SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 6351

Kontrola ulaza korištenjem beskontaktnih kartica

Filip Ptiček

Zagreb, lipanj 2019.

Umjesto ove stranice umetnite izvornik Vašeg rada.
Da bi ste uklonili ovu stranicu obrišite naredbu `\izvornik`.

SADRŽAJ

1. Uvod

U današnjem svijetu pokušavamo povezati sve više stvari, uređaja i pomagala s tehnologijom. Na taj način se pokušava olakšati korištenje i mogućnost automatizacije pomoću jednog centralnog mjesta. Takva rješenja nam omogućuju korištenje jednog uređaja za upotrebu u plaćanju, identifikaciji te mnogim drugim stvarima. Neki od centralnih mjesta su mobilni telefoni te beskontaktne kartice koje se nalaze u džepu većine današnjeg stanovništva.

Kontrola ulaza je jedan od poslova koji se od antičkih civilizacija prepuštalo da obavlja čovjek. Vrata koja su koristila mehanizme ključa i brave nisu dopuštale odstupanje od te norme. Tek pojavom kartica s magnetskom trakom došlo je do promjena. Takve kartice dopuštale su da se svakoj osobi dodijeli jedinstveni identifikator. Pomoću čitača kartica koje su sadržavale spremljene identifikatore moglo se dopustiti ulaz samo određenim osobama i ujedno voditi evidencija pristupa. Jedna od mana ovakve tehnologije je što korisnik treba karticu dovesti u direktan kontakt s čitačem te mogućnost jednostavnog repliciranja informacija spremljenih na njima.

Tehnologije kao što su radio-frekvencijska identifikacija (**RFID**) ,nastala 1983. godine, te beskontaktna tehnologija niže frekvencije (**NFC**), nastala 2003. godine, dozvoljavaju udaljenu komunikaciju između čitača i kartice ili oznake. U današnje vrijeme zamijenile su upotrebu kartica s magnetskom trakom zbog veće sigurnosti i u slučaju RFID-a mogućnosti za praćenjem položaja kartice ili oznake korištenjem komunikacije velikih frekvencija.

Danas se najčešće za kontrolu ulaza koristi beskontaktna tehnologija niže frekvencije zbog raširenosti u mobilnim telefonima i u slučaju studentske akademske zajednice Republike Hrvatske u njihovim akademskim iskaznicama, što ne zahtijeva uvođenjem posebnih oznaka kao u slučaju radio-frekvencijske komunikacije.

2. Opis problema

Sustav za kontrolu ulaza treba se sastojati od nekoliko dijelova:

- Softverskog dijela za autentifikaciju
- Hardverskog dijela za očitavanje NFC kartica
- NFC kartice
- Mehaničke brave za otvaranje ulaza

Potrebno je pronaći adekvatni uređaj na kojem bi se izvršavala autentifikacija te uređaj za čitanje kartica. Oba uređaja trebaju biti kompaktni, laki za montiranje i neinvazivni. Proces kojim bi sustav trebao raditi je:

- Korisnik prilaže NFC karticu u bliski domet čitača
- Čitač čita identifikacijsku informaciju s NFC kartice
- Čitač šalje informaciju uređaju za autentifikaciju
- Uređaj pomoću programa koji se izvodi na njemu radi autentifikaciju korisnika
- Program zapisuje evidenciju o čitanju
- Uređaj vraća čitaču informaciju ima li korisnik autorizaciju za otvaranje ulaza
- Čitač ovisno o odgovoru uređaja otvara mehaničku bravu ili signalizira korisniku da nema pristup

3. Korišteni razvojni alati

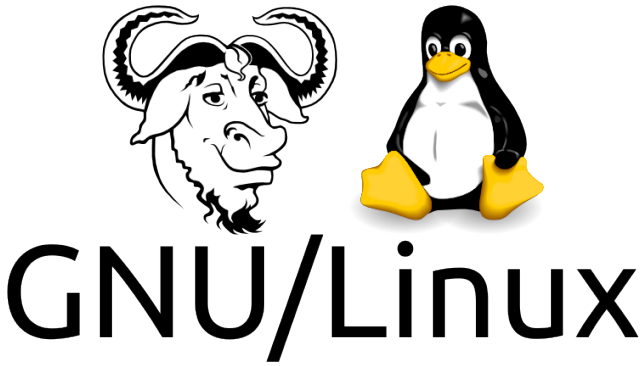
3.1. GNU/Linux

GNU/Linux je operacijski sustav temeljen na Linux jezgri i GNU programskoj potpori te je temeljen na principima otvorenog koda. Jezgra je nastala 1991. godine od strane Linusa Torvaldsa. Razvijana je po uzoru na UNIX operacijski sustav. GNU je sloj iznad jezgre koji se sastoji od skupa programskih paketa koji omogućuju da zadnji aplikacijski sloj cijelog operacijskog sustava funkcionira.

GNU/Linux danas pokreće većinu poslužiteljskih računala, mobilnih telefona, ugrađenih sustava i sve više osobnih računala. Raznovrsnost i rasprostranjenost ovog operacijskog sustava nam omogućuje da razvijamo aplikacije koje će se izvršavati na što više uređaja.

Operacijski sustav dolaze putem različitih distribucija. Neke od popularnih su: Debian, Ubuntu, SUSE, Red Hat Enterprise Linux te onih namijenjenih za slabija i manja računala poput Raspbiana temeljenog na Debianu.

Za razvoj ovog rada korištena je distribucija Ubuntu zbog svoje dobre programske i korisničke podrške. Za konačnu implementaciju i izvršavanje se koristi Raspbian koji se pokreće na Rapberry Pi-u.



Slika 3.1: Operacijski sustav GNU/Linux

3.2. Raspberry Pi

Raspberry Pi je serija računala malih dimenzija, velikih performansi i niske cijene. Ova računala se koriste zbog svojeg dizajna u razvoju, edukaciji i kao poslužitelji za male projekte.

U ovom radu korišten je Raspberry Pi 2 Model B, a njegove karakteristike su:

- 900MHz četverojezgreni ARM Cortex-A7 procesor
- 1GB RAM
- 100BASE Ethernet
- 4 USB priključka
- HDMI priključak
- Micro SD utor za karticu

Razlog odabira ovog računala je njegova mala veličina, jednostavnost korištenja i mala potrošnja električne energije. Na računalu se izvršava GNU/Linux distribucija naziva Raspbian.

Slika 3.2: Računala Raspberry Pi

3.3. Gemini 2000 Orbit IP

Gemini 2000 Orbit IP je beskontaktni čitač niže frekvencije. Čitać je napajan preko Etherneta (eng. *Power over Ethernet*) (PoE) standard IEEE 802.3af-2003 te se može koristiti PoE mrežni preklopnik ili aktivni 48V ubrizgač. Čitač također koristi Ethernet priključak za komunikaciju s web poslužiteljem.

Orbit IP radi kao samostalan web klijent te komunicira s web poslužiteljem tako da šalje HTTP zahtjeve i tada čeka povrati odgovor također u obliku HTTP zahtjeva.

Čitač podržava ISO 14443 Tip A i B oznake.

Čitač je odabran za ovaj rad zbog jednostavnog razvoja, jer se koristi jednostavna HTTP komunikacija s jednostavnim naredbama, te mogućnosti da se rade složeniji sustavi s više čitača jer se spaja u mrežu.

Kod implementacije rada napravljena je mreža samo između Raspberry Pi-a i čitača te tako cijeli sustav ostaje izoliran.

Slika 3.3: Gemini 2000 Orbit IP

3.4. Python 3

Python je interpretativni programski jezik nastao od strane Guido van Rossuma 1991. godine. Najčešće je zbog svoje jednostavnosti i proširivosti različitim bibliotekama popularan u različitim područjima razvoja. Neka od područja su:

- Skriptiranje - kao interpretativni jezik koristi se za podešavanje sustava te pisanje jednostavnih skripti za različite namjene
- Matematika - zbog velikog izbora matematičkih bibilioteka Python je jezik koji se često koristi za statistiku i strojno učenje
- Razvoj web aplkacija - poslužiteljski dio aplikacija sve više je razvijan u Pythonu. Interpretativni način rada iako nije najbrži dozvoljava brži razvoj i testiranje aplikacija. Također zbog mnogo razvojnih okvira koji nude brzi razvoj i proširivost

Za razvoj ovog rada odabran je Python zbog svoje rasprostranjenosti, količine bibilioteka i neovisnosti o sustavu na kojem se izvršava. Python također karakterizira i čitljivost koda te kao takav je savršen za razvoj manjih projekata.

Slika 3.4: Programski jezik Python

3.5. TinyDB

TinyDB je mala baza podataka orijentirana na dokumentima. Podaci se spremaju u jednu datoteku koja je zapisana u JSON formatu. Zbog malog broja informacija koje se moraju spremati u bazu, nema potrebe za bazama podataka koje imaju više mogućnosti, brže su, ali i zauzimaju više memorije kod izvršavanja.

Slika 3.5: Baza podataka TinyDB

3.6. Vim

Vim je tekstualni editor koji zbog svoje velike mogućnosti proširenja i efikasnosti kod pisanja programa je savršeni alat za razvijanje programskih rješenja. Neke od njegovih glavnih značajka su:

- dosljedno, više razinsko stablo poništavanja
- veliki izbor nadogradnji
- podrška za stotine programskih jezika i datotečnih formata
- snažna pretraga i promjena teksta
- mogućnost integracija s mnogo alata

Slika 3.6: Tekstualni editor Vim

3.7. Git

Git je besplatan distiribuirani sustav za upravljanje izvornim kodom nastao 2005. godine od strane Linus Torvalds. Git kao alat služi da bi se efikasno i lako pratile promjene nastale u razvoju programskog kod.

U ovom projektu git-ova glavna svrha je bila spremanje promjena te mogućnost lakog dohvaćanja istog na drugom računalu više nego kao sustav za kontrolu verzijama. Upravitelj repozitorija korišten je GitHub.

Slika 3.7: Sustav za upravljanje izvornim kodom Git

4. Beskontaktna tehnologija niže frekvencije (NFC)

Beskontaktna tehnologija niže frekvencije, (eng. *Near field communication*) ili skraćeno NFC je vrsta beskontaktne komunikacije između uređaja poput mobilnih telefona i beskontaktnih kartica. Beskontaktna komunikacija dozvoljava komunikaciju na male udaljenosti bez potrebe da uređaji dolaze u neposredni doticaj.

NFC dozvoljava uređaju, koji služi kao čitač, tj. ispitivač, proizvodi aktivno radio frekvenciju što mu dozvoljava da komunicira s ostalim NFC kompatibilnim uređajima ili karticama. Pasivni uređaji poput beskontaktnih kartica i oznaka, imaju spremljenu informaciju te tu informaciju razmjenjuju s čitačima, ali ne dozvoljavaju čitanje informacija drugih uređaja. Kod komunikacije dva aktivna uređaja postoji obostrana komunikacija slanja i primanja informacija.

Integracija beskontaktnih tehnologija u kreditne, putničke i kuponske kartice dozvoljava korisnicima da obavljaju plaćanja, ukrcavanje na javni prijevoz i razmjenu informacija preko jednostavnog približavanja kartica. Također postoji mogućnost integracije više usluga preko mobilnih telefona te tako eliminirati korištenje različitih kartica za različite usluge.

U današnje vrijeme sve više poduzeća implementira beskontaktne tehnologije u svoje usluge, uključujući implementacije virtualnih kartica koje dozvoljavaju plaćanjem na kartičnim terminalima pomoću mobilnih telefona.

4.1. Povijest

Beskontaktna tehnologija niže frekvencije postoji na temeljima radio-frekvencijske identifikacije (RFID). NFC je podset RFID-a s kraćim dometom zbog sigurnosnih razloga.

2004. godine Nokia, Sony i Philips zajedno su formirali NFC Forum. Njihov zajednički cilj je bio promoviranje sigurnosti, jednostavnosti korištenja i popularnosti

Slika 4.1: Upotreba NFC-a

beskontaktne tehnologije niže frekvencije. Zaduženje Foruma je održavanje standarda koji dozvoljava da tehnologija može funkcionirati između sva uređaja. Ako netko želi proizvesti NFC uređaj potrebno je slijediti standarde postavljene od strane NFC Foruma. To osigurava da korisnik s bilo kojim NFC uređajem može komunicirati s nekim drugim NFC uređajem.

Iako je NFC Forum formiran 2004. godine, prve specifikacije za NFC oznake su se pojavile 2006. godine. NFC oznake su mali objekti koji sadrže informacije koje NFC čitaći mogu pročitati. Informacije na oznakama se u većini slučajeva mogu samo čitati, ali postoje i oznake u koje se mogu upisati nove ili promijeniti stare informacije.

Prvi mobilni telefon koji je sadržavao NFC sposobnosti je bila Nokia 6131 NFC predstavljena 2006. godine. Sazrijevanjem tehnologije dolazile su i nove specifikacije koje su osim podržavanja plaćanja dozvoljavale i razmjenu slika, videa i ostalih informacija. Danas je NFC tehnologija dostupna na većini mobilnih uređaja, od telefona, pametnih satova do integracije u automobile.

Slika 4.2: NFC Forum

4.2. Tehnološki standard

Kada se razvijaju NFC uređaji potrebno je pratiti NFC standarde. Standardi postoje da sve vrste NFC uređaja mogu međusobno komunicirati kako oni dizajnirani u prošlosti tako i u budućnosti. Postoje dvije vrste glavnih specifikacija za NFC tehnologiju:

- ISO/IEC 14443 koja definira identifikacijske kartice koje spremaju informacije, poput NFC oznaka
- ISO/IEC 18000-3 koja definira radio-frekvencijsku identifikacijsku komunikacijsko NFC uređaja

ISO/IEC 18000-3 je internacionalni standard za sve uređaje koji komuniciraju bežično na frekvenciji 13.56MHz. Uređaji trebaju biti na minimalnoj udaljenosti od 4cm prije nego dolazi razmjene informacija. Ovi standardi opisuju način na koji čitač i NFC oznaka s koje se čita trebaju komunicirati međusobno.

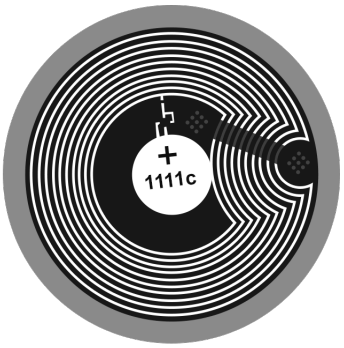
Čitač šalje signal oznaci. Ako su uređaji dovoljno blizu jedan drugom oznaka postaje napajana preko signala čitača. To dozvoljava da oznaka bude malena i bez potrebe da sadrži bateriju ili svoj vlastiti izvor napajanja.

Ta dva uređaja kreiraju visoko frekvencijsko magnetsko polje između zavojnica uređaja. Jednom kada je polje uspostavljeno, dolazi do konekcije i razmjene informacija između čitača i oznake. Čitač šalje prvu poruku oznaci kako bi doznao vrstu komunikacije koju oznaka koristi, tip A ili tip B. Kada oznaka odgovori čitač šalje prvu naredbu koja mora odgovarati specifikaciji.

Oznaka nakon primitka naredbe provjerava je li ona ispravna. Ako nije, oznaka ne odgovara. U protivnom odgovara s zahtijevanom informacijom. Kod nekih komunikacija poput kartičnih plaćanja dolazi do uspostave sigurnog komunikacijskog kanala i sve informacije poslane su enkriptirane.

NFC oznake rade na principu da u jednom trenutku mogu samo primati ili slati informacije, dok čitači mogu primati informacije i slati naredbe istovremeno. Naredbe se šalju s čitaća preko modulacijskog faznog podrhtavanja (eng. *phase jitter modulation*) (PJM) kako bi modificirao okružujuće polje i poslao signal. Oznaka odgovara koristeći induktivnost kako bi poslala naboj preko svojih zavojnica.

Prateći ove specifikacije osiguravamo da svi NFC uređaji mogu međusobno komunicirati.



Slika 4.3: Unutrašnjost NFC oznake

4.3. Sigurnosni problemi

Korisnici NFC tehnologije se naravno pitaju koji su sigurnosni rizici pogotovo oni koji ju koriste za kartična plaćanja. Jesu li njihove informacije sigurne i otporne na krađu? U nastavku su neki od sigurnosnih problema koji se mogu pojaviti i kako NFC tehnologija ih sprječava.

4.3.1. Prisluškivanje

Prisluškivanje se dešava kada osoba prati komunikaciju između čitača i oznaka. Nije potrebno pratiti svaki signal da bi se prikupila privatna informacija. Postoje dvije metode kako spriječiti prisluškivanje:

- udaljenost na kojoj NFC radi. Kako NFC radi na maloj udaljenosti prisluškivanje se može desiti na jako malom području.
- Sigurni kanali. Kada je uspostavljen sigurni kanal, sve informacije koje se razmjenjuju su enkriptirane i samo ovlašteni uređaji ih mogu dekriptirati. Potrebno je samo provjeriti koriste li čitaći sigurne kanale.

4.3.2. Korupcija i manipulacija podataka

Korupcija i manipulacija se dešavaju kada osoba manipulira informacijama koje se šalju čitaću ili posreduje informacijama tako da budu iskvarene i beskorisne kada dođu do čitača. Kako bi se spriječila korupcija i manipulacija koriste se sigurni kanali. Neki uređaji mogu prepoznati takve napade i spriječiti ih prije nego što se dogode.

4.3.3. Presretanje

Presretanje je slično manipulaciji podataka. Kod presretanja postoji posrednik koji čita, mijenja sve informacije koje se razmjenjuju između čitača i oznake. Ovakva vrsta napada je složena za izvesti i rjeđe se dešava. Kako bi se spriječio jedan uređaj mora djelovati aktivno, a drugi pasivno. To znači da jedan šalje, a drugi prima informacije umjesto da oba šalju i primaju informacije.

4.3.4. Krađa

Ako dođe do krađe mobilnog telefona ili kartice kradljivac može lako oponašati osobu te tako dobiti pristup plaćanjima i ulazima. Zato je potrebno upotrijebiti dodatne mjere sigurnosti kao postavljanje sigurnosnih lozinka na svoje mobilne uređaje ili u slučaju kartica i oznaka prijaviti krađu svim administratorima sustava kod kojih je ta kartica ili oznaka zabilježena.

4.4. Jedinstveni identifikacijski broj (UID)

5. Arhitektura i dizajn sustava

U ovome poglavlju objašnjena je cijela struktura sustava za kontrolu ulaza korištenjem beskontaktnih kartica. Sustav se sastoji od:

- sustava za autentifikaciju
- Orbit IP čitača

Sustav je zamišljen kao jednostavan sustav za kontrolu ulaza koji omogućuje autentifikaciju preko beskontaktnih kartica koje sadrže NFC tehnologiju. Naravno postoji mogućnost autentifikacije i mobilnim telefonima koji podržavaju NFC.

5.1. Struktura programske podrške

Programska podrška se sastoji od skripte za pokretanje aplikacije za kontrolu ulaza, aplikacije za kontrolu ulaza, aplikacije za unošenje identifikatora u bazu podataka i baze podataka.

```
NFC_Handler
├── database_input.py
├── db.json
├── nfc_handler.py
└── startup.sh
```

Slika 5.1: Struktura programske podrške

5.2. Baza podataka

Baza podataka je jednostavna datotečna baza u JSON formatu. Nalazi se u datoteci **db.json**.

JSON je tekstualni format koji je jezično neovisan, ali koristi konvencije koje su poznate većini programera.

JSON podaci se spremaju u obliku {'ime' : 'vrijedost'} koje predstavljaju jedan objekt. Objekti su međusobno odijeljeni zarezima.

Kod baze podataka ovog rada postoji glavni objekt **_default** koji predstavlja korijen te sadrži objekte nizane sljednim brojevima. Svaki broj sadrži objekt imena **uid** (eng. *Unique ID*) i vrijednost koja predstavlja identifikacijski broj NFC oznake. Identifikacijski broj oznake sadrži vrijednosti zapisane u **heksadecimalnom** formatu, a veličina identifikatora ovisi o proizvođaču, ali je najčešće veličine 7 bajta.

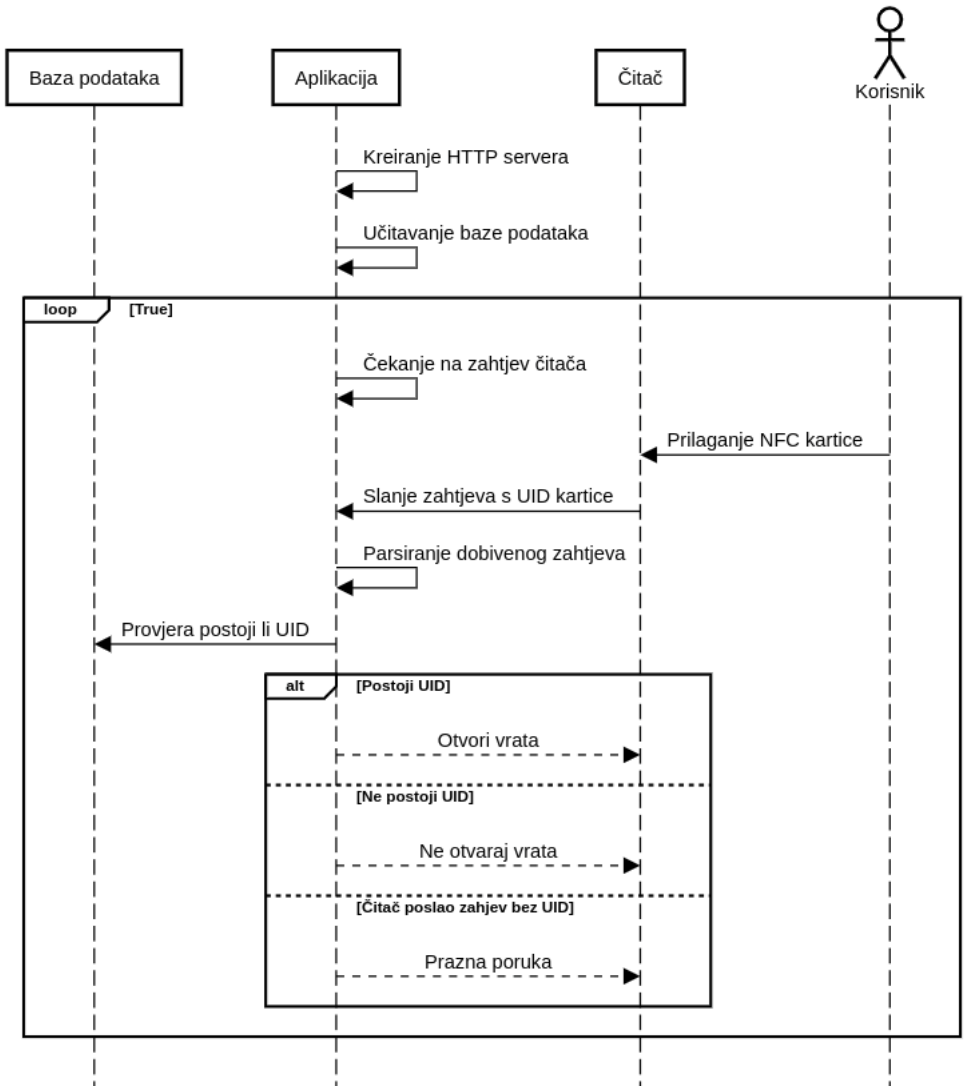
```
{"_default": {
  "1": {"uid": "AEF056"},
  "2": {"uid": "123456"},
  "3": {"uid": "555555"},
  "4": {"uid": "31053B71"},
  "5": {"uid": "123456"}
}}
```

Listing 5.1: Primjer izgleda datoteke db.json

5.3. Autentifikacija

Autentifikacija korisnika se provodi preko aplikacije **nfc_handler.py**.
Tijek izvršavanja programa je sljedeći:

- Kreiranje HTTP servera
- Učitavanje baze podataka
- Čekanje na zahtjev čitača
- Korisnik prilaže NFC karticu
- Čitač pošalje zahtjev
- Aplikacija parsira dobiven zahtjev
- Aplikacija provjerava u bazi podataka
  - Postoji UID, vrati čitaču da otvori vrata
  - Ne postoji UID, vrati čitaču da ne otvara vrata
  - Nema UID u poslanom zahtjevu, pošalji prazan odgovor
- Vrati se na čekanje zahtjeva



Slika 5.2: Sekvencijski dijagram sustava

5.3.1. Inicijalizacija aplikacije

Aplikacija se incijalizira uključivanjem iz biblioteka:

- **http.server** klase:
  - HTTPServer
  - BaseHTTPRequestHandler
- **tinydb** klase
  - TinyDB
  - Query
- **datetime**, klasu datetime
- **socket**

Postavlja **HOST_NAME** i **PORT_NUMBER**. Te varijable su postavljene na IP adresu '192.168.7.191' i vrata 80 jer čitač zadano šalje svoje zahtjeve na tu adresu i vrata.
Dolazi do inicijalizacije web poslužitelja i dohvaćanje baze podataka. Nakon toga aplikacija čeka na zahtjeve od čitača.

```
#!/usr/bin/python3
from http.server import HTTPServer, BaseHTTPRequestHandler
from tinydb import TinyDB, Query
from datetime import datetime
import socket

HOST_NAME = '192.168.7.191'
PORT_NUMBER = 80

try:
    server = HTTPServer((HOST_NAME, PORT_NUMBER), Server)
    print ("%s Started httpserver on port %d" % (str(datetime.now()), PORT_NUMBER))
    db = TinyDB('db.json')

    server.serve_forever()

except KeyboardInterrupt:
    print ('^C received, shutting down the web server')
    server.socket.close()

```

Slika 5.3: Inicijalizacija aplikacije

5.3.2. Parsiranje zahtjeva

Kada aplikacija dobije zahtjev od čitača, čitač sve informacije šalje putem URL-a. Zato je potrebno parsirati URL i pronači UID. Za to služi funkcija **handle_nfc**.
UID se pronalazi prvo traženjem njegove duljine u bajtima, a zatim se traži identifikacijski broj.
Funkcija vraća ili duljinu UID-a i UID ili vrijednosti False, False ovisno o uspješnosti pronalaska istih u zahtjevu.

```
def get_uid(url):
    if "ulen" in url:
        uid_len = int(url[url.find("ulen") + 5])
        uid_start = int(url.find("uid")) + 4
        uid = url[uid_start:(uid_start + uid_len * 2)]
        return (uid_len, uid)
    else:
        return (False, False)

```

Slika 5.4: Parsiranje zahtjeva

5.3.3. Provjera baze podataka

Nakon parsiranja zahtjeva dolazimo do provjere postoji li dobiveni UID u bazi podataka. Koristimo funkciju **handle_nfc**.
Provjera se izvršava preko instancirane baze **db** i njene funkcije **search**.
Ako postoji UID u bazi zabilježavamo vrijeme, otvaranje vrata i UID te vraćamo **'Open'**.
Ako ne postoji UID u bazi zabilježavamo vrijeme, neautoriziran pristup i UID te vraćamo **'Close'**.
Ako u zahtjevu nije postojao UID, tj. čitač je poslao svoj dijagnostički zahtjev, vraćamo **'PING'**.

```
35 def handle_nfc(url):
36     uid_len, uid = get_uid(url)
37     print ('Uid_len:', uid_len)
38     print ('Uid:', uid)
39     if uid_len is not False:
40         Uid = Query()
41         if db.search(Uid.uid == uid):
42             print ("%s Door opening. UID: %s" % (str(datetime.now()), uid))
43             return 'Open'
44         else:
45             print("%s Unauthorized access. UID: %s" % (str(datetime.now()), uid))
46             return 'Close'
47     else:
48         return 'PING'
49
```

Slika 5.5: Provjera baze podataka

5.3.4. Obrada zahtjeva

Klasa Server čeka HTTP zahtjev od strane čitača. Kada čitač pošalje zahtjev klasa šalje URL zahtjeva prethodno opisanoj funkciji **handle_nfc**.
Kada funkcija vrati povratnu informaciju ovisno o njoj šalje čitaču odgovor.
Nakon slanja odgovora aplikacija se vraća u stanje čekanja zahtjeva.

```
HTTP/1.0 200 OK
Server: BaseHTTP/0.6 Python/3.6.7
Date: Mon, 10 Jun 2019 14:11:13 GMT
Content-type: text/html

<ORBIT>
GRNT=05
UI=820432

</ORBIT>^C
```

Slika 5.6: Odgovor kod uspješne autentifikacije

```
HTTP/1.0 200 OK
Server: BaseHTTP/0.6 Python/3.6.7
Date: Mon, 10 Jun 2019 14:10:07 GMT
Content-type: text/html

<ORBIT>
DENY=05
UI=A00332

</ORBIT>^C
```

Slika 5.7: Odgovor kod neuspješne autentifikacije

```
HTTP/1.0 200 OK
Server: BaseHTTP/0.6 Python/3.6.7
Date: Mon, 10 Jun 2019 14:12:39 GMT
Content-type: text/html

<ORBIT>RLY=1
UI=000000

</ORBIT>
```

Slika 5.8: Odgovor kod dijagnostičkog zahtjeva

```
11 class Server(BaseHTTPRequestHandler):
12     def do_GET(self):
13         command = handle_nfc(self.path)
14 
15         if command is 'Open':
16             self.send_response(200)
17             self.send_header('Content-type','text/html')
18             self.end_headers()
19             self.wfile.write("<ORBIT>\nGRNT=05\nUI=820432\n\n</ORBIT>".encode("utf-8"))
20 
21         elif command is 'Close':
22             self.send_response(200)
23             self.send_header('Content-type','text/html')
24             self.end_headers()
25             self.wfile.write("<ORBIT>\nDENY=05\nUI=A00332\n\n</ORBIT>".encode("utf-8"))
26 
27         elif command is 'PING':
28             self.send_response(200)
29             self.send_header('Content-type','text/html')
30             self.end_headers()
31             self.wfile.write("<ORBIT>RLY=1\nUI=000000\n\n</ORBIT>".encode("utf-8"))
32 
33         return
34 
```

Slika 5.9: Obrada zahtjeva

5.4. Implementacija

Čitač i uređaj na kojoj se pokreće poslužiteljska aplikacija, Raspberry Pi Model 2 B, komuniciraju putem Ethernet priključka.
Čitač je također i napajan preko Ethernet priključka te je za njegov rad potreban PoE mrežni preklopnik ili PoE ubrizgivač. Kod implementacije rada koristio se Mikrotik Gigabit PoE ubrizgivač u koji se dovodi upredena parica i 48V istosmjerni pretvornik.

Slika 5.10: Mirkotik Gigabit PoE ubrizgivač

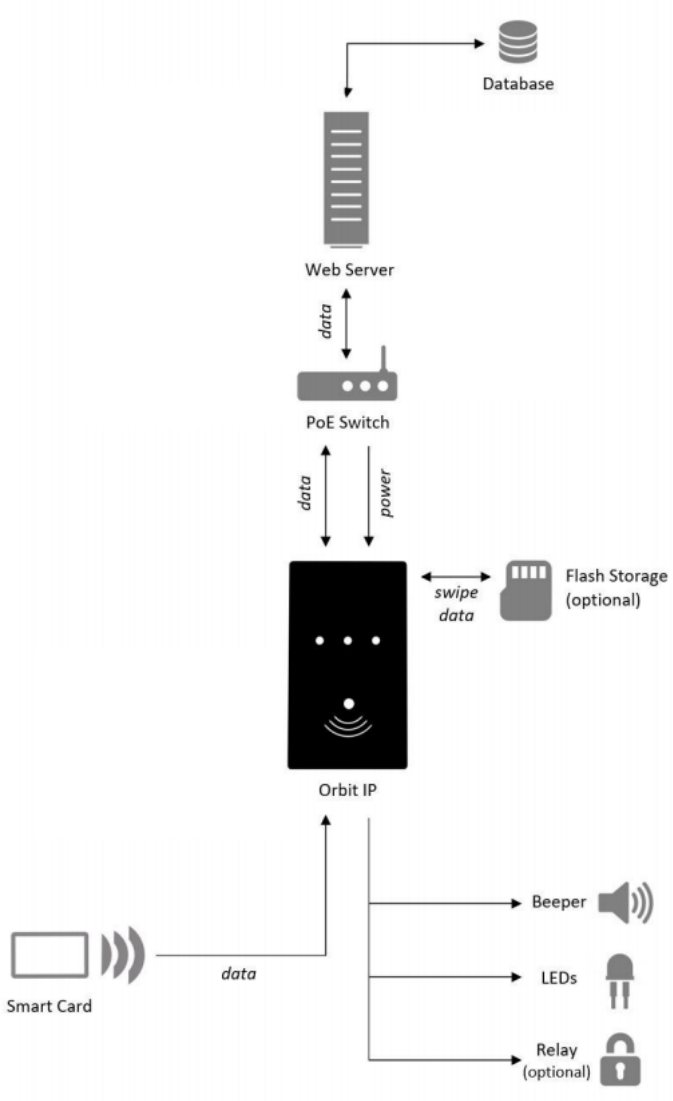
Čitač i Raspberry Pi povezani su međusobno bez upotrebe usmjeritelja. Zato je potrebno na Raspberry Pi-u uspostaviti mrežu. Mreža se uspostavlja preko skripte **startup.sh**.
Skripta također služi za instalaciju baze podataka **TinyDB** i pokreće poslužiteljsku aplikaciju **nfc_handler.py**.

```
1 #!/bin/bash
2
3 pip3 install tinydb
4
5 sudo ip ad add 192.168.7.191/24 dev enp2s0
6
7 sudo python3 nfc_handler.py
```

Slika 5.11: Bash skripta startup.sh

Zadnja dio sustava je elektronička brava koja otvara bravu kada dobije napon od 12V. Napon dobiva iz čitača koji putem releja kod odgovora od poslužiteljske aplikacije daje napon kroz žice.

Slika 5.12: Elektronička brava



Slika 5.13: Shema sustava

6. Zaključak

Zaključak.

LITERATURA

Kontrola ulaza korištenjem beskontaktnih kartica

Sažetak

Sažetak na hrvatskom jeziku.

Ključne riječi: Ključne riječi, odvojene zarezima.

Title

Abstract

Abstract.

Keywords: Keywords.