izradi rješenja za parkirne prostore

Blagajčević, Mirza

Undergraduate thesis / Završni rad

2024

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Organization and Informatics / Sveučilište u Zagrebu, Fakultet organizacije i informatike**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:211:221664

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Mirza Blagajčević

PRIMJENA IOT UREĐAJA U IZRADI RJEŠENJA ZA PARKIRNE PROSTORE

ZAVRŠNI RAD

Varaždin, 2024.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Mirza Blagajčević

Matični broj: 0016155053

Studij: Informacijski i poslovni sustavi

PRIMJENA IOT UREĐAJA U IZRADI RJEŠENJA ZA PARKIRNE PROSTORE

ZAVRŠNI RAD

Mentor/Mentorica:

Doc. dr. sc. Marko Mijač

Varaždin, rujan 2024.

Mirza Blagajčević

Izjava o izvornosti

Izjavljujem da je moj završni rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor potvrdio prihvaćanjem odredbi u sustavu FOI-radovi

Sažetak

Ovaj završni rad istražuje primjenu IoT tehnologija u domeni upravljanja parkirnim prostorima, s ciljem optimizacije i pojednostavljenja korištenja parkirnih mjesta. Fokus je na izradi IoT senzora i sustava koji omogućuju bolju organizaciju parkirnih kapaciteta, uključujući detekciju zauzetosti i optičko prepoznavanje registracijskih oznaka vozila kako bi se moglo locirati konkretno vozilo unutar parkirnog prostora.

U sklopu rada izrađen je prototip klijentske aplikacije koja prikuplja podatke s parkirnih senzora, omogućavajući detaljniji uvid u zauzetost parkirnih prostora. Također, rad analizira različite komunikacijske protokole te objašnjava zašto je odabrani protokol idealan za ovu vrstu sustava. Ovaj sustav omogućuje praćenje u stvarnom vremenu, rezervaciju parkirnog mjesta i brojne druge funkcionalnosti, čime pridonosi modernizaciji upravljanja parkirnim prostorima.

Ključne riječi: razvoj softvera, internet stvari (IoT), automatizacija, parkiranje, optičko prepoznavanje znakova

Sadržaj

1. Uvod

U današnjem svijetu, jedno od ključnih pitanja s kojima se gradovi suočavaju je upravljanje prometom i parkirnim prostorima. S rastućim brojem vozila, pronalaženje slobodnog parkirnog mjesta često predstavlja izazov za vozače, dok pretrpanost parkirališta može uzrokovati dodatne prometne gužve i povećanu emisiju štetnih plinova. Ova situacija zahtijeva inovativna rješenja koja će omogućiti efikasnije korištenje dostupnih parkirnih mjesta.

Tehnologija interneta stvari (*skraćeno: IoT*) postaje sve važnija u digitalnoj transformaciji raznih sektora, uključujući prometa i infrastrukture. IoT predstavlja mrežu povezanih uređaja koji međusobno komuniciraju putem interneta, razmjenjuju podatke i omogućuju upravljanje na daljinu. U kontekstu parkirnih prostora, IoT tehnologija može značajno unaprijediti upravljanje parkiranjem pomoću pametnih sustava koji u stvarnom vremenu prate zauzetost parkirnih mjesta, automatski obavještavaju vozače o slobodnim mjestima, te omogućuju optimizaciju parkiranja kroz napredne funkcije poput rezervacije parkirnih mjesta ili automatske naplate. Pametni parkirni sustavi također omogućuju vozačima smanjenje vremena provedenog u traženju parkirnog mjesta, što izravno utječe na smanjenje prometnih gužvi i emisije štetnih plinova.

Cilj ovog rada je razvoj i implementacija sustava za upravljanje parkirnim prostorima uz korištenje IoT tehnologije, s naglaskom na detekciju zauzetosti parkirnih mjesta i identifikaciju registarskih oznaka vozila. Ključni dio rješenje je senzor temeljen na Raspberry Pi platformi, koji omogućuje prikupljanje podataka o stanju parkirnih mjesta putem kamere i optičkog prepoznavanja vozila. Rješenje uključuje razvoj softvera koji će korisnicima omogućiti praćenje zauzetosti parkirnih mjesta, identificiranje parkiranih vozila i rezervaciju parkirnog mjesta putem klijentske aplikacije.

Primjena IoT tehnologija u upravljanju parkirnim prostorima omogućuje stvaranje pametnih gradova, gdje će upravitelji infrastrukture imati bolji uvid u korištenje parkirnih kapaciteta, a vozači će imati pristup informacijama o slobodnim parkirnim mjestima u stvarnom vremenu. Ovaj pristup ne samo da smanjuje potrebu za dodatnom gradnjom novih parkirnih kapaciteta, već i omogućuje bolje iskorištavanje postojećih resursa, što rezultira smanjenjem troškova i povećanjem efikasnosti. U kontekstu urbanizacije i sve veće potrebe za održivim razvojem, ovakva rješenja predstavljaju važan korak prema ekološki prihvatljivijem prometu.

2. Internet stvari

Internet stvari se može definirati kao mreža fizičkih uređaja koji su opremljeni sustavom za međusobnu komunikaciju kao i senzorima za prikupljanje informacija te mogućnosti autonomnog odlučivanja i djelovanja na temelju prikupljenih informacija. “Internet stvari (engl. Internet of Things - IoT) se može definirati kao međusobno povezivanje nekoliko na jedinstven način prepoznatljivih, sveprisutnih i internetom povezanih stvari koje su opremljene senzorskim/aktuatorskim i komunikacijskim jedinicama te su sposobne za inteligentno donošenje odluka. Uređaji su također interoperabilni [1]i mogu se samostalno konfigurirati i programirati“ prema IEEE. Iako prethodna definicija vrlo dobro opisuje što je to internet stvari, važno je napomenuti kako su „stvari“ u samom pojmu „internet stvari“ vrlo slabo definirane. Prema nekim definicijama svaka „stvar“ koja koristi internet kao primarno sredstvo povezivanja i izmjenjivanja informacija se može definirati pod pojmom interneta stvari [1]. Iako je ta definicija točna, možda bi bilo ispravnije reći da se internetom stvari smatra uređaj koji je napravljen od specijaliziranog hardvera koji će biti vrlo ograničen u aspektu performansi što će pridonijeti vrlo pristupačnoj cijeni i maloj potrošnji električne energije. Međutim taj uređaj će imati dovoljne performanse kako bi odradio specifični zadatak za koji je osmišljen i napravljen.

2.1. Razvoj Interneta stvari

Iako Kevin Ashton spominje sam pojam „Internet stvari“ tek 1999. godine tijekom sudjelovanja na projektu optimizacije opskrbnog lanca američkog proizvođača kućnih potrepština Procter & Gamble prema [1]. Važno je napomenuti kako se prvi oblik IoT uređaj može prepoznati u samoslužnom automatu Coca-Cola proizvoda napravljen čak u ranih 1980.-ih godina. Automat je imao mogućnost pratiti trenutnu napunjenost pića u automatu i temperaturu unutar uređaja. Naime, nije se radilo o uređaju za široku primjenu već o pokušaju povezivanja svakodnevnog uređaja na mrežu u sklopu eksperimenta na jednom američkom fakultetu. Nakon toga je u 21. stoljeću IoT doživio veliku popularizaciju zbog napretka tehnologija ključnih za IoT poput: bežičnih mreža, mikro kontrolera, senzora, računalstva u oblaku i sl. IoT postaje sve pristupniji i korisniji u svim aspektima ljudskog života jer značajno pojednostavljuje svakodnevni dan čovjeku.

[1] Sposobnost uređaja za međusobno pružanje i primanje informacija od drugih uređaja

2.2. Tehnologije ključne za IoT

Internet stvari se oslanja na veliki niz drugih tehnologija koje omogućuju njegovo nesmetano korištenje i primjenu kakvu danas znamo. Ove tehnologije uključuju, ali nisu ograničene na, specijalizirani hardver, razne softverske pristupe i platforme, te inovativne komunikacijske protokole.

2.2.1. Važnost namjenskoga hardvera

Iako se pri izradi osobnog računala često pretpostavlja da su komponente s višim performansama uvijek bolji izbor, to može biti opravdano jer će računalo služiti za obavljanje različitih zadataka i pokretanje brojnih programa. Međutim kod izrade IoT uređaja moramo uzeti u potpunosti različiti pristup kako bi taj uređaj imao smisla.

Kod odabira hardvera potrebnog za izvršavanje nekog zadatka IoT uređaja uzimamo u obzir kako sami hardver mora biti sposoban zadovoljiti minimalne uvjete softvera. Međutim ne smije imati neiskorištene performanse hardvera kako se ne bi uzaludno trošili resursi pri izradi uređaji i potrošnja električne energije istoga. Vrlo često se kod izrade takvih uređaja koriste specijalizirani mikro kontroleri napravljeni specifično za takve namjene, poput: ESP32, ESP8266, Raspberry Pi i sl. Prethodno navedeni mikro kontroleri su ipak više orijentirani „kućnom“ korisniku koji nema sredstva i potrebe za izradu u potpunosti specijaliziranog hardvera. Dok će kod svake ozbiljnije izrade IoT uređaja za masovnu prodaju biti izrađen od specijaliziranog hardvera namijenjenog isključivo za primjenu u konkretnoj domeni.

Ključna komponenta IoT uređaja su i senzori, koji omogućuju prikupljanje podataka iz okoline. Primjeri uključuju senzore temperature, vlage, svjetlosti, pokreta i mnoge druge. Aktuatori, s druge strane, omogućuju uređajima da izvršavaju akcije na temelju prikupljenih podataka, poput otvaranja ventila, pokretanja motora ili paljenja svjetla.

2.2.2. Softver

Naravno, uz hardver, softver je ključna komponenta IoT uređaja jer ona definira samo ponašanje uređaja. IoT operativni sustavi, kao što su FreeRTOS, Contiki, i Zephyr, optimizirani su za rad na uređajima s ograničenim resursima. Osim operativnih sustava, postoje i softverske platforme poput ThingSpeak, AWS IoT, i Azure IoT koje omogućuju razvoj, upravljanje i analizu IoT aplikacija. Problematična strana razvoja softvera za IoT uređaje je upravo ta vrlo velika ograničenost hardvera kako bi se maksimalno smanjili troškovi proizvodnje i troškovi električne energije tj. dulje vrijeme trajanje akumulatora uređaja ukoliko

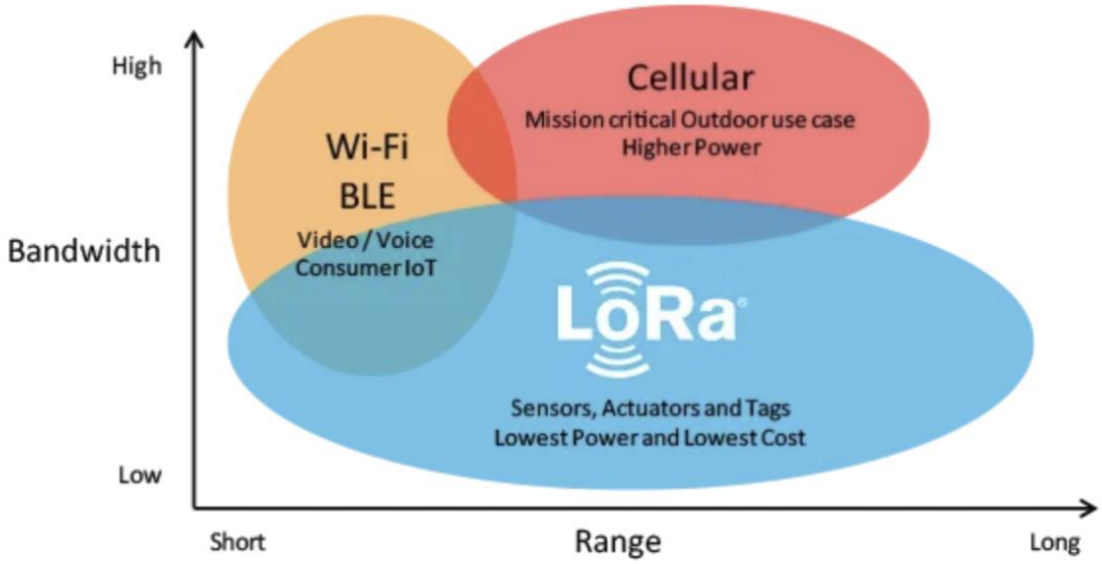
se radi o bežičnom uređaju. Također, IoT uređaji generiraju ogromne količine podataka koji trebaju biti prikupljeni, pohranjeni i analizirani. IoT platforme omogućuju oblak i rubno računalstvo (*engl. edge computing*) kako bi se osigurao učinkovit prijenos podataka, analiza u stvarnom vremenu i donošenje odluka na temelju tih rezultata obrade podataka.

2.2.3.Povezivanje uređaja

Povezanost i sveprisutnost su jedne od ključnih svojstva IoT uređaja jer su oni sami po sebi „beskorisni“, tj. njihova uloga je biti manji dio puno većeg sustava koji koristi podatke dobivene od strane IoT uređaja. Povezanost samih uređaja s centralnim serverom i/ili klijentskom aplikacijom možemo postići raznim hardverskim i softverskim protokolima. Protokoli mogu biti opći poput HTTP-a, webSocket-a i sl. koji nisu nužno napravljeni za primjenu u IoT-u, ali su i dalje vrlo popularni, pa do protokola poput MQTT-a, ZigBee-a čija je primjena gotovo isključivo u IoT-u. Odabir protokola kojeg ćemo koristiti za prijenos podataka je usko povezan s hardverskim i softverskim ograničenjima sustava unutar kojeg se koristi. Neki od zastupljenijih protokola i tehnologija, kao i važnost odabira istih ovisno o ograničenjima sustava će biti daljnje razrađena u poglavlju.

2.2.3.1.LoRa i LoRaWAN

LoRa (Long Range) je napredna tehnologija bežičnog prijenosa podataka koja koristi Chirp Spread Spectrum (CSS) modulaciju za efikasnu komunikaciju na velikim udaljenostima. Ova tehnologija omogućava prijenos podataka na udaljenostima do nekoliko kilometara u urbanim mjestima, dok u ruralnim područjima domet može doseći i desetke kilometara. Ključna prednost LoRa tehnologije je iznimno niska potrošnja energije, što omogućava dugotrajno korištenje baterija, što je posebno važno za IoT uređaje i senzore koji se nalaze u teško dostupnim ili udaljenim područjima. LoRa je dizajnirana za aplikacije koje ne zahtijevaju visoku brzinu prijenosa podataka, već stabilan i dug domet pri niskim brzinama, što je dovoljno za većinu IoT aplikacija koje šalju male količine podataka u redovitim intervalima. Tehnologija koristi napredne metode enkodiranja za osiguranje integriteta i sigurnosti komunikacije. LoRa se široko koristi u različitim IoT aplikacijama, uključujući pametne gradove, poljoprivredu, praćenje okoliša i pametne mjerne uređaje, gdje dugotrajna baterija i veliki dometi igraju ključnu ulogu.



Slika 1 – Usporedba LoRa-e s alternativnim načinima prijenosa podataka [3]

LoRaWAN (engl. Long Range Wide Area Network) je komunikacijski protokol temeljen na prethodno spomenutoj LoRa tehnologiji, te on zapravo komunikacijski protokol kojeg možemo koristiti na samim IoT uređajima. Neke od prednosti LoRaWAN-a su:

- vrlo niska potrošnja električne energije
- veliki domet signala zbog male frekvencije odašiljanja
- velika sigurnost kod prijenosa podataka zbog AES-128 enkripcije

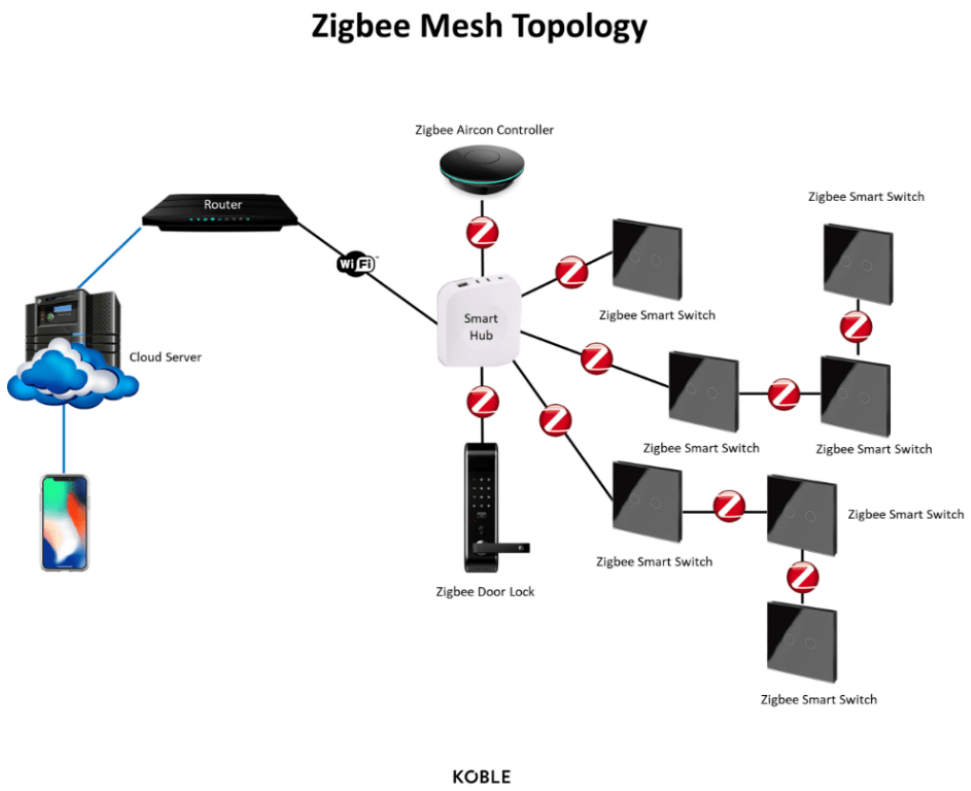
upravo je iz tih razloga LoRa, tj. LoRaWAN idealan za primjenu kod IoT uređaja kao što su senzori jer oni zahtijevaju vrlo malu propusnost podataka i zbog male potrošnje im može dugo trajati baterija.

2.2.3.2. ZigBee

ZigBee je bežični protokol namijenjen primjeni na IoT uređajima iz vrlo sličnih razloga kao što je i LoRaWAN, a to su: mala potrošnja električne energije i sigurnost. Međutim za razliku od LoRaWAN-a ZigBee ima jednu vrlo veliku prednost – mrežnu topologiju. Naime, prema [4], ZigBee protokol omogućuje svakom IoT uređaju koji je dio neke mreže da se ponaša kao usmjerivač[2] za drugi uređaj u mreži, tj. svaki uređaj može biti usmjerivač, što znači da nema potrebe za centralnim usmjerivačem ili posebnom infrastrukturom. Upravo je iz tih

[2] uređaj koji usmjerava podatkovne pakete na njihovom putu kroz mrežu

razloga, ali i vrlo pristupačne cijene, ZigBee protokol vrlo popularan u IoT rješenjima kao što su: pametne kuće, industrijski pogoni, poljoprivredna područja i sl.



Slika 2 – Topologija ZigBee mreže

(izvor: https://koble.sg/4-reasons-zigbee-home-automation/, preuzeto 2024.)

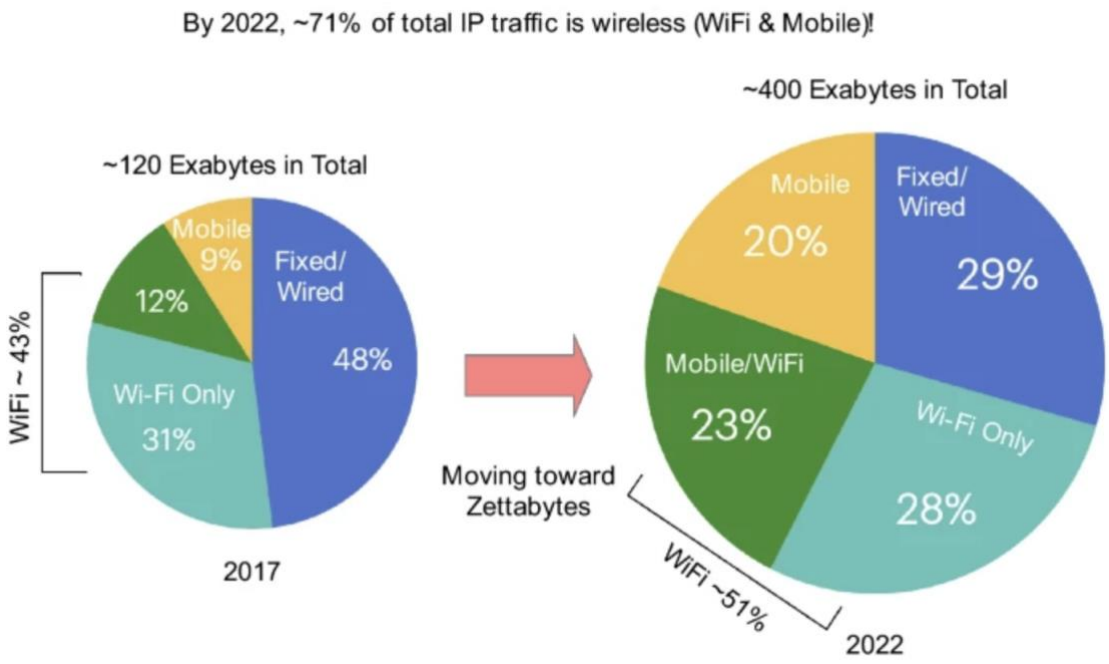
Na slici gore je prikazana mrežna topologija ZigBee-a gdje je zapravo svaki IoT uređaj koji je dio mreže također može komunicirati s drugim uređajima u mreži, a ne isključivo s centralnim upravljačem. To nam omogućuje i redundantnost u slučaju da jedan uređaj postane nedostupan ostali uređaji mogu ostvariti komunikaciju s centralnim upravljačem koristeći druge dostupne uređaje na mreži.

2.2.3.3. Wi-Fi

Wi-Fi (Wireless Fidelity) je svima poznat pojam za kojeg možemo reći da je postao sinonim za sami internet. Međutim Wi-Fi ili WLAN (*Wireless Local Area Network*) je tehnologija koja nam pruža bežičan pristup lokalnoj mreži uređaja.

Temeljen na IEEE 802.11 standardu osnovanog 1997. godine s maksimalnom propusnosti od samo 1-2 Mb/s Wi-Fi je u potpunosti promijenio način na koji pristupanja internetu i fleksibilnosti uređaja s kojih možemo pristupiti. Za razliku od klasičnog (žičanog)

pristupa lokalnoj mreži Wi-Fi je omogućio, do tad neviđenu, fleksibilnost što je doprinijelo masovnom korištenju Wi-Fi-a za pristup mreži. Čak je toliko popularan da danas nije rijetkost vidjeti uređaj koji je napravljen isključivo za pristup internetu preko Wi-Fi-a, tj. nema fizički RJ-45 ulaz. Prema izvoru [4] na ispod prikazanoj slici možemo vidjeti eksponencijalan rast uređaja koji koriste Wi-Fi-a za prijenos podatka u usporedbi s alternativnim načinima prijenosa podataka.



Slika 3 – količina prenesenih podataka preko različitih mreža [5]

U domeni interneta stvari, Wi-Fi isto tako igra vrlo važnu ulogu upravo iz razloga što je dostupan u skoro svakom domu diljem svijeta, te nema potrebe za novim hardverom kod implementacija novih rješenja interneta stvari. Međutim, nedostatak Wi-Fi-a je vrlo velika potrošnja električne energije jer nikada nije bio zamišljen za korištenje na uređajima s potrebom za minimalnom potrošnjom električne energije. Iako to ne predstavlja toliko veliki problem jer postoji jedan vrlo jasan i ključan razlog zašto bi koristili Wi-Fi prije ZigBee-a ili LoRaWAN-a a to je vrlo velika propusnost podataka. Ukoliko se radi o potrebi za prijenosom nekog medijskog podatka poput slike ili video zapisa, slanje putem Wi-Fi-a postaje gotovo nužno. Iz tog razloga, Wi-Fi ćemo koristiti ukoliko je apsolutno nužno i nemamo drugog rješenja za prijenos značajnijih podataka.

Iako je prva generacija Wi-Fi-a (IEEE 802.11), kao što je prethodno navedeno, imala propusnost svega 1-2 Mb/s. Trenutna, šesta generacija, Wi-Fi-a (IEEE 802.11ax) ima

teoretsku propusnost od čak 9.6 Gb/s što ga čini nevjerojatno brzim za prijenos bilo kakve vrste podataka potrebne u IoT-u. Međutim važno je napomenuti kako se tu radi o idealnom, teorijskom scenariju na frekvenciji od 6 GHz koje je vrlo nepraktična za primjene u IoT-u zbog vrlo ograničenog dometa.

2.2.3.4.Ethernet

Ethernet je, također, vrlo poznati pojam današnjice. Radi se o žičanom načinu povezivanja na mrežu i slanje tj. primanje paketa putem IP protokola. U načelu je vrlo sličan Wi-Fi protokolu uz vrlo očitu razliku – žično povezivanje. Iako je to možda na prvi pogled nedostatak, ukoliko se ne radi o situaciji gdje kabel nikako ne možemo izbjeći, ethernet donosi brojne prednosti poput: puno stabilnije veze prijenosa, manje smetnje kod prijenosa podataka, veće brzine prijenosa, veća udaljenost prijenosa.

Također je jedna od velikih prednosti tehnologija napajanja preko etherneta (Power Over Ethernet, skraćeno PoE) što nam u kontekstu IoT-a igra veliki plus jer možemo smanjiti potrebe kabele koje je potrebno „provući“ do IoT uređaja na samo jedan preko kojeg će ići mreža i napajanje. Temeljen na standardu IEEE 802.3af iz 2003. godine (kasnije dolazi 802.3at i 802.3.bt) PoE nam omogućuje napajanje uređaja snage do 15.4 W, dok kasniji standardi omogućuju napajanje snagom do čak 71.3 W.

2.2.3.5.Mobilna mreža (LTE, 5G)

Mobilne mreže prošle su kroz značajan razvoj, od prve generacije (1G) do najnovije, pete generacije (5G). Svaka generacija donijela je tehnološke inovacije i unaprijedila mogućnosti bežične komunikacije.

Četvrta generacija mobilnih mreža (4G), dovršena 2011. godine, donosi ključno unaprjeđenje – uvođenje internetskog protokola (IP), što omogućuje korisniku spajanje na pristupne točke u vlasništvu telekomunikacijskih operatera. Četvrta generacija mobilnih mreža pokreće revoluciju u mogućnosti IoT uređaja koji mogu biti u potpunosti neovisni i samostalni.

Peta generacija mobilnih mreža (5G), prema planovima bi do 2025. godine trebalo biti pokriveno 90% Hrvatske 5G mrežom. Donosi značajna unaprjeđenja naprema 4G mreže u pogledu IoT-a, neka od unaprjeđenja su:

- Smanjena potrošnja energije
- Povećana propusnost i smanjena latencija
- Veći broj uređaja može biti istovremeno povezan

upravo ga ti razlozi čine ključnim za budućnost razvoja IoT-a.

2.2.3.6.Serijska komunikacija – RS-232, RS-485

Serijska komunikacija, uključujući standarde RS-232 i RS-485, igra važnu ulogu u IoT-u. Ovi komunikacijski standardi omogućuju pouzdanu razmjenu podataka između uređaja, čime se osigurava učinkovita integracija različitih komponenti IoT sustava.

RS-232 je jedan od najstarijih standarda za serijsku komunikaciju, razvijen 1960-ih godina. Unatoč svojoj starosti, još uvijek se koristi u mnogim industrijskim i komercijalnim aplikacijama. Glavne karakteristike RS-232 standarda uključuju:

- Jednostavnost – isključivo za komunikaciju između dva uređaja
- Kratki domet – maksimalna udaljenost do 15 metara između uređaja
- Niska propusnost - brzine prijenosa do 115.2 kbps

U kontekstu IoT-a, RS-232 se često koristi za povezivanje senzora, kontrolera i drugih uređaja s centralnim upravljačima ili računalima. Na primjer, može se koristiti za povezivanje toplinskog senzora sa sustavom za praćenje temperature ili za povezivanje starijih industrijskih strojeva s modernim IoT sustavima.

RS-485 je napredniji standard za serijsku komunikaciju, dizajniran za komunikaciju na većim udaljenostima i s većim brojem uređaja. Ključne značajke RS-485 uključuju:

- Povezivanje do 32 uređaja na jednoj komunikacijskoj liniji
- Prijenos podataka na udaljenostima do 1200 metara
- Vrlo visoka otpornost na vanjske elektromagnetske smetnje

RS-485 se često koristi u IoT rješenjima koje zahtijevaju pouzdanu i dugotrajnu komunikaciju između različitih uređaja. Primjeri uključuju sustave za automatizaciju zgrada, industrijske kontrolne sustave, te sustave za praćenje i upravljanje energijom.

U IoT-u serijska komunikacija putem RS-232 i RS-485 standarda omogućava povezivanje različitih uređaja i senzora sa centralnim kontrolerom za obradu podataka. To je posebno korisno u situacijama gdje su potrebne stabilne i pouzdane komunikacijske veze, te gdje druge tehnologije poput WiFi ili Bluetooth nisu prikladne zbog pouzdanosti, jednostavnosti, dugotrajnosti i sl.

Iako su to relativno stari komunikacijski standardi njihova pouzdanost, jednostavnost i otpornost na smetnje čine ih relevantnim i danas, posebno u IoT rješenjima poput

automatizacija velikih industrijskih postrojenja, poslovnim zgradama i sl. gdje je izrazito važna robusna i dugotrajna uporaba.

2.3. Prilike i izazovi IoT-a

Internet stvari predstavlja jednu od najvažnijih tehnoloških inovacija našeg vremena, omogućavajući povezivanje fizičkih uređaja na Internet i njihovu međusobnu komunikaciju. Iako IoT donosi brojne mogućnosti koje mogu revolucionirati razne industrije, on također donosi niz izazova koji se moraju prevladati kako bi se ostvario njegov puni potencijal. Neke od tih prilika i izazova će biti detaljnije raspisane u ovom poglavlju.

2.3.1. Prilike IoT-a

IoT pruža ogromne prilike za poboljšanje učinkovitosti, kvalitete života i ekonomske produktivnosti. U industriji, IoT omogućuje stvaranje pametnih tvornica gdje su strojevi, senzori i kontrolni sustavi međusobno povezani, omogućujući optimizaciju proizvodnih procesa i smanjenje troškova. U poljoprivredi, IoT tehnologije omogućuju precizno praćenje uvjeta usjeva, što rezultira boljim prinosima i smanjenim korištenjem resursa poput vode i gnojiva. Pametni gradovi, koji koriste IoT za upravljanje infrastrukturom, mogu značajno poboljšati kvalitetu života svojih stanovnika kroz optimizirano upravljanje prometom, energijom i otpadom.

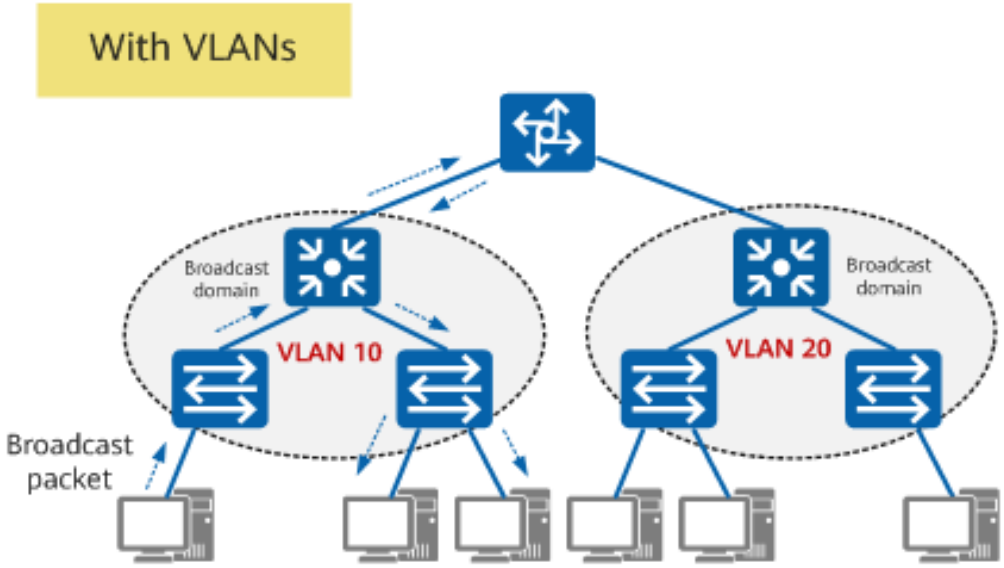
Pored industrijskih primjena, IoT također ima značajan utjecaj na svakodnevni život pojedinaca kod kuće. Pametni kućanski uređaji omogućuju korisnicima upravljanje svojim domovima na daljinu, stvarajući sigurnije i energetski učinkovitije životne prostore. IoT također igra ključnu ulogu u zdravstvenom sektoru, gdje pametni medicinski uređaji omogućuju praćenje vitalnih znakova pacijenata u stvarnom vremenu, čime se poboljšava kvaliteta skrbi i omogućuje pravovremena medicinska intervencija.

2.3.2. Izazovi IoT-a

Međutim, unatoč svim brojnim prilikama i pozitivnim stranama, IoT donosi i niz izazova. Jedan od najvećih izazova vezanih uz IoT je sigurnost. S obzirom na to da su IoT uređaji međusobno povezani i često imaju pristup osjetljivim podacima, oni predstavljaju potencijalne mete za kibernetičke napade. Sigurnosni propusti u IoT uređajima mogu rezultirati ozbiljnim posljedicama, uključujući gubitak podataka, neovlašten pristup mrežama, pa čak i fizičku štetu u slučaju napada na industrijske ili medicinske uređaje što može dovesti do velike materijalne, pa čak i štete opasne po život korisnika.

Također, postoji izazov vezan uz interoperabilnost između različitih IoT sustava. S obzirom na raznolikost proizvođača i standarda, često je teško osigurati da svi uređaji unutar IoT sustava mogu nesmetano komunicirati i razmjenjivati podatke. Ovaj problem može rezultirati fragmentacijom tržišta i ograničenjem mogućnosti za širu primjenu IoT rješenja. Nadalje, velika količina podataka koju generiraju IoT uređaji postavlja izazove u vezi s pohranom, analizom i zaštitom tih podataka, posebno kada se radi o osjetljivim ili osobnim informacijama .

Još jedan od ključnih problema IoT-a i samih IoT uređaja je sigurnost. Neupitno je kako je Internet donio brojne pozitivne promjene u svijetu, te bi se moglo reći da je čak postao i nužnost poput struje i vode gdje bi život bez njega postao gotovo nezamisliv. Međutim iako Internet donosi ogroman broj pogodnosti za čovjeka, jer omogućuje sveprisutnost i povezanost cijelog svijeta, tako upravo iz te povezanosti proizlaze brojni sigurnosni problemi. Naime, kako nam Internet omogućuje pristupanju našeg uređaja, tako i omogućuje pristupanju uređaju svim ostalim osobama na Internetu te tu vrlo brzo možemo shvatiti veliki problem. Najbolje rješenje bi bilo onemogućiti IoT uređajima spajanje na Internet nego ih konfigurirati na odvojenu mrežu tj. VLAN[3] koji bi dozvolio IoT uređajima isključivo komunikaciju s centralnim upravljačem kojem bi, po potrebi, dalje bio omogućen pristup internetu za udaljeni pristup.



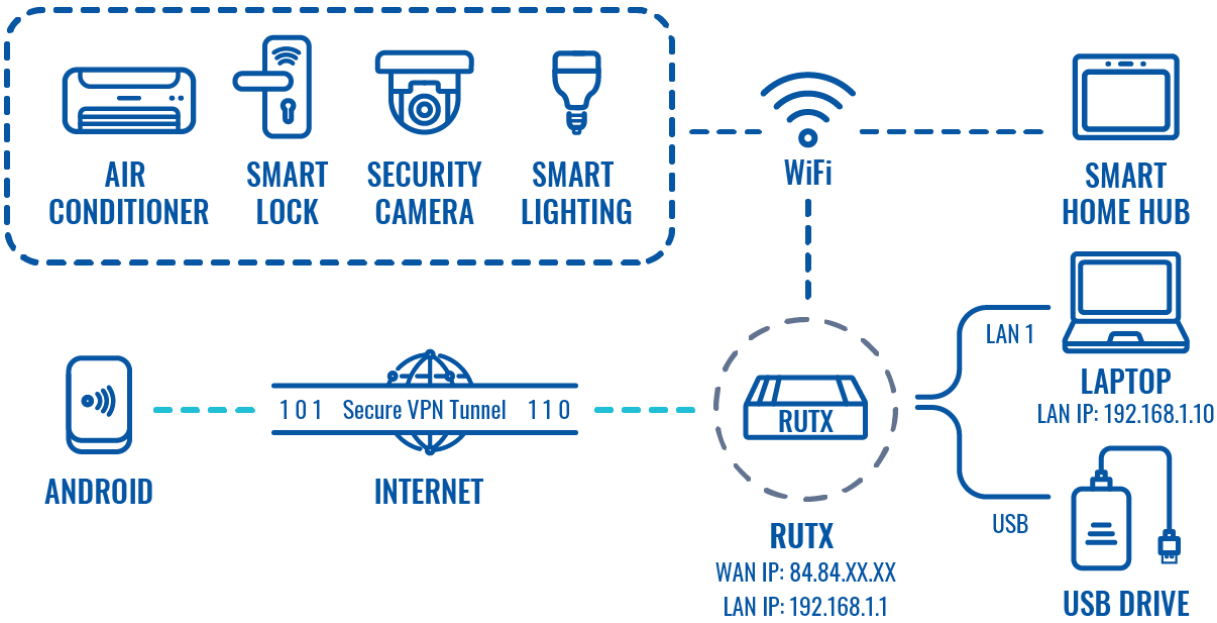
Slika 4 - Primjer segregacije mreže po VLAN-ovima

(izvor: https://info.support.huawei.com/info-finder/encyclopedia/en/VLAN.html, preuzeto 2024.)

[3] Virtual Local Area Network (VLAN) – omogućuje podjelu mreže na logičke cjeline

Međutim u svijetu kućnih IoT uređaja nije rijetko vidjeti da je jedan od zahtjeva uređaja kako bi uopće funkcionirao pristup Internetu. Razlog takvog uvjeta je nužna komunikacija uređaja sa serverima proizvođača kako bi korisniku bile omogućene željene funkcionalnosti uređaja ili kako bi proizvođač mogao dodatno naplatiti određene funkcionalnosti, npr. u obliku mjesečne pretplate. Iako možda to prosječnoj osobi ne stvara problem, te će bez promišljanja spojiti taj uređaj na Internet, tu zapravo dolazi do velikog sigurnosnog problema za cijelu kućnu mrežu i sve uređaje koji se nalaze na njoj. Spajanjem uređaja na servere nepoznatog vlasnika omogućujemo obostranu komunikaciju između naše kućne mreže i servera, što znači da ukoliko neka osoba ima maliciozne namjere može dobiti pristup kućnoj mreži na kojoj se uređaj nalazi preko uređaja. Naravno da taj rizik proizvođači uređaja uvijek pokušavaju smanjiti na minimum, međutim rizik je uvijek postojeći. Zato je vrlo važno, pogotovo kod uređaja povezanih na Internet, održavati uređaj ažuriranim na zadnjoj softverskoj inačici jer se na taj način popravljaju svi poznati sigurnosni problemi.

Najbolja praksa za održavanje visoke razine sigurnosti za IoT uređaje na kućnoj mreži je korištenje uređaje otvorenog koda koji komuniciraju isključivo sa centralnim kontrolerom koji se nalazi na lokalnoj mreži te onemogućiti pristup centralnom kontroleru pristup internetu, tj. vanjskoj mreži. Međutim ukoliko onemogućimo udaljeni pristup tada gubimo jedan od glavnih funkcionalnosti IoT sustava kućne automatike. Kako bi ipak mogli zadržati tu funkcionalnost uz neznatno manju količinu sigurnosti kućne mreže možemo implementirati tzv. Virtualnu Privatnu Mrežu (skraćeno VPN). Koja nam omogućuje realiziranje sigurne, privatne i kriptirane konekcije između kućne mreže na kojoj se nalaze IoT uređaji i udaljenog uređaja s kojim pokušavamo pristupiti IoT uređaju.



Slika 5 - Topologija IoT mreže s udaljenim pristupom preko VPN mreže

(izvor: https://wiki.teltonika-networks.com, preuzeto: 2024.)

Međutim, VPN nije jedina opcija za osiguranje pristupa IoT uređajima. Jedna od naprednijih i sigurnijih alternativa je implementacija Zero Trust mreže. Zero Trust pristup temelji se na principu da se nijedan uređaj, bez obzira na lokaciju, ne može automatski smatrati pouzdanim. Ovaj pristup značajno povećava sigurnost sustava jer eliminira implicitno povjerenje i stalno provjerava svaki zahtjev za pristup.

Korištenje Zero Trust mreža predstavlja napredniju alternativu VPN-u za osiguranje IoT uređaja. Implementacijom rješenja poput Twingate-a, korisnici mogu osigurati svoju kućnu mrežu i IoT uređaje protiv različitih prijetnji, dok istovremeno zadržavaju funkcionalnost i fleksibilnost udaljenog pristupa. Zero Trust pristup omogućuje bolju kontrolu, povećava sigurnost i olakšava upravljanje mrežnim resursima u sve složenijem IoT okruženju.

2.4. Interakcija korisnika s IoT uređajima

Naime korisnik će vrlo rijetko vršiti interakciju direktno s IoT uređajem jer sami uređaj ima vrlo ograničene sposobnosti, u velikom broju slučajeva su to samo – očitavanje vrijednosti senzora i/ili podešavanje stanja aktuatora. Iz tog razloga se u pravilu uvijek koristi aplikacija između samog IoT uređaja i korisnika, odnosno čovjeka.

Iako je pametni telefon najčešće korišten način za interakciju s IoT uređajima zbog njegove velike dostupnosti i jednostavnosti korištenja, izbor uređaja za interakciju može značajno varirati ovisno o primjeni. Na primjer, za kućne primjene, kao što su sustavi "pametne kuće", pametni telefon predstavlja optimalan izbor.

S druge strane, za interakciju s ozbiljnijim ili kompleksnijim sustavima u kontroliranim okruženjima, često je prikladnije razviti i koristiti stolne ili web aplikacije. Ove aplikacije omogućuju korisnicima pristup detaljnijim informacijama i naprednijim funkcijama koje mogu biti potrebne za upravljanje složenijim sustavima ili za profesionalnu primjenu. Stolne aplikacije, zbog svoje veće prilagodljivosti i funkcionalnosti, često su bolji izbor za rad u okruženju gdje je potrebna veća sigurnost i stabilnost.

U konačnici, odabir vrste aplikacije za interakciju s IoT uređajem ovisi o specifičnostima primjene, potrebama korisnika i sigurnosnim zahtjevima okoline u kojoj se aplikacija koristi. Tako, dok pametni telefon predstavlja jednostavno i praktično rješenje za kućne primjene, stolne i web aplikacije pružaju dodatnu funkcionalnost i sigurnost potrebnu za profesionalne i kompleksnije sustave

3. IoT u domeni upravljanja parkirnim prostorima

S povećanjem broja vozila, pronalazak slobodnog parkirnog mjesta postaje sve teže, što rezultira gubitkom vremena, povećanom emisijom štetnih plinova i frustracijom vozača. Tradicionalni načini upravljanja parkirnim prostorima često se oslanjaju na ručne procese ili zastarjele tehnologije, koje nisu u stanju učinkovito rješiti današnje zahtjeve. U tom kontekstu, tehnologije Interneta stvari (IoT) nude revolucionarne mogućnosti za optimizaciju upravljanja parkirnim prostorima, omogućujući stvaranje pametnih parkirnih sustava koji su prilagođeni potrebama modernih gradova.

Budući da je primjena IoT-a ključna za izradu ovog rada, važno je detaljno objasniti razloge za takav pristup. Primjerice, ako bismo trebali implementirati IoT rješenje u garažni prostor u Zagrebu, možemo pretpostaviti da bi bilo potrebno nadzirati oko 359 parkirnih mjesta, što je prema izvoru [5] prosječan broj parkirnih mjesta u javnim garažama u Zagrebu. S druge strane, trgovački centri poput Arena Centra u Zagrebu imaju čak 3000 parkirnih mjesta koja također zahtijevaju kontinuirani nadzor, 24 sata dnevno, 7 dana u tjednu. Upravo ovakve situacije ukazuju na važnost IoT sustava za učinkovito upravljanje parkiralištima.Vrlo brzo možemo shvatiti kako se tu radi o povelikom pothvatu gdje je ključna optimizacija hardvera kako bi se maksimalno smanjila ušteda klijentu. IoT je upravo osmišljen za veliki broj uređaja koji spajaju fizički svijet s virtualnim, tj. putem senzora prikupljaju željene parametre koji se obrađuju i izmjenjuju putem interneta uz minimalni trošak hardvera i potrošnje električne energije. Važno je istaknuti da senzori u nekim slučajevima ne moraju biti povezani žičanim putem, za razliku od postojećih rješenja koja zahtijevaju složeno kabliranje cijele garaže. U takvim sustavima ključno je voditi računa o maksimalnim dopuštenim dužinama kabela, što u velikim garažama može brzo postati značajan izazov. Takvo nešto je s IoT uređajima lako moguće jer su vrlo optimizirani što se tiče potrošnje električne energije. Što bi značilo ukoliko koristimo protokol poput Zigbee-a gdje bi svaki senzor bio interoperabilan i bežičan, vrlo brzo bi mogli integrirati rješenje za nadzor parkirnih mjesta.

3.1. Problematika upravljanja parkirnih prostora

Jedan od ključnih problema u upravljanju parkirnim prostorima je nedostatak informacija o trenutnoj zauzetosti parkirnih mjesta u stvarnom vremenu. Vozači često provode značajno vrijeme tražeći slobodno parkirno mjesto, što ne samo da povećava prometne gužve, već i negativno utječe na okoliš kroz povećanu potrošnju goriva i emisiju CO_2.

Nadalje, nedostatak preciznih informacija o zauzetosti parkirnih mjesta otežava optimizaciju korištenja parkirnih kapaciteta, što može rezultirati neefikasnim korištenjem dostupnog prostora. Tradicionalni sustavi često se oslanjaju na fizičke barijere, ulazne karte i ručne provjere, što može biti skupo za održavanje i nepraktično za vozače.

Osim toga, sigurnost parkirnih prostora predstavlja još jedan važan izazov. Neovlašteno parkiranje, vandalizam i krađe vozila su česti problemi koji zahtijevaju učinkovita rješenja. Mnogi postojeći parkirni sustavi nisu opremljeni potrebnim alatima za praćenje i identifikaciju vozila, što otežava provođenje sigurnosnih mjera i zaštitu korisnika parkirališta kao i pronalazak automobila u velikim parkirnim prostorima gdje se vrlo lagano može biti izgubiti tijekom potrage za parkiranim automobilom.

3.2. Potencijal IoT-a u upravljanju parkirnih prostora

IoT tehnologije pružaju rješenja koja mogu značajno unaprijediti način na koji se upravlja parkirnim prostorima. Korištenjem mreže povezanih senzora, kamera i drugih uređaja, pametni parkirni sustavi mogu pratiti zauzetost parkirnih mjesta u stvarnom vremenu i pružiti vozačima informacije o dostupnosti parkirnih mjesta putem mobilnih aplikacija, digitalnih prikazivača ili sl. Ovi sustavi omogućuju optimizaciju korištenja prostora, smanjenje vremena traženja parkirnog mjesta i traženja parkiranog vozila, kao i povećanje ukupnog zadovoljstva korisnika.

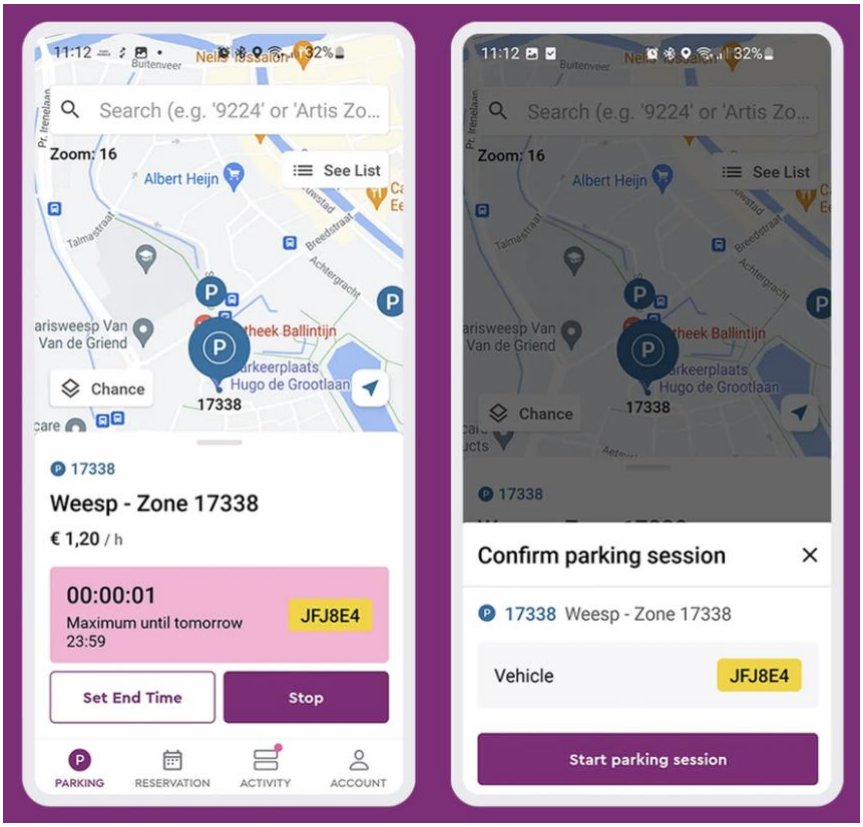
Jedan od ključnih elemenata pametnih parkirnih sustava je mogućnost identifikacije vozila. Upotrebom tehnologije optičkog prepoznavanja znakova (*skraćeno: OCR*), sustavi mogu automatski prepoznati registarske oznake vozila pri parkiranju vozila na parkirno mjesto, što omogućuje vođenje evidencije o ulasku i izlasku vozila, automatsko naplaćivanje parkiranja i provođenje raznih sigurnosnih mjera parkirnog prostora poput zabrane parkiranja vozila na rezervirana mjesta. Također, integracija ovih sustava s mobilnim aplikacijama bi mogla omogućiti korisnicima rezervaciju parkirnog mjesta unaprijed i pronalazak parkiranog vozila u parkirnom prostoru, što dodatno poboljšava iskustvo parkiranja.

3.3. Primjeri postojećih rješenja

Na tržištu već postoje različita IoT rješenja za upravljanje parkirnim prostorima koja demonstriraju potencijal ovih tehnologija. Jedan takav primjer je sustav *Smart Parking*, koji koristi senzore postavljene na parkirnim mjestima za detekciju zauzetosti. Podaci se prikupljaju u stvarnom vremenu i prenose na centralni sustav koji vozačima omogućuje pretragu

slobodnih mjesta putem mobilne aplikacije. Sustav također nudi mogućnost rezervacije parkirnih mjesta te automatsko naplaćivanje putem aplikacije, čime se eliminira potreba za papirnatim karticama i ručnim plaćanjem na automatima za naplatu parkinga koji su danas vrlo prisutni.

Drugi primjer je *ParkNow*, sustav koji koristi OCR tehnologiju za prepoznavanje registarskih oznaka vozila. Ovaj sustav omogućuje beskontaktno ulazak i izlazak iz parkirnih prostora, automatsku naplatu parkiranja te pruža detaljne izvještaje operaterima parkirališta o korištenju prostora. Integracija s pametnim telefonima omogućuje korisnicima jednostavno upravljanje svojim parkiranjem te dobivanje obavijesti o slobodnim mjestima u stvarnom vremenu.



Slika 6 - ParkNow aplikacija

(izvor: https://www.easypark.com, preuzeto: 2024.)

Također je važno napomenuti kako sami sustav za upravljanje predstavlja kompletni softver koji sudjeluje u svakodnevnom izvršavanju zadataka potrebnih za besprijekorno iskustvu krajnjem korisniku. Prethodna dva primjera predstavljaju odličan primjer gotovog kompletnog rješenja. Važno je napomenuti kako uz gotova kompletna rješenja postoje i samo hardverska rješenja koja se dalje integriraju u sustave za upravljanje parkirnih prostora kao što

su dva prethodna primjera. Jedno od takvih hardverskih senzora je *Cleverciti* IoT parking senzor koji se nakon postavljanja na visoko mjesto kako bi moglo pokriti što veću parkirnu površinu ima mogućnost pokriti čak 100 vozila. Međutim tu se dobiva isključivo informacija o zauzetosti mjesta bez mogućnost identifikacije vozila koje se nalazi na samom mjestu, te s time i gubimo veliki broj funkcionalnosti poput pronalaska vozila, rezervacije mjesta, naplate i sličnoga.

Slika 7 - Cleverciti senzor

(izvor: https://www.cleverciti.com, preuzeto: 2024.)

3.4. Relevantne IoT tehnologije u domeni upravljanja parkirnim prostorima

Za učinkovito upravljanje parkirnim prostorima, ključno je korištenje specifičnih IoT tehnologija koje uključuju kombinaciju softverskih rješenja, hardverskih komponenti i komunikacijskih protokola.

3.4.1. Senzori zauzetosti

Jedan od najvažnijih elemenata u pametnim parkirnim sustavima su senzori zauzetosti. Ovi senzori mogu koristiti različite tehnologije, uključujući ultrazvuk, infracrveno svjetlo ili magnetske senzore, za detekciju prisutnosti vozila na parkirnim mjestima. Senzori su obično povezani bežičnim mrežama, poput LoRaWAN ili ZigBee, što omogućuje prikupljanje podataka u stvarnom vremenu i njihovu obradu na centralnom sustavu. Međutim uz prethodno navedene tehnologije moguće je koristiti i optičko prepoznavanje zauzetosti vozila što je najnovija i naprednija tehnologija koja se trenutno koristi jer može točno identificirati koje se vozilo nalazi na mjestu, a ne samo nalazili se vozilo na mjestu. Takvi senzori su opremljeni optičkom kamerom i naprednim algoritmom za obradu videa i strojno prepoznavanje.

3.4.2. Optičko prepoznavanje znakova (OCR)

OCR je tehnologija koja omogućuje pretvaranje čovjeku čitljivog teksta s neke fotografije u strojno čitljivi tekst. OCR koristi složene algoritme računalnog vida i prepoznavanja uzoraka kako bi identificirao i izdvojio tekst iz slika, čime ga pretvara u digitalni format koji se može pretraživati, uređivati ili pohranjivati. Prvi komercijalni OCR sustavi pojavili su se 1960-ih godina, omogućujući prepoznavanje teksta pisanog na tipografskim strojevima. Tijekom godina, OCR je prošao kroz mnoge inovacije i napretke, od jednostavnih sustava koji su prepoznavali samo ograničen broj znakova do sofisticiranih tehnologija koje mogu prepoznati više jezika, stilova pisanja i rukopisa.

U kontekstu parkirnog sustava, OCR ima ključnu ulogu u automatizaciji i optimizaciji parkiranja. Konkretno, OCR se koristi za prepoznavanje registarskih pločica vozila prilikom ulaska i izlaska s parkinga. ANPR (engl. Automatic Number Plate Recognition) kamere se postavljaju na ulazu i izlazu te snimaju registracijske pločice vozila, nakon čega OCR softver analizira slike, prepoznaje i čita registracijsku oznaku vozila.

3.4.3. Komunikacijski protokoli

Za povezivanje svih IoT uređaja unutar parkirnog sustava koriste se različiti komunikacijski protokoli. U kontekstu parkiranja, najčešće se koriste bežične tehnologije poput Wi-Fi, ZigBee i LoRaWAN. Wi-Fi omogućuje prijenos većih količina podataka, kao što su slike s kamera, dok ZigBee i LoRaWAN pružaju nisku potrošnju energije i dug domet, što je idealno za senzore koji prate zauzetost parkirnih mjesta.

3.4.4. Mobilna ili stolna aplikacija

Uloga mobilnih aplikacija u pametnim parkirnim sustavima je ključna jer omogućuju korisnicima jednostavan pristup informacijama o parkiranju, rezervaciju mjesta i beskontaktno plaćanje. Aplikacije također omogućuju vozačima navigaciju do slobodnih parkirnih mjesta i dobivanje obavijesti u stvarnom vremenu. Dok je uloga stolne aplikacije ključna za osoblje parkirnih prostora jer omogućuje uvid u bitne analitike parkirnog prostora kao što je trenutna zauzetost, pronalazak vozila i sl.

4. Implementacija rješenja za parkirne prostore

Razvoj IoT uređaja, u maloj količini, je zapravo vrlo jednostavan i dostupan pojedincu. Međutim, ukoliko se radi o izrade namjenskog IoT uređaja za masovnu proizvodnju i prodaju tj. ugradnju, istraživanje, razvoj i izrada uređaja drastično poskupljuje. Čak do te mjere gdje se nekad razvoj uređaja neće niti isplatiti dok se ne proda nekoliko desetaka ili stotina uređaja. Kako je ovo jedno od takvih rješenja gdje bi sam razvoj bio vrlo kompliciran i skup jer se radi o potrebi za jako velikom broju uređaja kako bi se opremio jedan garažni prostor. Ja sam odlučio u sklopu ovog završnog rada napraviti samo jedan IoT senzor kao koncept rješenja (engl. Proof Of Concept, skraćeno PoC).

4.1. Opis problema

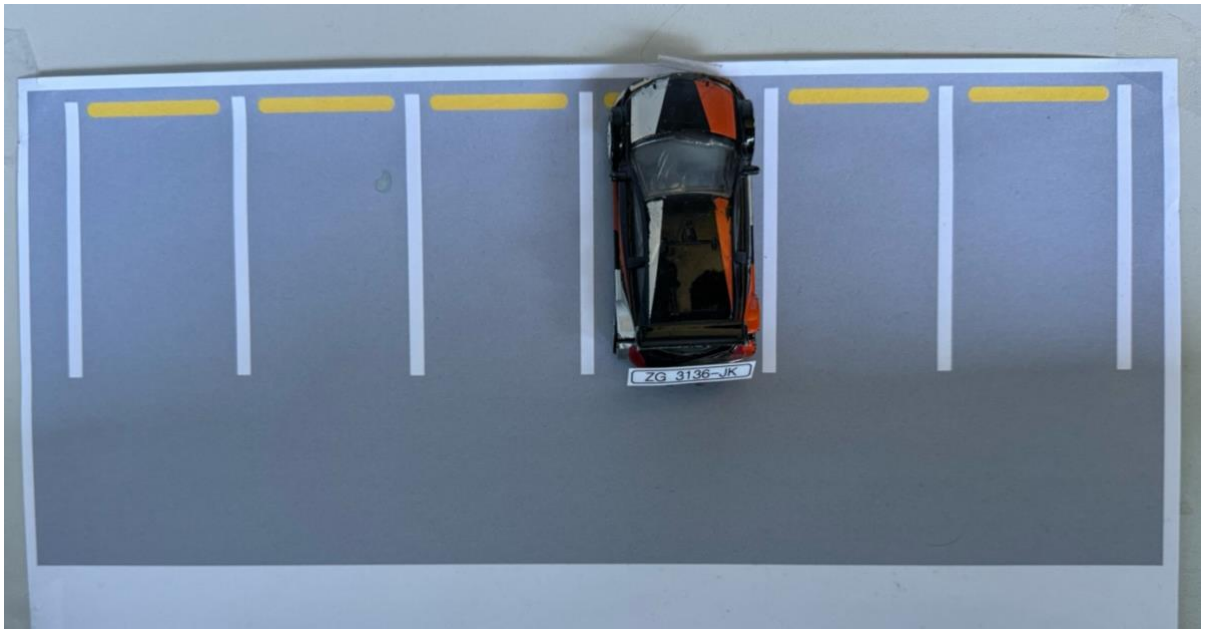
Ideja ovog rješenja jest drastično olakšati rad zaposlenicima parkirnih prostora i korisnicima istih. Naime provjera dostupnosti nekog parkirnog mjesta nije ništa novo, naprotiv, dostupno je u skoro svakoj garaži u svijetu. Međutim, novo je mogućnost identifikacije parkiranog vozila na parkirnom mjestu. S tom mogućnošću možemo implementirati vrlo veliki skup funkcionalnosti koji će u potpunosti promijeniti način organizacije i korištenja parkirnih prostora.

Dok postojeći senzori, temeljeni na zastarjelim tehnologijama poput ultrazvučnog senzora, imaju mogućnost vraćanja dva stanja (zauzeto ili slobodno) tj. tri stanja u slučaju da se dogodila neka greška na senzoru to se prijavljuje centrali. Moje rješenje će koristiti kameru kako bi s optičkim prepoznavanjem predmeta moglo zaključiti radili se o zauzetom mjestu ili slobodnom. Također, za razliku od postojećih senzora gdje je za svako parkirno mjesto potreban jedan senzor, ovdje će jedna kamera moći pokriti više parkirnih mjesta.

Zbog ograničenja i jednostavnosti testiranja rješenja napravio sam model parkirnog prostora te sam taj model koristio dalje kao reprezentaciju realnog parkirališta popratno s malim modelima automobila koji će dalje služiti u testiranju detekcije zauzetosti parkirnog mjesta. Parkirni senzor će javljati centralnom sustavu status parkirnog mjesta (zauzeto ili slobodno) zajedno s registracijskom oznakom automobila, ukoliko se radi o zauzetom mjestu.

Sustav će imati sljedeće funkcionalnosti:

- Prikazivanje stanja parkirnog prostora
- Prikazivanje opće zauzetosti parkirnog prostora
- Pretraživanje vozila prema registracijskoj oznaci unutar parkirnog prostora
- Rezerviranje parkirnih mjesta za određena vozila
- Obavještavanje ukoliko se nedozvoljeno vozilo parkira na rezervirano mjesto



Slika 8 - Model parkirnog prostora korišten u testiranju rješenja

Gornja slika prikazuje izrađeni model parkirnog prostora korišten tijekom testiranja softvera IoT senzora. Model se sastoji od šest parkirnih mjesta modelirana te isprintana na A4 papiru i četiri modela automobila različitih boja s nalijepljenim proizvoljno izabranom kombinacijom slova i brojeva koje su vjerodostojne sadržaju registracijskih oznaka u Republici hrvatskoj. Izabrani su automobili različitih boja kako bi se kvalitetno moglo testirati detektiranje vozila na

parkirnom mjestu jer se isto izvršava putem detekcije kontrasta i rubova na odabranom parkirnom mjestu. Neke boje vozila su se prikazale izazovnije za detekciju od drugih kao i promjena uspješnosti detektiranja u ovisnosti o ambijentalnom osvjetljenju modela, međutim pojedinosti problema s kojim sam se susretao tijekom testiranja na modelu će biti dodatno raspisane kasnije u poglavlju.

4.2. Opis rješenja

Zbog potrebe identifikacije vozila na svakom mjestu unutar garažnog prostora postalo je vrlo jasno kako će biti potrebno čitanje registracijskih oznaka na vozilima. Što znači kako se mora raditi o senzorima s kamerom i sposobnošću jednostavnijeg procesiranja slika. Kako bi se to postiglo, senzor mora biti sposoban slati slike vozila u centralni upravljački sustav, koji će potom obraditi dobivene slike i očitati registarske oznake, odnosno identificirati vozila na parkirnom mjestu. Alternativno, mogu se koristiti senzori s ugrađenom mogućnošću optičkog prepoznavanja, tako da senzor odmah očita registarske oznake i zajedno sa statusom vozila pošalje podatke u centralni upravljački sustav.

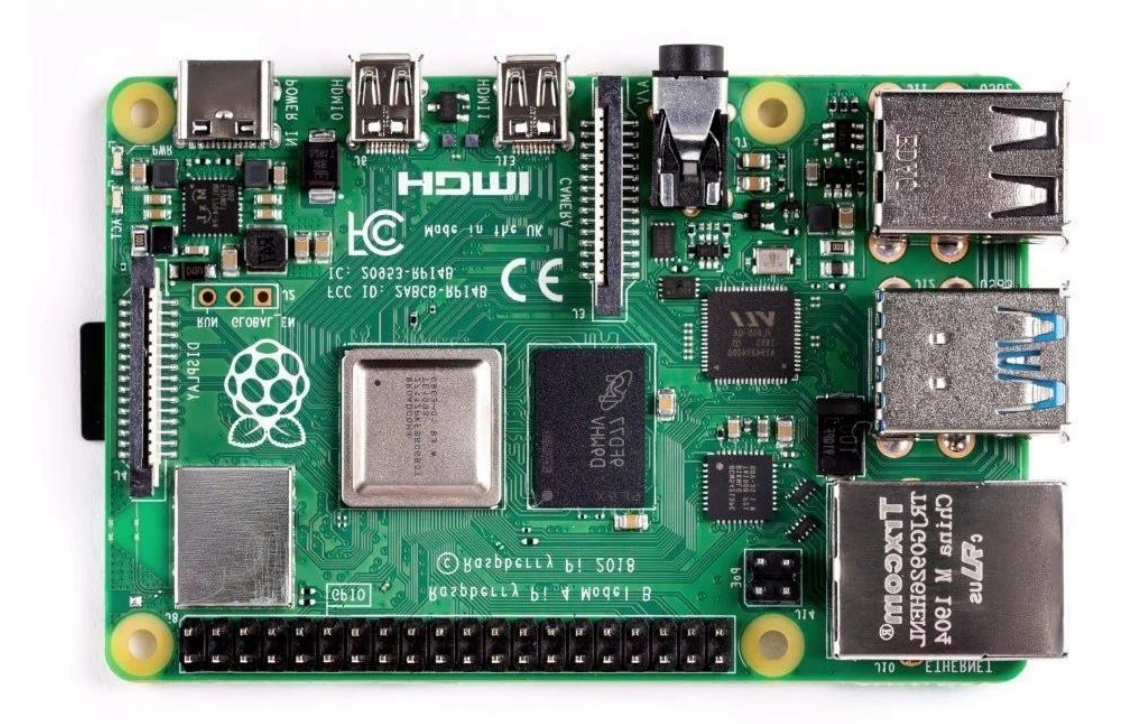
Ja sam se odlučio za drugu opciju gdje će se slati isključivo status mjesta i registracijska oznaka na mjestu jer će se na taj način drastično smanjiti mrežno opterećenje i opterećenje centralnog upravljača koji tada neće morati obavljati još taj zadatak uz mnoge druge gdje je svaka sekunda bitna. Dok na samom senzoru vremenski odaziv nije toliko kritičan te čak i ako to procesuiranje potraje nekoliko sekundi ništa negativno se neće dogoditi. Nakon uspješne obrade senzor bi podatke trebao proslijediti podatke na centralni upravljač koji će naknadno odlučivati što točno raditi s njima.

4.2.1. Odabir hardvera senzora

Uzevši prethodne zahtjeve u obzir znam da mi treba optički senzor tj. kamera i performanse dovoljne kako bi obavio neke osnovne funkcije optičkog prepoznavanja zauzetosti prostora i optičkog čitanja znakova tj. slova s registracijskih oznaka tablica. U idealnom svijetu, gdje bi se ovaj uređaj koristio u masovnoj primjeni, radio bi se namjenski hardver koji je u potpunosti prilagođen zadatku kojeg će odrađivati. Međutim, kako se ovdje radi o potrebi za jednim uređajem u sklopu izrade koncepta rješenja to je neizvedivo. Tako da sam se odlučio koristiti vrlo poznato i popularno računalo upravo za takve kućne primjene zbog njegove cijene i sposobnosti – Raspberry Pi.

4.2.1.1.Raspberry Pi

Raspberry Pi je maleno, jeftino i svestrano računalo razvijeno od strane Raspberry Pi Foundation. Dizajnirano je za edukativne svrhe, ali je našlo široku primjenu i među amaterima, profesionalcima i industrijskim korisnicima. Uređaj se sastoji od jedne ploče koja uključuje procesor, memoriju, USB i HDMI portove, GPIO pinove za povezivanje s vanjskim uređajima i podršku za operativne sustave poput Linuxa. Raspberry Pi se koristi za razne projekte, uključujući robotiku, automatizaciju kuće, multimedijalne centre i IoT aplikacije, zahvaljujući svojoj pristupačnosti i širokom ekosustavu podrške.

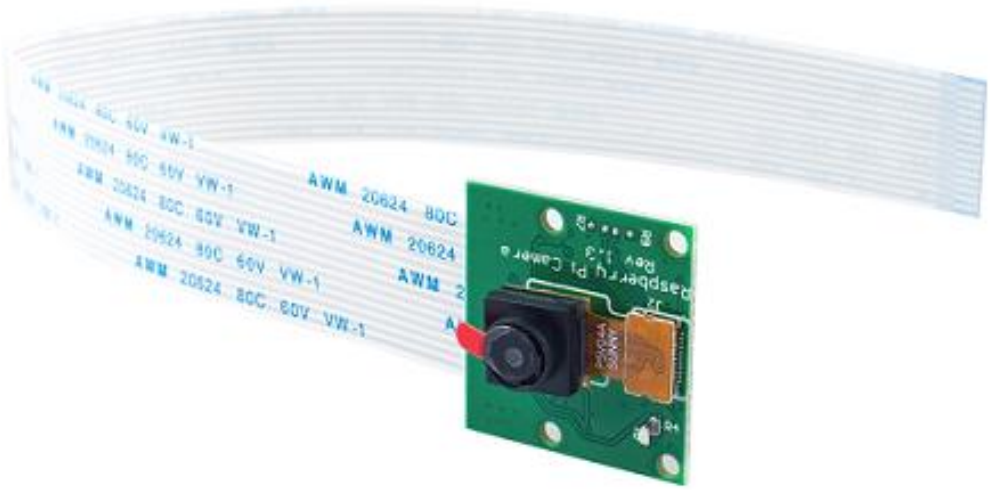


Slika 9 - Raspberry Pi 4 Model B

(izvor: https://www.raspberrypi.com, preuzeto: 2024.)

U izradi ovog projekta ja sam odlučio koristiti stariji model Raspberry Pi-a kojeg sam imao od prije, tzv. Raspberry Pi 4 Model B koji je predstavljen 2019. godine. Ova generacija Raspberry Pi-a je opremljena Cortex-A72 ARM procesorom s četiri jezgre i brzinom od svega 1.8GHz, što je možda s usporedbi s današnjim procesorima smiješno. Moramo uzeti u obzir da se ovdje radi o cijelom računalu koji je cijenom pristupačniji nego najjeftiniji aktualni procesor proizvođača Intel-a ili AMD-a i troši svega 15W struje na maksimalnom opterećenju što ga čini izrazito prikladnim za ovu primjenu. Još jedna od odličnih ideja proizvođača jest mogućnost odabira RAM memorije kod kupnje uređaja što značajno utječe na cijenu, te ukoliko imamo primjenu na uređaj za koju znamo da ne zahtjeva značajnu količinu RAM memorije

možemo kupiti uređaj s manjom količinom i uštediti. Uređaj koji ću ja koristiti za senzor ima 4 GB LPDDR4 memorije brzine od 3200 MHz što je i više nego dovoljno za potrebnu primjenu te bi za primjenu u realnom svijetu bilo dovoljno čak i 1 GB memorije. Uređaj je, također, opremljen GPU-om (engl. Graphics processing unit, skraćeno GPU) koji nije vrlo snažan, ali je dovoljan i vrlo bitan u pogledu optičkog procesiranja video prijenosa. Kako bi GPU imao neki video prijenos za obrađivanje potreban nam je sami video prijenos, za to ću koristiti službenu kameru istoga proizvođača „Raspberry Pi Camera Module“ koji na predviđeni SCI (engl. Camera Serial Interface) konektor koji se nalazi na samom Raspberry Pi 4 uređaju preko serijske konekcije ostvaruje video prijenos s kamere.



Slika 10 - Službeni dodatak kamere za Raspberry Pi 4

(izvor: https://www.raspberrypi.com, preuzeto: 2024.)

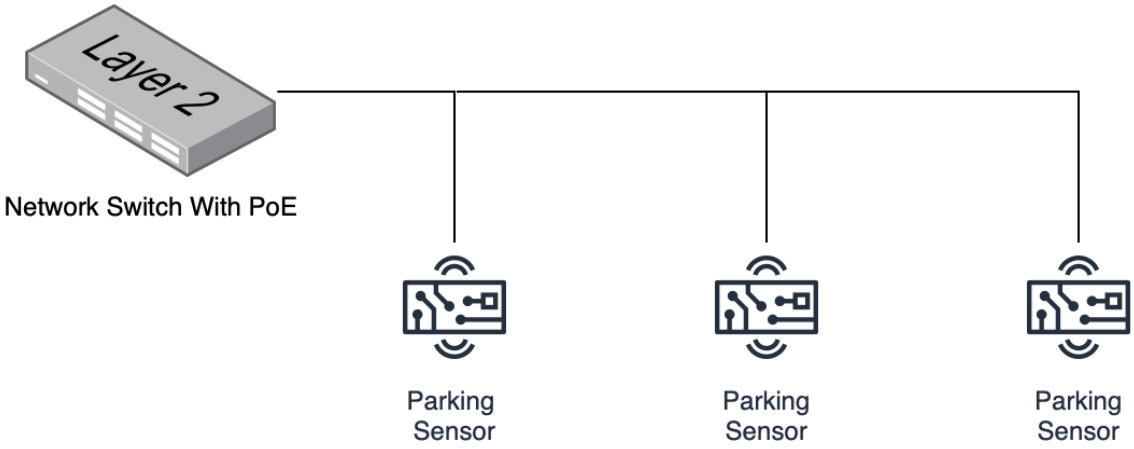
Kamera sadrži OV5647 senzor rezolucije 5 megapiksela te kut „gledanja“ od 65 stupnjeva, što ju čini idealnom za pokrivanje nekoliko parkirnih mjesta od jednom. Također, ima dosta dobre mogućnosti u okruženjima slabijeg ambijentalnog osvjetljenja, što ga čini idealnim za primjenu u slabo osvjetljenim garažnim prostorima. Jedan od problema s kojim sam se susreo tijekom podešavanja kamere jest to da je Raspberry Pi kompanija s objavom zadnje inačice Rasbian OS-a ukinula podršku za tim modulom te sam morao skinuti stariju verziju Rasbian OS-a kako

bi mogao dobiti natrag nativnu podršku kamere. Za implementaciju rješenja koristio sam Rasbian Buster OS koji je napravljen na distribuciji Linuxa. Raspbian je službeni operativni sustav za Raspberry Pi uređaje, optimiziran kako bi najbolje iskoristio hardverske mogućnosti Raspberry Pi uređaja. Raspbian Buster je verzija Raspbiana temeljena na Debian Buster distribuciji. Jedna od ključnih prednosti Raspbiana je njegova optimizacija za Raspberry Pi hardver, što uključuje prilagođene drivere i firmware koji omogućuju bolju performansu i pouzdanost. Raspbian također podržava GPIO (General Purpose Input/Output) pinove, što omogućuje jednostavno povezivanje i upravljanje vanjskim uređajima, senzorima i aktuatorima.

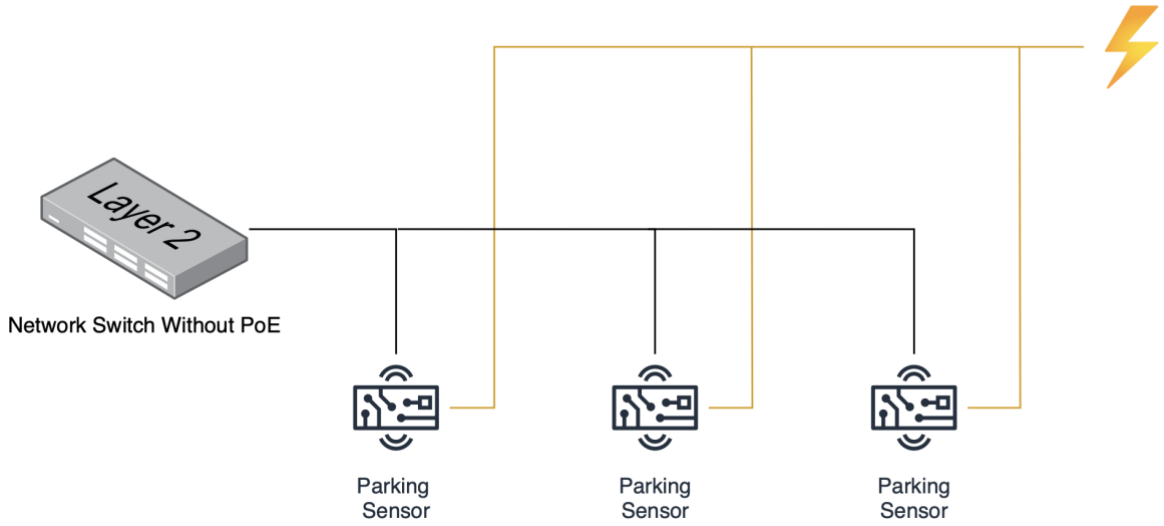
4.2.1.2.PoE HAT

Jedna od odličnih funkcionalnosti Raspberry Pi-a su tzv. „HAT“ dodatci što su zapravo hardverski dodatci koji se dodaju na sam Raspberry Pi uređaj kao „šešir“ te proširuju njegove funkcionalnosti. Postoji velika količina HAT-ova od raznih vrsta zaslona do 4G mrežnih kartica.

Kako bi ja u sklopu ovog projekta iskoristio i tu mogućnost Raspberry Pi-a odlučio sam iskoristiti PoE (Power over Ethernet) HAT koji omogućuje Raspberry Pi uređaju napajanje i komunikaciju preko jednog, Ethernet, kabela. PoE HAT ne samo da pojednostavljuje instalaciju smanjujući broj potrebnih kabela, već također povećava pouzdanost sustava eliminirajući potrebu za dodatnim napajanjem, što može biti ključno u okruženjima kao što su parkirališta gdje je praktičnost i sigurnost instalacije prioritet. U ovom poglavlju neću značajno ulaziti u specifikacije PoE-a jer je to malo detaljnije objašnjeno u poglavlju 6.1.4. samo je važno napomenuti kako PoE, uz korištenje PoE preklopnika, omogućuje napajanje uređaja preko Ethernet kabela. Također, PoE HAT omogućuje jednostavnije održavanje sustava. U slučaju potrebe za resetiranjem ili nadogradnjom, sve operacije mogu se izvesti putem mrežne veze, bez potrebe za fizičkim pristupom uređaju. Ovo je posebno korisno u situacijama gdje su uređaji postavljeni na teško dostupnim mjestima. Integracija PoE HAT-a u parkirališni sustav također omogućuje skalabilnost. Kako se potrebe mijenjaju, novi uređaji i senzori mogu se lako dodati na mrežu bez potrebe za velikim infrastrukturnim promjenama. Ovo omogućuje sustavu da raste i prilagođava se novim zahtjevima bez značajnih prekida ili dodatnih troškova.



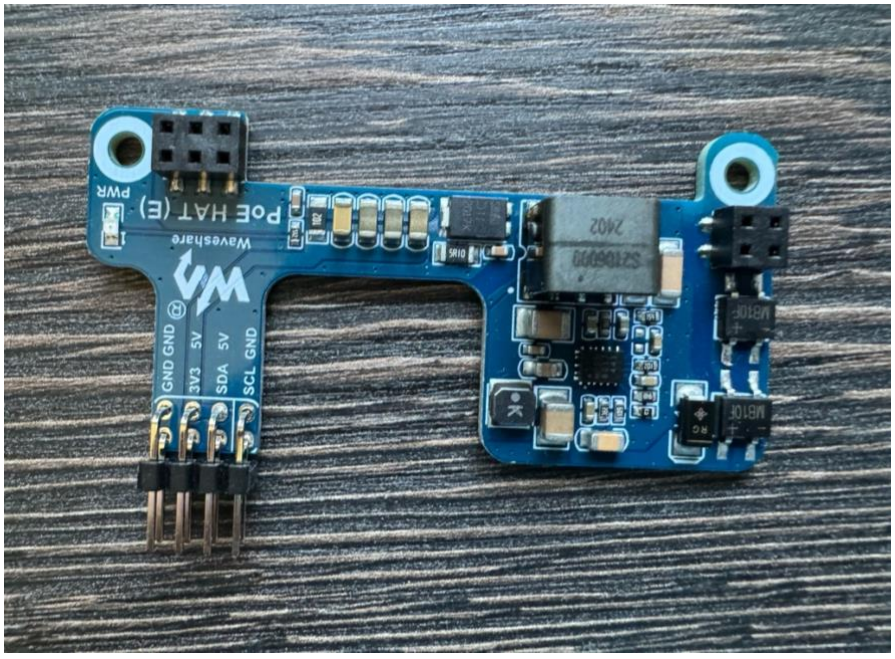
Slika 11 - Primjer senzora koji koriste PoE



Slika 12 - Primjer senzora koji ne koriste PoE

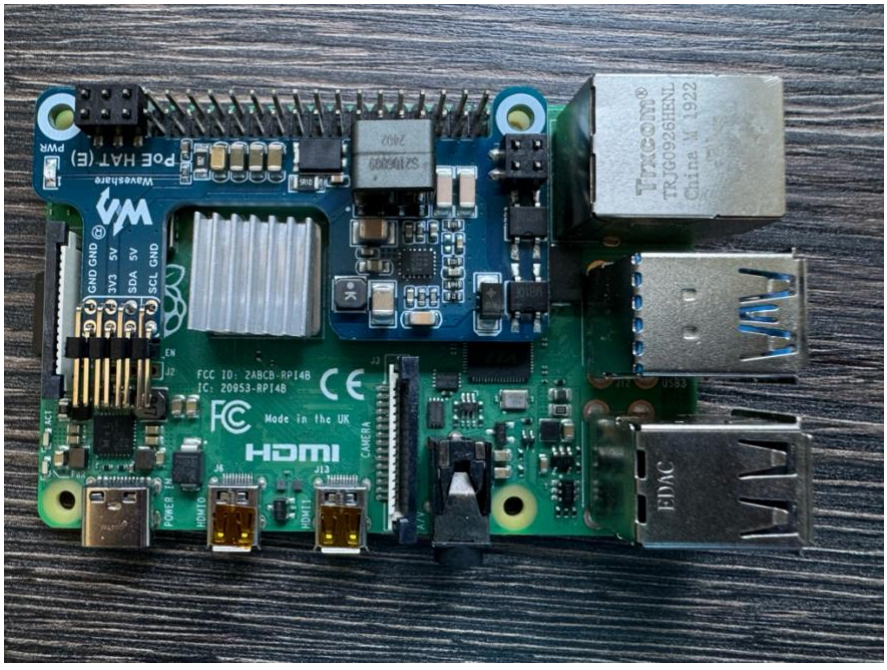
Iz slika 9 i 10 možemo vidjeti kako na slici 9, koriste prednosti PoE-a, imamo samo jedan kabel koji je potrebno postaviti u garažnom prostoru. Međutim uz tu prednost, dobivamo i prednost redundancije napajanja, gdje će se senzori napajati putem mrežnog preklopnika koji se nalazi u nekoj serverskoj sobi koja će vrlo vjerojatno imati baterijsko napajanje u slučaju nestanka struje gradske mreže. Cilj u tako velikim instalacijama je uvijek smanjiti količinu točaka grešaka, gdje bi se na primjer u običnoj konfiguraciji (bez PoE-a) moglo pokvariti veliki broj napajanja u zasebnim uređajima jer bi morali imati adapter ili bi se morala koristiti jednosmjerna struja koja drastično smanjuje moguću udaljenost. Na ovaj način imamo jedno

napajanje koje se može pokvariti, a svi ozbiljniji mrežni preklopnici uvijek imaju dva napajanja upravo iz takvog razloga. Kako bi na svom rješenju iskoristio ovu prednost kupio sam PoE HAT koji izgleda ovako:



Slika 13 - PoE HAT

Nakon ugradnje ovog HAT-a Raspbarry Pi je moguće napajati preko tvorničkog RJ-45 priključka sa svakim mrežnim preklopnikom koji zadovoljava IEEE 802.3af standard.



Slika 14 - Raspberry Pi 4 s ugrađenim PoE HAT-om

Slika 15 - Kompletan hardver korišten izradi u rješenja

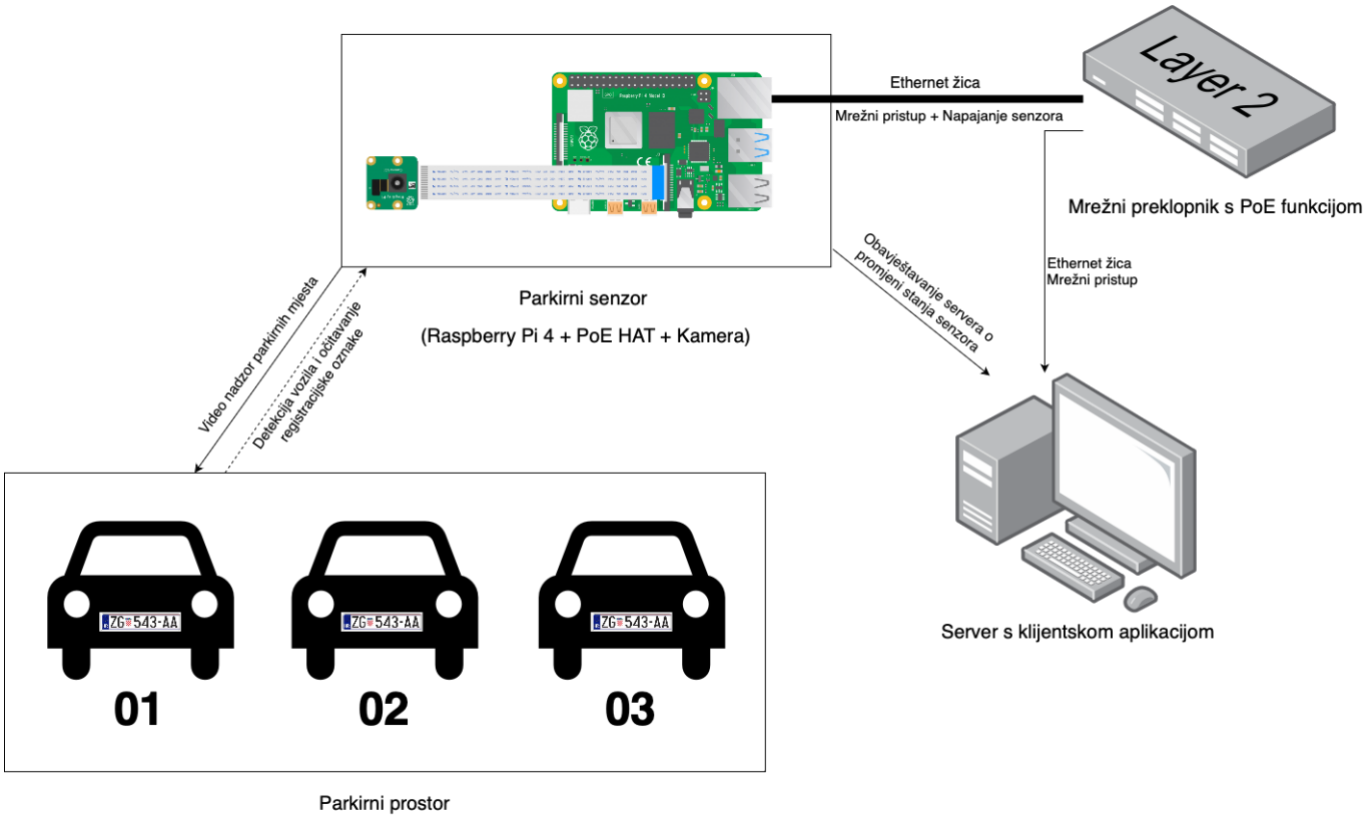
4.2.2. Arhitektura odabranog rješenja

U ovom poglavlju će biti detaljno prikazan i objašnjen dijagram arhitekture cijelog sustava. Cijeli sustav se sastoji od tri ključna elementa: senzora, mrežne infrastrukture i servera, koji istovremeno služi i kao klijentska aplikacija.

Senzorski dio sustava čini Raspberry Pi 4 s dodatnim PoE HAT-om i kamerom. Ovaj uređaj služi kao parkirni senzor smješten u parkirnom prostoru, odgovoran je za nadzor parkirnih mjesta i detekciju vozila. Kamera, koja je sastavni dio senzora, bilježi registracijske oznake vozila i šalje te podatke prema serveru. Napajanje senzora, kao i mrežna komunikacija, ostvaruje se putem jednog Ethernet kabela zahvaljujući Power over Ethernet (PoE) tehnologiji, čime se smanjuje potreba za dodatnim napajanjem i pojednostavljuje instalacija.

Mrežna infrastruktura uključuje mrežni preklopnik s PoE funkcijom koji služi kao središnja točka za povezivanje svih senzora i servera. Ovaj preklopnik ne samo da omogućava napajanje senzora putem Ethernet kabela, već osigurava i stabilan mrežni pristup, neophodan za komunikaciju između senzora i servera. Uloga mrežnog preklopnika je kritična jer omogućava povezivanje više senzora prema serveru koji će obrađivati podatke.

Server, koji u ovom rješenju istovremeno obavlja i funkciju klijentske aplikacije, ključni je dio cijelog sustava. Na serveru se odvijaju sve ključne operacije kao što su obrada podataka dobivenih sa senzora, pohrana i analiza podataka, te generiranje grafičkog prikaza za informiranje korisnika. Server je smješten na računalu koje je putem mrežnog preklopnika povezano sa senzorima, a uz to omogućava i pristup korisnicima koji putem klijentske aplikacije mogu pregledavati status parkirnih mjesta u stvarnom vremenu.



Slika 16 - Dijagram sveukupne arhitekture rješenja

Na gore prikazanom dijagramu su jasno vidljive sve komponente koje sudjeluju u radu ovog parkirnog sustava, te prikazuje kako su međusobno povezane. Važno je napomenuti kako bi se u produkcijskom okruženju znatno povećao broj parkirnih senzora te bi se promijenila mrežna arhitektura sukladno broju senzora. Ovdje je prikazana arhitektura odabranog rješenja.

4.3. Izrada softvera senzora

Kao i kod svakog IoT uređaja hardver je beskoristan bez softvera, naravno vrijedi i obrnutu. Kod pisanja svakog softvera prvi korak je odlučiti se za programski jezik. Svaki programski jezik ima svoje prednosti i nedostatke, no s obzirom na to da se u ovom projektu radi o optičkom prepoznavanju objekata i znakova na slikama, odnosno video prijenosu, zaključio sam da će Python biti najbolji izbor. Razlog tome su brojne biblioteke koje su posebno razvijene za ovakve vrste primjena.

Među svim bibliotekama koje su dostupne u Pythonu, posebno se ističe OpenCV (Open Source Computer Vision Library), koja pruža opsežan skup alata za obradu i analizu slike i videa. OpenCV omogućuje jednostavno izvođenje operacija poput detekcije objekata, prepoznavanja lica i analiza pokreta, što su ključne funkcionalnosti za domenu projekta.

4.3.1.OpenCV

U izradi ovog projekta OpenCV je igrao ključnu ulogu jer se koristio na velikom broju mjesta od detekcije zauzetosti mjesta do izoliranje registracijske tablice vozila kako bi se olakšao posao optičkom prepoznavanju znakova kod čitanja registracijske oznake.

4.3.1.1. Detekcija zauzetosti mjesta

```
def is_space_occupied(image, space):
    poly = space['poly']
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    mask = np.zeros_like(gray)
    cv2.fillPoly(mask, [poly], 255)

    masked_roi = cv2.bitwise_and(gray, gray, mask=mask)
    blurred = cv2.GaussianBlur(masked_roi, (5, 5), 0)

    thresh = cv2.adaptiveThreshold(blurred, 255,
cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.THRESH_BINARY_INV, 11, 2)
    contours, _ = cv2.findContours(thresh, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

    for contour in contours:
        x, y, w, h = cv2.boundingRect(contour)
        center = (x + w // 2, y + h // 2)

        if cv2.pointPolygonTest(poly, center, False) >= 0:
            contour_area = cv2.contourArea(contour)
            if contour_area > 0 and contour_area < 5000:
                return True

    return False
```

U gore napisanom kodu možemo vidjeti funkciju kojoj se prosljeđuje trenutna slika kao „image“ i mjesto za koje provjeravamo dostupnost kao „space“. Parkirna mjesta su u kodu definirana unutar config.py datoteci u obliku običnog polja s dvije dimenzije u kojem se nalaze koordinate parkirnih mjesta. Kako bi se korisniku olakšalo označavanje parkirnih mjesta s koordinatama napisan je *spacemapper.py* program koji prilikom pokretanja korisniku otvara prozor s video prijenosom kamere senzora. Korisnik može lijevim klikom označiti sve kutove parkirnog mjesta, nakon čega će ih program u konzoli ispisati u obliku koordinata. Ispisane koordinate korisnik može jednostavno unijeti u config.py datoteku.

4.3.1.2. Izoliranje registracijske oznake vozila

Za očitavanje znakova je bilo potrebno proslijediti sliku gdje se nalazi isključivo registracijska oznaka vozila i ništa drugo kako ne bi došlo do pogrešnog očitavanja registracijske oznake. Kako bi to učinio napisao sam funkciju u koju ulazi trenutna slika iz koje je potrebno izolirati registraciju, zatim ROI (engl. Region of Intrest) koji je također definiran u *config.py* datoteci i u njemu se nalaze koordinate prostora gdje se očekuje da će se tablica vozila od nekog mjesta nalaziti, te za kraj se nalazi *crop_percent* što je postotak koliko će se finalna slika smanjiti sa lijeve i desne strane prije nego što se proslijedi očitavanju znakova.

```
def extraction(image, roi, crop_percent=0.05):
    gray = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
    blurred = cv2.GaussianBlur(gray, (5, 5), 0)
    edged = cv2.Canny(blurred, 50, 200)
    contours, _ = cv2.findContours(edged, cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)
    contours = sorted(contours, key=cv2.contourArea, reverse=True)[:10]
    screenCnt = None

    for c in contours:
        peri = cv2.arcLenglth(c, True)
        approx = cv2.approxPolyDP(c, 0.02 * peri, True)

        if len(approx) == 4:
            screenCnt = approx
            break

    if screenCnt is None:
        return None

    rect = cv2.boundingRect(screenCnt)
    px, py, pw, ph = rect
    plate = roi[py:py + ph, px:px + pw]

    plate_h, plate_w, _ = plate.shape
    crop_width = int(plate_w * crop_percent)
    plate_cropped = plate[:, crop_width:plate_w - crop_width]

    return plate_cropped
```

Slika 17 - Tablica vozila izolirana iz ostatka slike

Kod radi tako što iz dobivene slike unutar ROI područja traži predmet 4 rubne točke, tj. pravokutnik. Ukoliko je nađen dio na slici koji zadovoljava taj uvjet program izlazi iz for petlje te primjenjuje rezanje slike kako bi se pojednostavio zadatak očitavanja znakova na tablici.

4.3.2. Python-tesseract

Nakon završenog dijela s OpenCV bibliotekom uspješno je napravljeno pola zadatka senzora – detekcija zauzetosti mjesta. Još je preostala identifikacija parkiranog vozila. Za taj zadatak sam odlučio koristiti Tesseract što je biblioteka otvorenog koda napravljena za optičko prepoznavanje znakova od strane Google-a. Nakon instalacije Tesseract-a u Python okruženju implementirao sam sljedeću funkciju:

```
import cv2
import pytesseract

def ocr(plate):
    gray = cv2.cvtColor(plate, cv2.COLOR_BGR2GRAY)

    _, thresh = cv2.threshold(gray, 0, 255, cv2.THRESH_BINARY +
cv2.THRESH_OTSU)

    config = '--psm 8 -c
tessedit_char_whitelist=ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789-'

    text = pytesseract.image_to_string(thresh, lang="engl")
    text = ''.join(e for e in text if e.isalnum() or e == '-')
    if text:
        return text

    return None
```

Kao izlaz iz funkcije ćemo dobiti sadržaj registracijske oznake koji je očitan sa slike koja je proslijeđena funkciji kod pozivanja iste.

U nastavku se nalazi programski kod koji se nalazi u main.py datoteci. Taj kod je zadužen za procesuiranje parkirnog mjesta, tj. on će koristiti pomoćne funkcije u ostalim datotekama kako bi procesiralo cijelo parkirno mjesto, od zauzetosti do očitavanja registracijskih oznaka auta koje se nalazi na mjestu. Također, u programu se nalazi dio koda koji će označiti mjesto zelenom bojom s natpisom „Slobodno“, tj. crvenom bojom s natpisom „Zauzeto“ i registracijskim oznakama vozila koje se nalazi na tome mjestu.

```
import asyncio
import cv2
import numpy as np
import picamera
import picamera.array
from config import SPACES
from processing import is_space_occupied, extraction
from ocr import ocr
from networking import send_parking_status_update

async def process_parking_space(image, space):
    occupied = is_space_occupied(image, space)


    if occupied and not space['occupied']:
        space['status_changed'] = True
        await asyncio.sleep(5)
        global camera
        with picamera.array.PiRGBArray(camera) as stream:
            camera.capture(stream, 'bgr', use_video_port=True)
            currentImage = stream.array

            x, y, w, h = cv2.boundingRect(space['plate_roi'])
            plate_roi = currentImage[y:y+h, x:x+w]

            plate = extraction(currentImage, plate_roi)
            if plate is not None:
                text = ocr(plate)
                if text:
                    space['last_plate'] = text
                    print(f"Parking Space {space['ID']} License Plate:
{text}")
                    send_parking_status_update(space['ID'], occupied,
space['last_plate'])

    elif not occupied and space['occupied']:
        space['last_plate'] = None
        space['status_changed'] = False
        send_parking_status_update(space['ID'], occupied, None)

    space['occupied'] = occupied

    color = (0, 0, 255) if occupied else (0, 255, 0)
    cv2.polylines(image, [space['poly']], True, color, 2)
    cv2.putText(image, 'Zauzeto' if occupied else 'Slobodno',
tuple(space['poly'][0]), cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 2)
```

```
    if occupied and space['last_plate']:
        cv2.putText(image, space['last_plate'], (space['poly'][0][0],
space['poly'][0][1] - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 2)

async def main():
    global camera
    with picamera.PiCamera() as camera:
        camera.vflip = True
        camera.hflip = True
        with picamera.array.PiRGBArray(camera) as stream:
            while True:
                camera.capture(stream, 'bgr', use_video_port=True)
                image = stream.array

                tasks = [process_parking_space(image, space) for space
in SPACES]
                await asyncio.gather(*tasks)

                cv2.imshow('Parking Lot', image)

                stream.truncate(0)

                if cv2.waitKey(1) & 0xFF == ord('q'):
                    break

    cv2.destroyAllWindows()
```

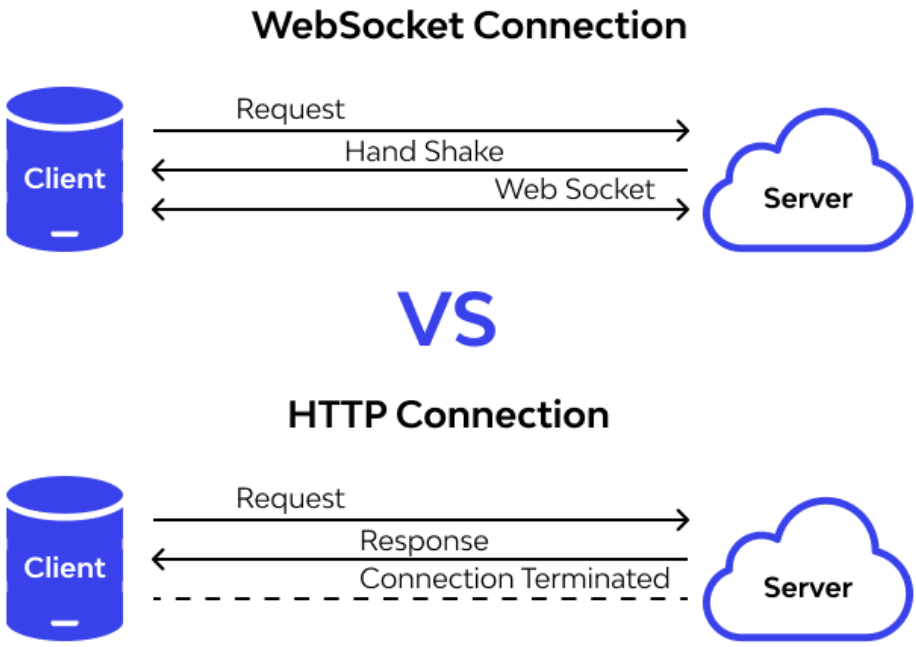
Tijekom izrade *main* funkcije u kojoj će se izvršavati svi potprogrami koristio sam pristup asinkronog programiranja kako bi se smanjilo opterećenje glavne dretve kod procesuiranja slika u pomoćnim potprogramima.

4.3.3. Slanje podataka sa senzora

Još jedan vrlo bitni aspekt IoT uređaja, kako je navedeno u prethodnom poglavlju, je upravo taj prijenos podataka s IoT uređaja na aplikaciju koja obrađuje te podatke. Tijekom odabira načina prijenosa podataka razmatrao sam opciju MQTT protokola, međutim taj protokol je poprilično spor ukoliko bi se morale slati slike automobila u aplikaciju. Razlog slanja slika je što bi se kasnije mogla implementirati funkcionalnost optičkog prepoznavanja znakova i sa serverske strane rješenja kako bi se mogli uspoređivati rezultati jer trenutno rezultati očitavanja nisu 100% točni, što u primjeni ovog rješenja nije baš idealno. Također, moglo bi se implementirati opcija kad osoba traži auto u parkirnom prostoru da mu se ponude slike automobila sa sličnom registracijskom oznakom koju je osoba unijela kod pretraživanja.

Upravo sam ja iz tih razloga odlučio koristiti websocket za prijenos podataka. Prednost websocketa, naprema običnog HTTP prijenosa, je što za websocket nije potrebno konstantno

slati zahtjeve i čekati odgovor nego će se na početku poslati jedan zahtjev od strane servera te će senzor sam slati odgovore kod svake promjene stanja na nekom parkirnom mjestu.



Slika 18 - Usporedba WebSocketa s HTTP-om

(izvor: https://www.wallarm.com, preuzeto: 2024.)

Kako bi implementirao websocket u Pythonu koristio sam postojeću „*socket*“ biblioteku koju sam samo trebao definirati poruku koja se šalje te adresu servera koja prima poruku. Adresa servera se nalazi u config.py datoteci, a poruci se nalazi ID parkirnog mjesta, stanje mjesta (zauzeto ili slobodno), te registracijska oznaka vozila na mjestu.

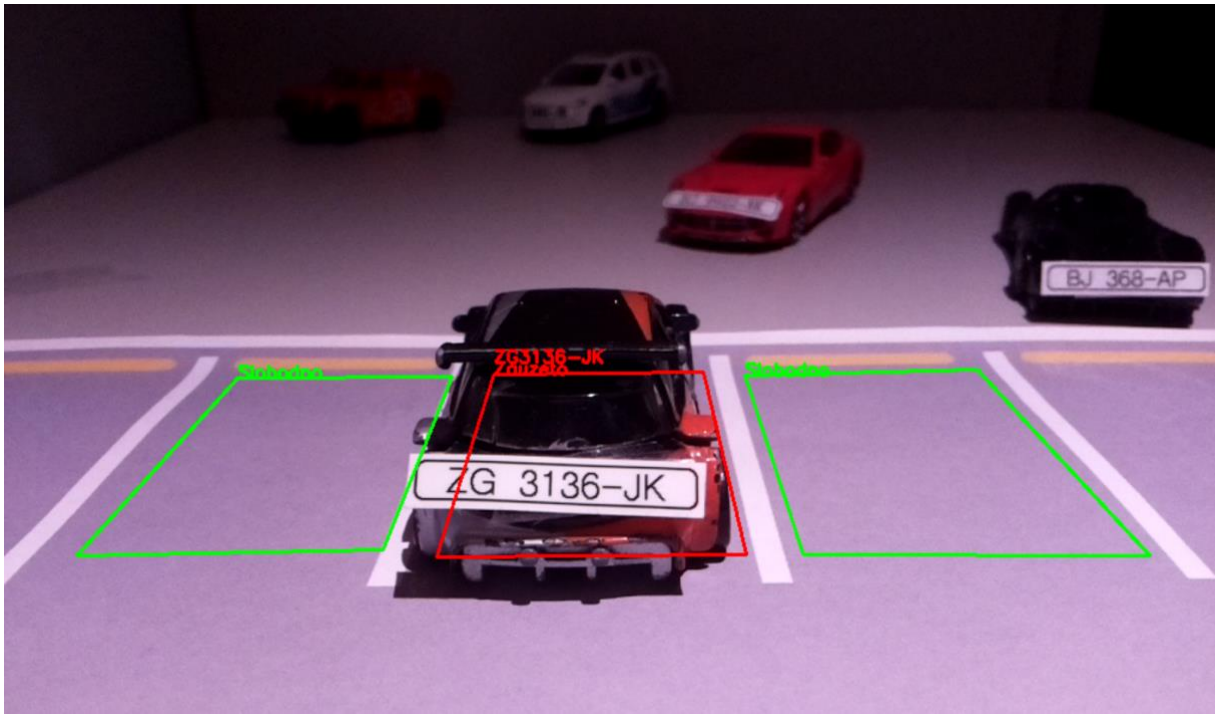
```
import socket
import cv2
import numpy as np
from config import SERVER_ADDRESS

def send_parking_status_update(space_id, occupied, license_plate):

    message = f"{space_id};{occupied};{license_plate if license_plate
else 'None'}<EOF>"

    try:
        with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as sock:
            sock.connect(SERVER_ADDRESS)
            sock.sendall(message.encode())
    except Exception as e:
        print(f"Error sending data: {e}")
```

Senzor šalje podatke u sljedećem obliku:

```
{space_id};{occupied};{license_plate if license_plate else 'None'}<EOF>
```

- space_id (int) – Identifikacijski broj parkirnog mjesta za koje se šalje podatak
- occupied (boolean) – status parkirnog mjesta, true – zauzet, false – slobodan
- license_plate (string) – ukoliko je vozilo parkirano sadrži registracijsku oznaku

S time je uspješno završena implementacija funkcionalnosti IoT senzora za parkirna mjesta. Kako bi mogao provjeriti radili program kao predviđen bez potrebe za serverskom stranom koju ću izraditi tek kasnije napravio sam „prozor“ u kojem se prikazuje uživo prijenos kamere senzora te se označuje zauzetost mjesta i registracijska oznaka ukoliko je mjesto zauzeto. Izgled toga prikaza možemo vidjeti na sljedećoj slici:



Slika 19 – Prikaz kamere senzora s informacijama o zauzetosti

Na slici se može vidjeti označeno svako mjesto koje senzor pokriva (u ovom slučaju se radi o tri parkirna mjesta), te trenutni status mjesta: slobodno tj. zauzeto. Ukoliko je mjesto zauzeto izvršava se prethodno objašnjeno izoliranje tablice te očitavanje znakova s tablice. Ukoliko je očitavanje uspješno izvršeno registracijska oznaka vozila se ispisuje pored statusa parkirnog mjesta.

4.4. Izrada klijentske aplikacije

Nakon što je softverska podrška senzora uspješno dovršena krenuo sam s izradom klijentske aplikacije s kojom će zapravo korisnik sustava vršiti interakciju. Putem klijentske aplikacije će biti moguće izvršiti sve funkcionalnosti sustava kao što su:

- Prikazivanje stanja parkirnog prostora
- Prikazivanje opće zauzetosti parkirnog prostora
- Pretraživanje vozila prema registracijskoj oznaci unutar parkirnog prostora
- Rezerviranje parkirnih mjesta za određena vozila
- Obavještavanje ukoliko se nedozvoljeno vozilo parkira na rezervirano mjesto

Kako se ovdje radi o internoj aplikaciji namijenjenoj za korištenje u kontroliranom okruženju parkirnog prostora, zaključio sam kako je najbolje napraviti stolnu aplikaciju za Windows operativni sustav jer su takva računala već u uporabi na mjestima gdje bi se ovo rješenje koristilo. U ovom slučaju bi klijentska aplikacija služila i kao serverski poslužitelj senzorima jer bi ona komunicirala sa senzorima i prikupljala informacije koje joj senzori šalju putem websocketa. Za izradu desktop aplikacije sam se odlučio za .NET okvir te izradu Windows Forms aplikacije u C# programskom jeziku iz razloga jer sam se prethodno upoznao s navedenom tehnologijom na fakultetu.

4.4.1. Arhitektura klijentske aplikacije

Kao što je i navedeno u samom uvodu 9. poglavlja, stolna aplikacije će biti izrađena u programskom jeziku C# i koristiti će neke od koncepata objektno orijentiranog programiranja (OOP) uz dretvu zaduženu za dohvaćanje i obradu senzorskih podataka koja će raditi u pozadine aplikacije kako se ne bi „zagušivao“ glavni program. Naime, odvojena dretva kada primi zahtjev za kreiranje websocketa sa senzorom će ga kreirati i zatim, bez potrebe za ponovnom uspostavom komunikacije, započeti komunikaciju sa senzorom tj. započeti će prihvaćati poruke o promjeni stanja na parkirnom mjestu od strane senzora.

Na dolje prikazanom dijagramu klasa mogu se vidjeti sve ključne klase za izradu serverske, tj. klijentske aplikacije. Na vrhu dijagrama se nalazi *SensorDataReceiver* klasa koja u zasebnoj dretvi zapremljuje i obrađuje zahtjeve senzora. Kada klasa zaprimi zahtjev izmjenjuje stanje mjesta u *frmMainWindow* klasi gdje je zatim vizualno vidljivo stanje mjesta na panelu koji izmjenjuje boju iz crvenog za zauzeto i zelenog za slobodno mjesto, odnosno

žutog ukoliko se na rezervirano mjesto parkiralo nedozvoljeno vozilo kako bi upozorilo operatera.

U klasi SensorDataReceiver se kreira funkcija *StartListening()* u zasebnoj dretvi na sljedeći način:

```
public SensorDataReceiver()
{
      receiveThread = new Thread(StartListening);
      receiveThread.Start();
}
```

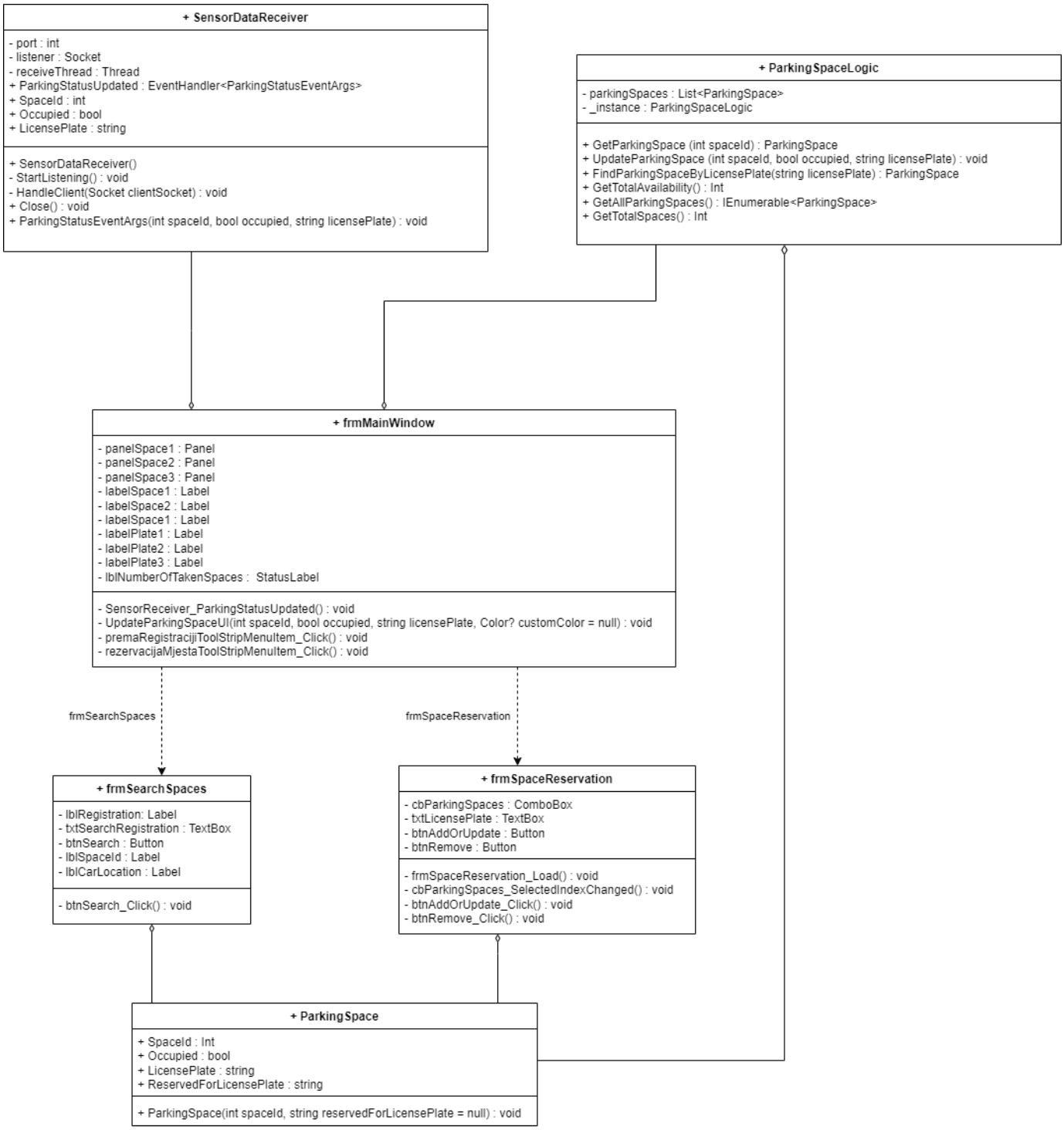
Funkcija *StartListening(),* koja je prikazana ispod, pokazuje kako će server prihvatiti konekciju od bilo koje IP adrese na predefiniranom portu, koji je u ovom slučaju *55655*. Te će za svakog klijenta u zasebnoj dretvi obrađivati zahtjev u funkciji *HandleClient()* u kojoj se primaju poruke senzora te parsiraju podatci iz poruke.

```
        private void StartListening()
         {
             IPEndPoint localEndPoint = new IPEndPoint(IPAddress.Any,
port);
             listener = new Socket(AddressFamily.InterNetwork,
SocketType.Stream, ProtocolType.Tcp);

             try
             {
                 listener.Bind(localEndPoint);
                 listener.Listen(10);

                 while (true)
                 {
                     Console.WriteLine("Waiting for a connection...");
                     Socket handler = listener.Accept();
                     Thread clientThread = new Thread(() =>
HandleClient(handler));
                     clientThread.Start();
                 }
             }
             catch (Exception ex)
             {
                 Console.WriteLine("Error: " + ex.Message);
             }
         }
```

Zatim se s dobivenim podacima ažurira *frmMainWindow()* forma. Također su uz glavnu formu dostupne i *frmSearchSpaces()* putem koje je moguće pretraživanje parkiranih automobila unutar parkirnog prostora putem registracijske oznake, te *frmSpaceReservation()* putem koje je moguća izmjena i dodavanje registracijske oznake automobila za rezervaciju parkirnog mjesta za određeno vozilo. Također, je napravljena podatkovna klasa *ParkingSpace()* koja se u ostatku aplikacije kako bi se definiralo parkirno mjesto. Svi atributi koji se nalaze u toj klasi su vidljivi na kompletnom dijagramu klasa prikazanom na dnu poglavlja. Sve funkcije za pretraživanje, izmjenu, dohvaćanje te inicijalizaciju parkirnih mjesta i sličnoga se nalaze u statičnoj klasi *ParkingSpaceLogic()* gdje su ključniji dijelovi koda prikazani u isječku koda ispod:

```
public ParkingSpace GetParkingSpace(int spaceId)
{
    return parkingSpaces.FirstOrDefault(space => space.SpaceId == spaceId);
}

public void UpdateParkingSpace(int spaceId, bool occupied, string licensePlate)
{
    var space = GetParkingSpace(spaceId);
    if (space != null)
    {
        space.Occupied = occupied;
        space.LicensePlate = licensePlate;
    }
}

public ParkingSpace FindParkingSpaceByLicensePlate(string licensePlate)
{
    return parkingSpaces.FirstOrDefault(space => space.Occupied &&
space.LicensePlate == licensePlate);
}

public int GetTotalAvailability()
{
    return parkingSpaces.Count(space => !space.Occupied);
}

public IEnumerable<ParkingSpace> GetAllParkingSpaces()
{
    return parkingSpaces;
}

public int GetTotalSpaces()
{
    return parkingSpaces.Count;
}
```

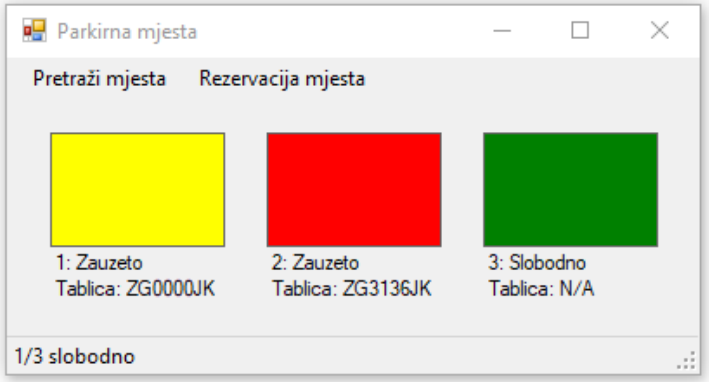
Te se cijeli dijagram klasa klijentske aplikacije, koji je spominjan tijekom ovog poglavlja, nalazi na idućoj stranici. U dijagramu klasa je moguće vidjeti sve klase koje su potrebne za funkcioniranje klijentske strane aplikacije.



Slika 20 - Dijagram klasa klijentske stolne aplikacije

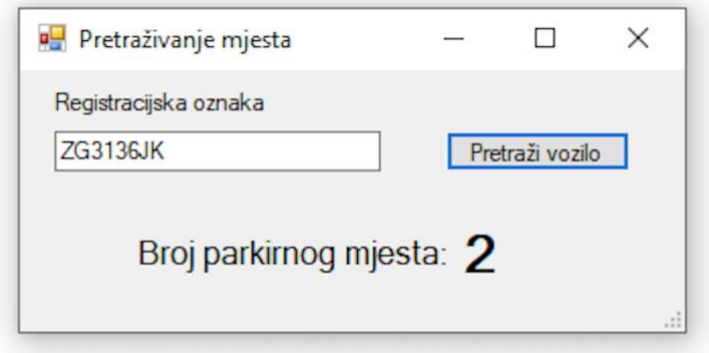
4.4.2.Izrađena klijentska aplikacija

Sama aplikacija je vrlo jednostavna te će u ovom poglavlju biti prikazane kako izgleda trenutna inačica aplikacije. Naime, iako ima još puno mjesta za napredak kada bi se išlo u izradu produkcijskog rješenja, ova aplikacije je napravljena kao PoC (*engl. Proof of Concept*) kako bi se pokazalo kako bi funkcionirala u produkcijskoj okolini ukoliko bi se krenulo u izradu iste. Na donjoj slici je početni prikaz koji se otvori nakon početnog otvaranja aplikacije. Taj prikaz nam omogućuje brzi uvid u stanje parkirnih mjesta kao što su: opću raspoloživost prostora, vozila na mjestima te koje je stanje parkirnih mjesta. Stanje mjesta je vrlo brzo vidljivo bojom mjesta, ukoliko je zeleno mjesto je slobodno, ukoliko je crveno onda je zauzeto, a ukoliko je žuto radi se o grešci na parkirnom mjestu. Greške trenutno mogu biti da je parkirano vozilo kojom nije bilo moguće očitati registracijsku oznaku ili da se radi o vozilu koje je nedozvoljeno parkirano tj. parkiralo se na mjestu rezerviranom za drugo vozilo.



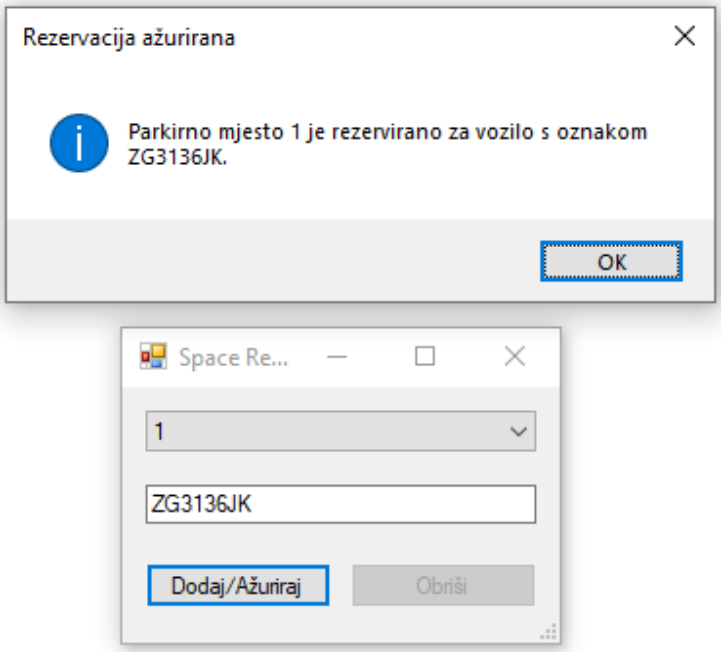
Slika 21 - Početni prikaz aplikacije

Također, na alatnoj traci početnog prozora se nalaze opcije za pretraživanje parkiranih vozila i rezerviranje parkirnog mjesta za određeno vozilo. Odabirom opcije za pretraživanje otvara nam se dolje vidljiv prozor:



Slika 22 - Prozor za pretraživanje parkiranih vozila

Nakon upisivanja traženog vozila ispisati će se broj mjesta na kojem se nalazi, ukoliko vozilo nije moguće pronaći ispisati će se greška o istome.
Te za kraj, ukoliko se na glavnom prozoru odabere opcija *Rezervacija mjesta* otvara se prozor za rezervaciju koji izgleda ovako:



Slika 23 - Prikaz rezervacije parkirnog mjesta za vozilo

Na prethodnoj slici je prikazan postupak rezervacije parkirnog mjesta s oznakom „1“ za vozilo s registracijskom oznakom „ZG3136JK“ gdje se nakon uspješne rezervacije ispisuje poruka o istome.

4.4.3. Alternativna arhitektura aplikacije

Bolji način izrade klijentske, odnosno serverske strane rješenja bi bilo fizički odvojiti serversku i klijentsku stranu. Preporučljivo je uvijek odvojiti serverski dio aplikacije, od dijela koji je dostupan korisniku, iz više razloga:

- Sigurnost
- Redundancija
- Dostupnost

Ukoliko bi ovo rješenje išlo u produkcijsko okruženje serverska strana bi se izvršavala na odvojenom računalu, odnosno serveru (poželjno više njih istovremeno). Serverska strana bi zadržala neke od funkcija trenutne aplikacije kao što je komunikacija sa senzorima, upravljanje

podacima i slično dok bi klijentska strana koja bi se izvršavala na računalu korisnika bila zaduženo isključivo za grafički prikaz podataka koje bi serverska strana poslužila. Prednost je fizičko odvajanje serverskog računala u sigurnu prostoriju gdje bi bio strogo kontrolirani pristup računalu te bi se isključila mogućnost slučajnog gašenja računala od strane korisnika. Ovaj pristup je izrazito bitan jer je serversko računalo tzv. jedna točka otkazivanja, gdje bi se zbog nedostupnosti servera cijeli parkirni prostor morao privremeno zaustaviti poslovanje dok se kvar ne otkloni. Osim toga, razdvajanje servera i klijenta omogućuje i bolje upravljanje opterećenjem. U slučaju povećanog broja korisnika ili senzora, opterećenje se može rasporediti na više servera (*engl. load balancing*), čime se osigurava da sustav ostane dostupan. Takav pristup također omogućava implementaciju rezervnih servera (*engl. failover servers*), koji se mogu automatski aktivirati u slučaju pada primarnog servera, čime se dodatno povećava pouzdanost i dostupnost sustava.

Uvođenjem ove arhitekture u produkcijsko okruženje značajno bi se poboljšala stabilnost, sigurnost i skalabilnost sustava za upravljanje parkirnih prostora, što bi osiguralo dugoročnu održivost i fleksibilnost sustava u skladu s rastućim potrebama korisnika i novim tehnološkim izazovima.

Iako sam razmatrao razdvajanje klijentske i serverske aplikacije, odlučio sam se za integrirano rješenje kako bih zadržao jednostavnost ovog završnog rada. Integrirana aplikacija pojednostavljuje razvoj i održavanje sustava, omogućujući brže testiranje i ispravljanje pogrešaka. Također, za manji prototip, ovakav pristup je praktičniji za krajnjeg korisnika jer olakšava instalaciju i korištenje bez potrebe za postavljanjem zasebnog servera. Time sam mogao fokusirati rad na ključnim funkcionalnostima sustava, što je omogućilo uspješnu realizaciju projekta.

4.5. Alternativno rješenje problema

Jedno od alternativnih načina rješavanja ovog problema, kojeg sam također razmatrao, je bio korištenje LPR (engl. License Plate Recognition) kamere koja se nalazi na samom ulazu te će uvijek na optimalniji način moći skenirati registracijske oznake vozila te spremiti registraciju vozila zajedno s markom, modelom, bojom vozila i još sličnim podatcima o vozilu koji bi daljnje pomogli kod praćenja vozila u parkirnom prostoru.

Kada bi se sustav sastojao od samo jedne LPR kamere na ulazu i još određeni broj kamera u samom parkirnom prostoru koji je dovoljan kako bi se pokrili svi dijelovi prostora, tada bi se teoretski vozilo moglo pratiti od samog ulaza u parkirni prostor do parkiranja na

određeno mjesto. Dostupnost samog mjesta bi se i dalje morala detektirati klasičnim ultrasoničnim senzorima, ali preko kamere bi se moglo zaključiti koje vozilo se nalazi na tom mjestu. Prednost takvog pristupa je puno jeftinija instalacija kod postojećih parkirnih prostora jer je većina garaža već opremljeno LPR kamerama na ulazu tj. izlazu iz garaže i imaju nekolicinu kamera u samom garažnom prostoru. Eventualno bi bilo potrebno još dodati kamere ako neki dijelovi garaže nisu dobro pokriveni te mapirati cijeli garažni prostor. Iako ovo rješenje ima prednosti s hardverske strane, smatrao sam kako će sa softverske strane ovaj pristup biti puno zahtjevniji zbog praćenja vozila unutar parkirnog prostora. Iako bi takav pristup možda u realnom svijetu bio puno bolji i jeftiniji za krajnjeg investitora.

5. Zaključak

Cilj rada bio je prikazati koncept rješenja (engl. Proof of Concept - PoC) koji bi mogao značajno unaprijediti upravljanje parkirnim prostorima. Razvijeno rješenje koristi kameru umjesto tradicionalnih senzora za detekciju zauzetosti parkirnih mjesta, omogućavajući ne samo identifikaciju slobodnih i zauzetih mjesta, već i prepoznavanje registarskih oznaka vozila koje se nalazi na parkirnom mjestu. Izradom ovog rada pokazano je da IoT tehnologije mogu ponuditi efikasnija i fleksibilnija rješenja u odnosu na tradicionalne metode. Rješenje bi moglo smanjiti potrebu za višestrukim senzorima, pokriti više parkirnih mjesta s jednom kamerom, te omogućiti jednostavnije nadgledanje i upravljanje parkirnim prostorom putem centraliziranog sustava.

Nadalje, iako je cilj ovog završnog rada bio razviti funkcionalan prototip, postoji mnogo prostora za buduća poboljšanja i proširenja. Jedno od ključnih područja za daljnji razvoj je povećanje skalabilnosti sustava kako bi mogao podržavati veće parkirne prostore s više senzora i kamera. Također, implementacija naprednih algoritama za analizu podataka mogla bi omogućiti predikciju zauzetosti parkirnih mjesta, što bi dodatno unaprijedilo funkcionalnost sustava.

Osim toga, jedna od najvećih mana trenutnog senzora koje sam primijetio tijekom testiranja i implementacije trenutnog rješenja jest – ograničena funkcionalnost prilikom slabijeg osvjetljenja. To je vrlo bitno ukoliko se radi o mračnom prostoru koje je osvjetljeno umjetnim osvjetljenjem kao što su parkirne garaže, gdje bi ti senzori bili najviše u uporabi. Naime, kod izrade senzora koji bi se zapravo koristio u realnom svijetu moralo bi se provesti puno više vremena i resursa u projektiranje i izradu samog senzora kao što je i navedeno tijekom ovog završnog rada.

Zaključno, iako je ovo rješenje trenutno na razini prototipa, njegovi rezultati i potencijal ukazuju na to da bi uz daljnji razvoj mogao postati učinkovito i pouzdano rješenje za moderne parkirne sustave. Kroz buduća poboljšanja i prilagodbe, ovaj sustav može doprinijeti značajnom unapređenju načina na koji se upravlja parkirnim prostorima u pametnim gradovima.

Popis literature

[1] H. R. Dongxiang Yang, »The Research on The Technology of Internet of Things And Embedded System.,« 2017.

[2] M. Š. I. D. M. J. M. Š. B. J. M. U. D. B. Marko Ševrović, »Analiza prometne potrebe izgradnje javnih parkirališnih garaža na području Donjeg Grada s ciljem unaprjeđebha idrživog prometnog sustava grada Zagreba,« Fakultet promethin znanosti, Zagreb, 2009.

[3] Z. M. D. J. P. E. Barro PA, »A Smart Cities LoRaWAN Network Based on Autonomous Base Stations (BS) for Some Countries with Limited Internet Access.,« *Future Internet,* 2019.

[4] S. J. D. a. A. Varol, »Comparison of Zigbee, Z-Wave, Wi-Fi, and Bluetooth Wireless Technologies Used in Home Automation,« *7th International Symposium on Digital Forensics and Security (ISDFS),* 2019.

[5] G. M. D. T. Forecast, »Cisco visual networking index: global mobile data traffic forecast update, 2017–2022.,« 2022.

[6] A. W. K. L. R. Laghari, A Review and State of Art of Internet of Things (IoT), 2021.

Popis slika

Dodaci

Programski kôd razvijen u okviru izrade završnog rada koji je referenciram u određenim dijelovima se nalazi na GitHub repozitoriju dostupnom na slijedećoj poveznici:

https://github.com/mblagajce21/IoT-Parking-System