ŠOLSKI CENTER VELENJE
ELEKTRO IN RAČUNALNIŠKA ŠOLA
Trg mladosti 3, 3320 Velenje

MLADI RAZISKOVALCI ZA RAZVOJ SAŠA REGIJE

RAZISKOVALNA NALOGA

PAMETNO ODKLEPANJE RAČUNALNIŠKIH UČILNIC

Tematsko področje: Računalništvo

Avtorja:

Blaž Osredkar, 3. letnik

Urban Krepel, 3. letnik

Mentorja:

Uroš Remenih, inž.

Samo Železnik, inž.

Velenje, 2023

Raziskovalna naloga je bila opravljena na ŠC Velenje, Elektro in računalniška šola, 2023.

Mentorja: Uroš Remenih, inž. inf., Samo Železnik, inž. inf.

Datum predstavitve: Marec 2023

KLJUČNA DOKUMENTACIJSKA INFORMACIJA

ŠD Elektro in računalniška šola Velenje, šolsko leto 2022/2023
KG mobilna aplikacija / programiranje / čitalci
AV OSREDKAR, Blaž / KREPEL, Urban
SA REMENIH, Uroš / ŽELEZNIK, Samo
KZ 3320 Velenje, Trg mladosti 3
ZA ŠC Velenje, Elektro in računalniška šola, 2023
LI 2023
IN **PAMETNO ODKLEPANJE RAČUNALNIŠKIH UČILNIC**
TD Raziskovalna naloga
OP
IJ SL
JI sl / en
AI

Dandanes je uporaba ključev še vedno najpogostejši način odklepanja vrat, vendar pa to prinaša nekaj nevšečnosti in varnostno tveganje. Namen raziskovalne naloge je na ŠC Velenje razviti in vzpostaviti pametni sistem za odklepanje učilnic, ki ima namen olajšati dostop do učilnic in pri tem zagotoviti večjo varnost. Sistem bo podpiral odklepanje učilnic s kartico ali mobilno aplikacijo. Vstop bo dovoljen samo dijakom, ki imajo določeni trenutek pouk v določeni učilnici. Uporaba sistema prinaša številne prednosti, kot so večja varnost, manjša možnost izgube ključev ter enostavnejši in hitrejši dostop do učilnic.

KEY WORDS DOCUMENTATION

ND Elektro in računalniška šola Velenje, šolsko leto 2022/2023

CX mobilna aplikacija / programiranje / čitalci

AU OSREDKAR, Blaž / KREPEL, Urban

AA REMENIH, Uroš / ŽELEZNIK, Samo

PP 3320 Velenje, Trg mladosti 3

PB ŠC Velenje, Elektro in računalniška šola, 2023

PY 2023

TI **SMART UNLOCKING OF COMPUTER LABS**

DT Raziskovalna naloga

NO

LA SL

AL sl / en

AB

Nowadays, the use of keys is still the most common method to unlock doors, but it comes with some inconveniences and security risks. The aim of this research is to develop and implement a smart classroom unlocking system at School Center Velenje that will facilitate and better secure access to classrooms. The system will support the unlocking of classrooms with a card or mobile application. Students will only be able to enter if they are currently being taught in a particular classroom. The system offers a number of benefits, including increased security, reduced risk of key loss and easier and faster access to classrooms.

KAZALO VSEBIN

KAZALO GRAFOV

KAZALO SLIK

SEZNAM OKRAJŠAV IN SIMBOLOV

angl. – angleško

SQL – strukturirani povpraševalni jezik (*angl*. structured query language)

GPIO – general purpose input/output

API – application programming interface

URL – enolični krajevnik vira (*angl*. uniform resource locator)

NFC – near-field communication (*slov.* komunikacija s sosednjim poljem)

DB – podatkovna baza (*angl*. database)

QR – quick response (*slov.* hiter odziv)

Zaledni del – del aplikacije, do katerega uporabnik ne dostopa neposredno, običajno odgovoren za shranjevanje in obdelavo podatkov.

USB – univerzalno serijsko vodilo (*angl*. universal serial bus)

HDMI – high-definition multimedia interface

ID – identifier

RFID – radiofrekvenčna identifikacija (*angl*. radio frequency identification)

GDPR – uredba o varstvu podatkov (*angl*. general data protection regulation)

1 UVOD

V sodobnem svetu ima tehnologija vse pomembnejšo vlogo in se vse bolj uporablja tudi v izobraževanju. V sodelovanju s ŠC Velenje smo se odločili raziskati možnost uporabe pametnega odklepanja učilnic, ki bi dijakom in profesorjem omogočila preprostejši in varnejši dostop do učilnic. Sistem pametnega odklepanja deluje na podlagi uporabe kartice ali aplikacije, ki omogoča odklepanje vrat samo dijaku, ki ima v tistem trenutku pouk v določeni učilnici. Tak način odklepanja bo prispeval k večji varnosti v šoli, saj bo zagotovil nadzor nad tem, kdo ima dostop do posameznih prostorov. Poleg tega bo sistem olajšal evidenco prisotnosti dijakov v šoli ter zmanjšal možnost izgube ključev. Namen te raziskovalne naloge je preučiti prednosti uporabe pametnega odklepanja učilnic ter opisati njegovo delovanje in implementacijo.

1.1 HIPOTEZE

1. hipoteza: Z uporabo aplikacije ŠCVApp in kartice kot ključa bo olajšan dostop do učilnic in drugih prostorov v šoli.

2. hipoteza: Bralnike kartic in krmilnik bo možno povezati z mobilno aplikacijo ŠCVApp, kar bo omogočalo odklepanje preko aplikacije.

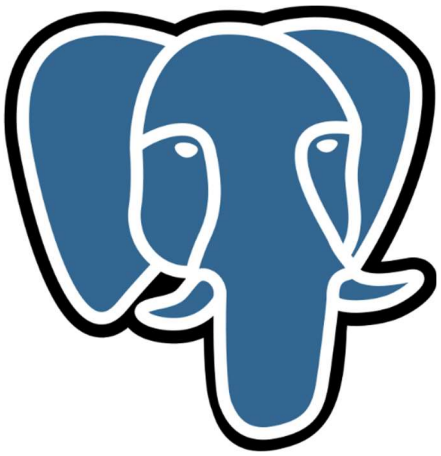
3. hipoteza: Sistem bo omogočal dostop dijakom ob času, ko imajo na urniku pouk v učilnici.

2 PREGLED OBJAV

2.1 PROGRAMSKA ORODJA

2.1.1 POSTGRESQL

PostgreSQL je zmogljiv odprtokodni sistem objektno-relacijske baze podatkov. Uporablja SQL-jezik in ga dopolnjuje skupaj z številnimi funkcijami, ki varno shranjujejo in prilagajajo najzapletenejše delovne obremenitve podatkov. PostgreSQL je opremljen s številnimi funkcijami, ki so namenjene razvijalcem v pomoč pri izdelavi aplikacij in skrbnikom pri zaščiti celovitosti podatkov in izdelavi okolij, odpornih na napake, in ki pomagajo upravljati podatke ne glede na to, kako velik ali majhen je nabor podatkov. (1)



Slika 1: PostgreSQL logo

2.1.2 UBUNTU SERVER

Ubuntu Server je odprtokodni operacijski sistem, namenjen gostovanju spletnih strani, skupni rabi datotek in različnih procesov. (2)

Slika 2: Ubuntu logo

2.1.3 FLUTTER

Flutter je odprtokodni komplet za razvoj programske opreme uporabniškega vmesnika, ki ga je ustvaril Google. Flutter se uporablja za razvoj aplikacij za več platform za Android, iOS, Linux, macOS, Windows, Google Fuchsia in splet iz ene kodne baze. Flutter je bil prvič opisan leta 2015, izšel pa je maja 2017.

Aplikacije Flutter so napisane v skriptnem programskem jeziku Dart in uporabljajo številne naprednejše funkcije tega jezika. (6)

Slika 3: Prikaz logotipa programskega orodja Flutter (7)

2.1.4 NESTJS

Nest (NestJS) je okvir ogrodja za gradnjo učinkovitih, razširljivih aplikacij na strani strežnika Node.js. Uporablja skriptni programski jezik JavaScript, ki je zgrajen iz programskega jezika TypeScript in ga v celoti podpira. Pod pokrovom Nest uporablja robustne okvire strežnika HTTP, kot je Express.

Slika 4: Prikaz logotipa ogrodja NestJS (8)

2.1.5 PYTHON

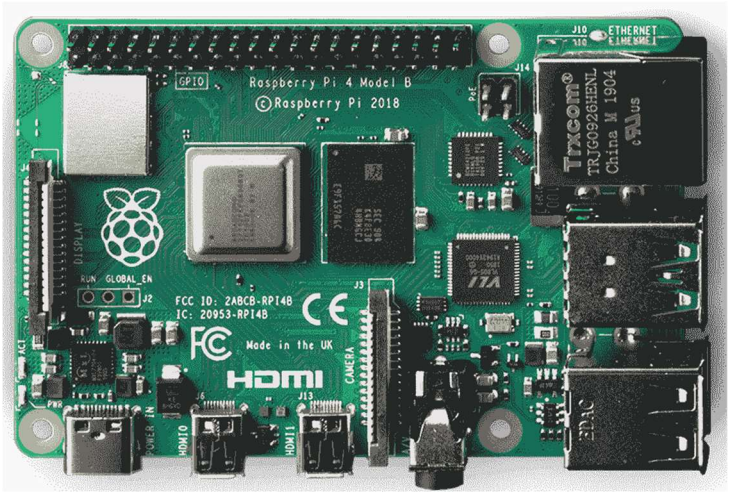
Python je programski jezik, ki je interpretiran in objektno usmerjen. S svojimi vgrajenimi podatkovnimi strukturami na visoki ravni, dinamičnim tipiziranjem in dinamičnim vezanjem predstavlja učinkovit jezik za hitri razvoj aplikacij ter za uporabo kot skriptni ali povezovalni jezik pri integraciji obstoječih komponent. (4)

Slika 5: Prikaz logotipa ogrodja Python

2.2 STROJNA OPREMA

2.2.1 RASPBERRY PI 4

Raspberry Pi 4 je majhen zmogljiv računalnik, ki ga je razvil Raspberry Pi Foundation. Napravo poganja 4-jedrni ARM-procesor. Na napravi so različni vhodi/izhodi, kot so USB, HDMI, internetni priključek in priključek za SD-kartico. Naprava vključuje tudi GPIO, ki je zadolžen za povezovanje različnih dodatnih senzorjev/komponent. (11)



Slika 6: Prikaz majhnega računalnika Raspberry Pi 4 (10)

2.2.2 RELE

Rele je elektromagnetno stikalo, ki ga krmili tok skozi magnetno navitje, pri čemer sta krmilni in močnostni tokokrog galvansko ločena. Njegova osnovna funkcija je krmiljenje večjega porabnika z uporabo majhne napetosti/toka.

Rele uporabljamo za krmiljenje večjih porabnikov, ki jih zaradi porabe večjega toka ne bi mogli krmiliti neposredno iz krmilnika. Rele uporavljamo tudi za ločevanje signalov in krmiljenje različnih tokokrogov. (15)

2.3 KOMUNIKACIJSKI PROTOKOLI

2.3.1 RADIOFREKVENČNA IDENTIFIKACIJA (RFID)

RFID je tehnologija, pri kateri izvajamo identificiranje s pomočjo elektromagnetnega valovanja na področju radijskih frekvenc. Ta tehnologija obstaja že več kot 60 let. Njeni glavni prednosti sta, da lahko RFID-odzivnike, ki hranijo določeno informacijo, prepoznamo brezkontaktno (tudi v primerih, ko niso vidni direktno), in da lahko vanje dodatno vpisujemo nove informacije. (12)

2.3.2 NFC

NFC pomeni komunikacijo bližnjega polja in omogoča, da telefoni, tablični računalniki, prenosniki in druge naprave zlahka delijo podatke z drugimi napravami, opremljenimi z NFC. NFC se je razvil iz tehnologije radiofrekvenčne identifikacije (RFID). RFID stoji za tistimi varnostnimi karticami za skeniranje, ki vas vsak dan »pripeljejo« v pisarno, pa tudi za plačilnimi brezstičnimi karticami. NFC je zelo podoben RFID-u, vendar je NFC omejen na komunikacijo znotraj približno 4 palcev. Eden izmed razlogov, zakaj se je NFC uveljavil kot varna alternativa kreditnim karticam, je njegova velika varnostna prednost. NFC lahko prenaša podatke, kot so videoposnetki, kontaktni podatki in fotografije, med dvema napravama, ki podpirata NFC. (13)

2.3.3 NFC ZNAČKA

NFC omogoča brezžično komunikacijo med dvema napravama. Tehnologijo je mogoče vgraditi v majhno »označko«, ki omogoča prenos podatkov med bližnjimi mobilnimi telefoni, prenosnimi in tabličnimi računalniki ter drugimi elektronskimi napravami.

Značka NFC deluje kot vse druge oznake RFID in komunicira prek radijskih valov. Dve napravi – oznaka NFC in bralnik NFC – si izmenjujeta informacije v formatu za

izmenjavo podatkov NFC. Značka NFC pošilja radijske valove, ki aktivirajo anteno v sprejemni napravi. Prejemnik potrdi informacije in tako zaključi izmenjavo informacij. Tehnologija deluje na zelo kratki razdalji, in sicer približno 4 palce. Oznake NFC delujejo brez baterije in črpajo energijo iz druge naprave, npr. pametnega telefona. (14)

3 MATERIAL IN METODE DELA

3.1 STROJNA OPREMA

3.1.1 IZBIRA KLJUČAVNICE

Na tržišču je danes na voljo veliko različnih vrst ključavnic, ki se med seboj razlikujejo glede na princip delovanja. Med najbolj znanimi vrstami so standardne ključavnice, ki delujejo na zatiče in se odklepajo s standardnim ključem, ključavnice z električnim motorjem ter ključavnice, ki temeljijo na elektromagnetnem principu. Obstajajo tudi »vse v enem« ključavnice, ki so sestavljene iz krmilnika z releji in ključavnice z električnim motorjem ali elektromagnetom.

Pri izbiri ključavnice smo se odločili za elektromagnetno ključavnico, ki deluje na principu ustvarjanja magnetnega polja s pomočjo tuljave. Magnetno polje dvigne kovinsko palčko v ključavnici in s tem sprosti ključavnico, da se lahko odprejo vrata.

Prednost te vrste ključavnice je, da jo je zelo težko mehansko odkleniti, npr. z uporabo magneta, in da je zelo kompaktna ter primerna za vgradnjo v ožje prostore.

3.1.2 IZBIRA KRMILNIKA

Na tržišču je danes na voljo mnogo vrst krmilnikov. Iščemo takšen krmilnik, ki ga bomo lahko upravljali preko lokalnega omrežja in ki bo podpiral 4 bralnike in 4 ključavnice. Odločili smo se za krmilnik, ki izpolnjuje vse naše zahteve, in sicer omogoča upravljanje preko aplikacije preko lokalnega omrežja, podpira 4 bralnike in 4 ključavnice ter nudi visoko raven varnosti.

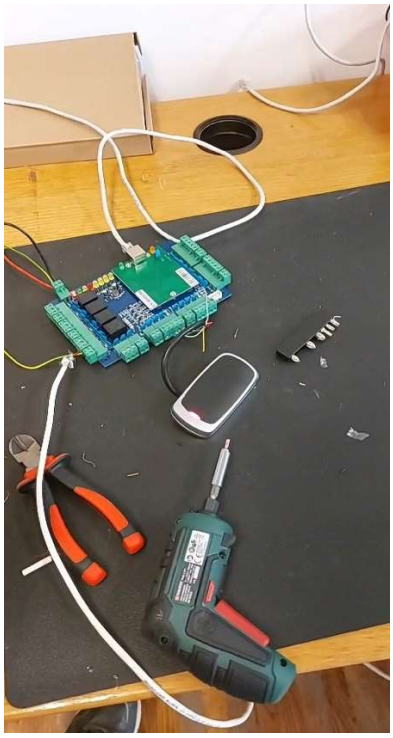
3.1.3 IZBIRA BRALNIKA

Dandanes na tržišču najdemo številne vrste bralnikov, opremljenih s funkcijami, kot so branje NFC-, QR-kod in bar-kod.

Pri izbiri ustrezne vrste bralnika je treba upoštevati, da mora biti ta združljiv z že obstoječimi karticami in s krmilnikom. Naše kartice delujejo na tehnologiji RFID, zato je ključnega pomena izbrati bralnik, ki podpira to tehnologijo.

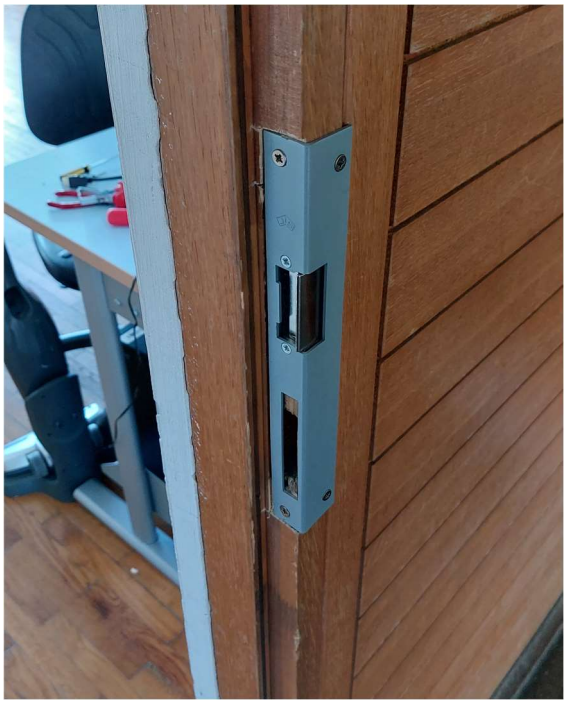
Izbrali smo bralnik, ki je združljiv z našim krmilnikom in podpira tehnologijo RFID. Poleg tega je opremljen z učinkovito svetlobno indikacijo, ki uporabniku sporoča, ali ima dovoljenje za dostop.

3.1.4 PRIPRAVA NA MONTAŽO BRALNIKOV



Slika 7: Prototip priklopa čitalnika na krmilnik (Vir: lasten)

Nato smo pripravili načrt, katera vrata bodo prva za montažo in kje se bo nahajal glavni krmilnik. Odločili smo se, da bomo začeli v učilnici C 503, kjer se bo nahajal tudi krmilnik. Začeli smo iskati že obstoječe žice v strehi (dovod elektrike in interneta). Ker je bila na strehi samo ena povezava do interneta, smo ta kabel razdelili na dve povezavi. Začeli smo z namestitvijo ključavnice.

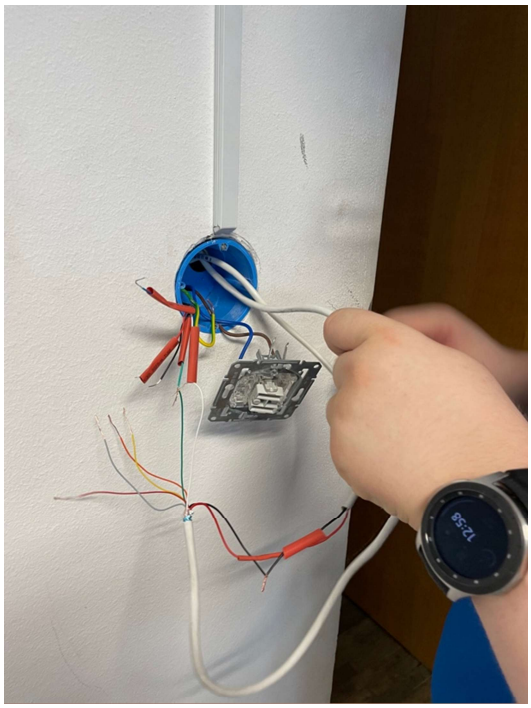


Slika 8: Električna ključavnica (Vir: lasten)

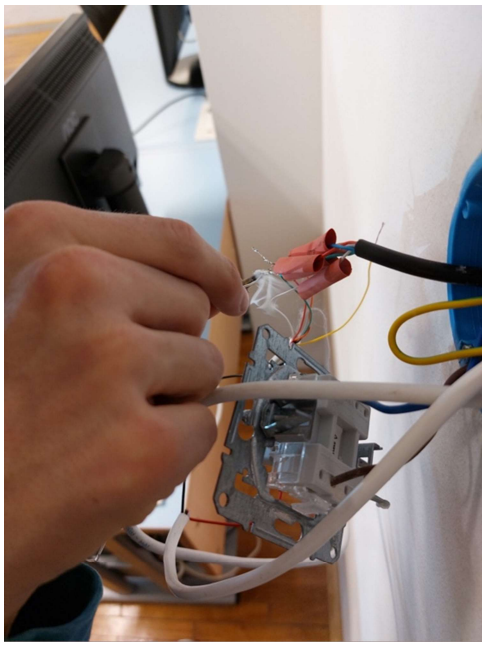
Sledila je namestitev bralnika. Prvi korak je bila priprava okvirja, na katerem leži bralnik kartic. Pri tem smo se osredotočili na natančno postavitev okvirja, da bi za namestitev bralnika kartic zagotovili stabilno in varno podlago.

Slika 9: Okvir bralnika kartic, nameščen na steno. (Vir: lasten)

Po uspešni pripravi okvirja smo nadaljevali z naslednjim korakom – priprava in spajanje kablov. Najprej smo na kablih odstranili približno 1–1,5 cm plastičnega ovoja ter si na vsak kabel pripravili toplotno krčljivo ovojnico, ki smo jo ob končanem spajanju tudi ovili okoli že omenjenega spoja.



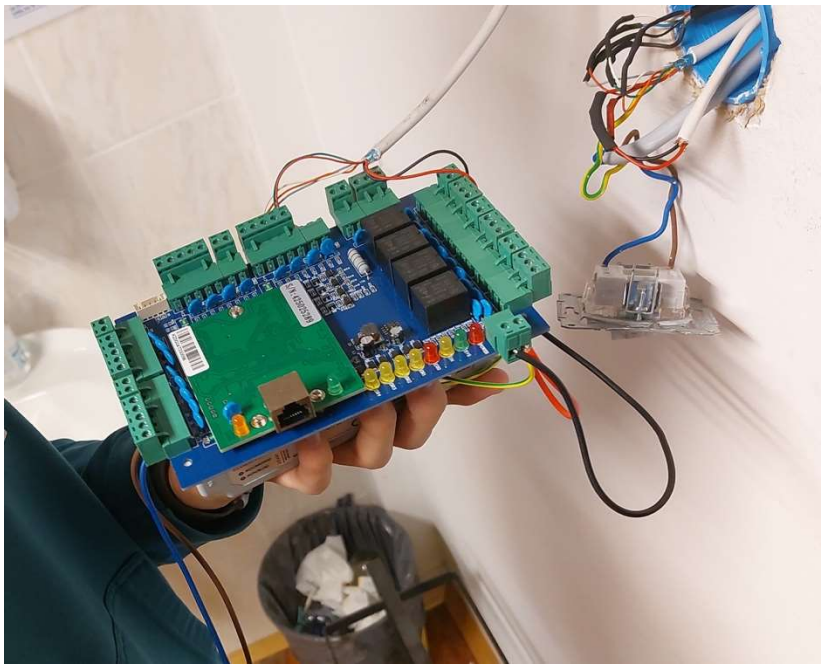
Slika 10: Priprava na spajanje kablov (Vir: lasten)



Slika 11: Spajanje kablov (Vir: lasten)

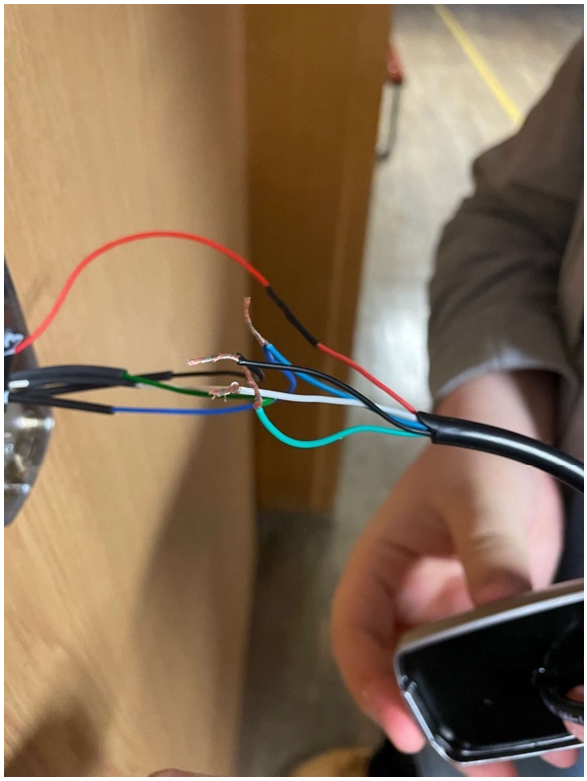
Slika 12: Izgled kablov ob končanem delu s spajanjem in krčenjem toplotne krčljive ovojnice (Vir: lasten)

Ena učilnica je bila pripravljena na preizkus. Najprej smo krmilnik poskusno priklopili, da smo preverili, ali bralnik deluje tako, kot je treba.



Slika 13: Preizkus bralnika kartic (Vir: lasten)

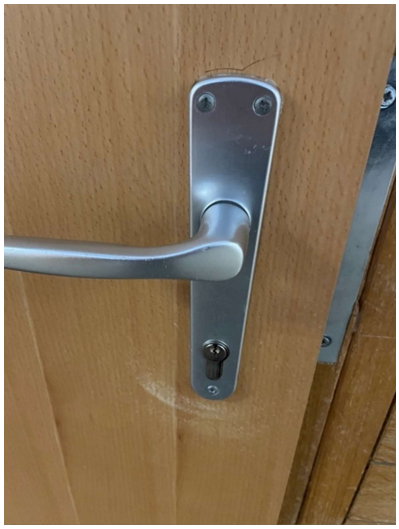
Po preizkusu je sledilo nameščanje bralnikov še na ostala vrata (učilnice C 502, notranji vhod v kabinet C 504, zunanji vhod v kabinet C 504).



Slika 14: Nameščanje bralnika kartic na zunanja vrata kabineta (Vir: lasten)

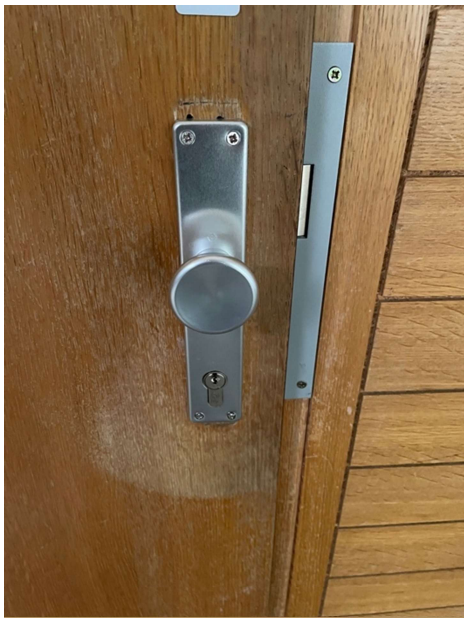
3.1.5 ZAMENJAVA KLJUKE

Kljuka, ki smo jo imeli na vratih, je bila neprimerna za nov sistem, ker ima na obeh straneh ročaj, s katerim je mogoče odpreti vrata (glej Slika 15).



Slika 15: Stara kljuka

Zato smo jo morali zamenjati z novo kljuko, ki ima na notranji strani ročaj, na zunanji strani pa kljukico, ki onemogoča odpiranje vrat z zunanje strani (Slika 16).

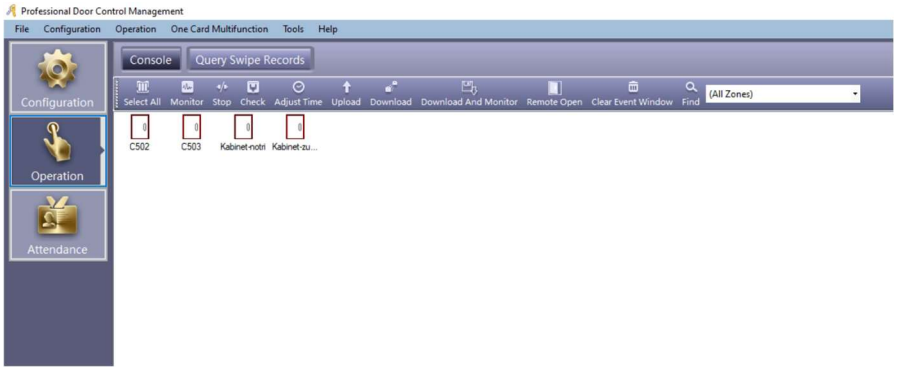


Slika 16: Nova kljuka

Ker smo želeli ohraniti možnost odklepanja s ključem, smo zamenjali tudi mehanizem ključavnice, tako da je zdaj moč ključ obrniti v levo in odkleniti vrata.

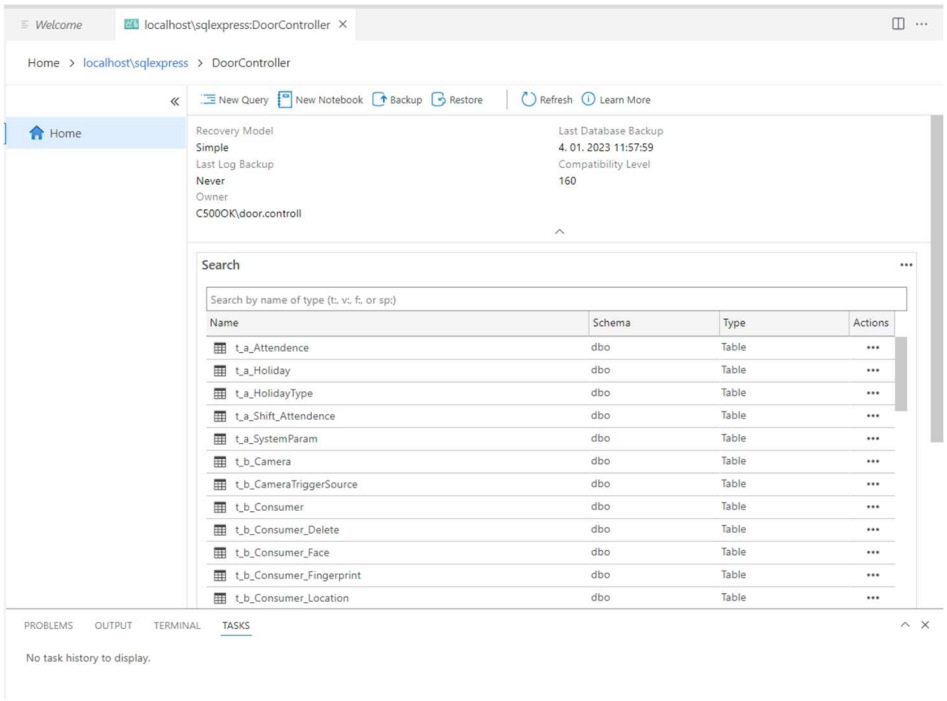
3.1.6 PRIPRAVA STREŽNIKA ZA GOSTOVANJE PROGRAMSKE OPREME ZA BRALNIKE KARTIC

Sledila je priprava programskega dela za bralnike kartic. Za bralnike smo pridobili specializiran program poimenovan »Professional Door Control Management«. Aplikacijo smo nato naložili na strežnik, ki ga je podarila Mestna občina Velenje.



Slika 17: Aplikacija Professional Door Control Management (Vir: lasten)

V aplikaciji smo najprej prevzeli krmilnik, ki smo mu določili imena vrat glede na učilnice (prikazano na Slika 17). Aplikaciji smo dodali seznam dijakov, njihov razred ter številko dijaške kartice. Enako smo naredili tudi za učitelje. Začasno smo si ustvarili časovne profile, kamor smo ročno vpisali ure za razrede, ki imajo pouk v teh učilnicah. Ko smo bolje spoznali program, smo ugotovili, da omogoča spremembo baze podatkov, zato smo na strežnik naložili SQL, potem pa sistem za upravljanje podatkovnih baz Azure Data Studio. Nato smo programu spremenili nastavitve za podatkovno bazo, kjer smo zamenjali podatkovno bazo z našo.



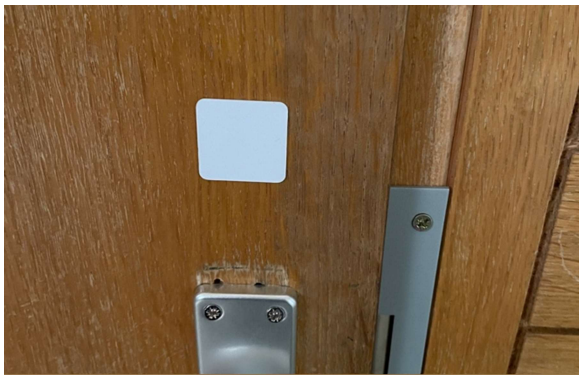
Slika 18: Prikaz baze podatkov v Azure Data Studio (Vir: lasten)

3.1.7 NADGRADNJA SISTEMA ZA POVEZAVO Z MOBILNO APLIKACIJO

Postavili smo kartični sistem in želeli omogočiti odklepanje vrat z mobilno aplikacijo. Da bomo to dosegli, moramo namestiti Raspberry Pi ter nanj priključiti releje, ki bodo prek GPIO-priključka na Raspberry Pi krmilili odklepanje vrat. Raspberry Pi bo deloval kot krmilnik, ki bo ukaze prejemal iz zalednega dela. Releji bodo omogočili odklepanje vrat, ko bodo združili dva kontaktorja na krmilniku za vrata.

Za odklepanje vrat z mobilnim telefonom je treba označiti vrata z ustrezno oznako. Izbrali smo NFC-značko in QR-kodo, na kateri smo shranili povezavo, ki omogoča odpiranje

naše aplikacije. NFC-značko smo namestili na vrata (glej Slika 19), medtem ko bomo QR-kodo natisnili kot nalepko in jo prilepili na značko.



Slika 19: NFC značka na vratih

3.2 PROGRAMSKI DEL

Ker je bila naša želja, da se vrata odpirajo tudi preko mobilne aplikacije ŠCVApp, je bilo treba povečati mobilno aplikacijo in zaledni del.

3.2.1 IZBIRA PODATKOVNE BAZE

Za podatkovno bazo smo izbrali PostgreSQL, ki je zmogljiv odprtokodni sistem objektno-relacijske baze podatkov. Uporablja SQL-jezik in ga dopolnjuje skupaj z mnogimi funkcijami, ki varno shranjujejo in prilagajajo najzapletenejše delovne obremenitve podatkov. PostgreSQL je opremljen s številnimi funkcijami, ki so namenjene razvijalcem v pomoč pri izdelavi aplikacij, skrbnikom pri zaščiti celovitosti podatkov in izdelavi okolij, odpornih na napake, ter ki pomagajo upravljati podatke, ne glede na to, kako velik ali majhen je nabor podatkov. (1)

3.2.2 OPERACISKI SISTEM ZA RASPBERRY PI

Ubuntu Server je odprtokodni operacijski sistem, ki je namenjen za gostovanje spletnih strani, skupni rabi datotek in različnih procesov. (2)

3.2.3 OGRODJE ZA MOBILNO APLIKACIJO

Flutter je odprtokoden komplet za razvoj programske opreme uporabniškega vmesnika, ki ga je ustvaril Google. Uporablja se za razvoj aplikacij za več platform za Android, iOS, Linux, macOS, Windows, Google Fuchsia in splet iz ene kodne baze. Flutter je bil prvič opisan leta 2015, izšel je maja 2017.

Aplikacije Flutter so napisane v skriptnem programskem jeziku Dart in uporabljajo njegove številne naprednejše funkcije. (6)

3.2.4 OGRODJE ZA ZALEDNI DEL

Nest (NestJS) je okvir ogrodja za gradnjo učinkovitih, razširljivih aplikacij na strani strežnika Node.js. Uporablja skriptni programski jezik JavaScript, ki je zgrajen iz programskega jezika TypeScript in ga v celoti podpira. Pod pokrovom Nest uporablja robustne okvire strežnika HTTP, kot je Express.

3.2.5 PYTHON

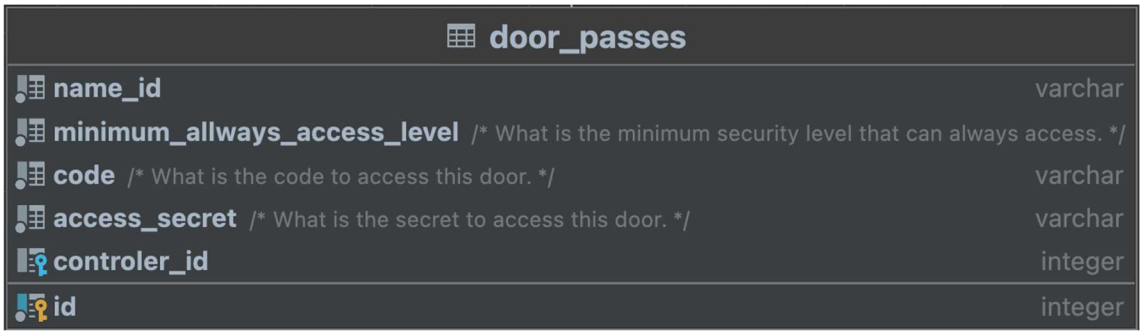
Python je programski jezik, ki je interpretiran in objektno usmerjen. S svojimi vgrajenimi podatkovnimi strukturami na visoki ravni, dinamičnim tipiziranjem in dinamičnim vezanjem predstavlja učinkovit jezik za hitri razvoj aplikacij in za uporabo kot skriptni ali povezovalni jezik pri integraciji obstoječih komponent. (4)

3.2.6 POVEČAVA ZALEDNEGA DELA

Zalednemu delu je bilo treba dodati podatkovno bazo in različne dodatne entitete.

Najprej smo začeli z izdelavo entitet.

Prva entiteta je bila izdelana za vrata. Entiteto smo poimenovali »door_passes«.



Slika 20: Entiteta za vrata (Vir: lasten)

Entiteta vsebuje 5 atributov, in sicer:

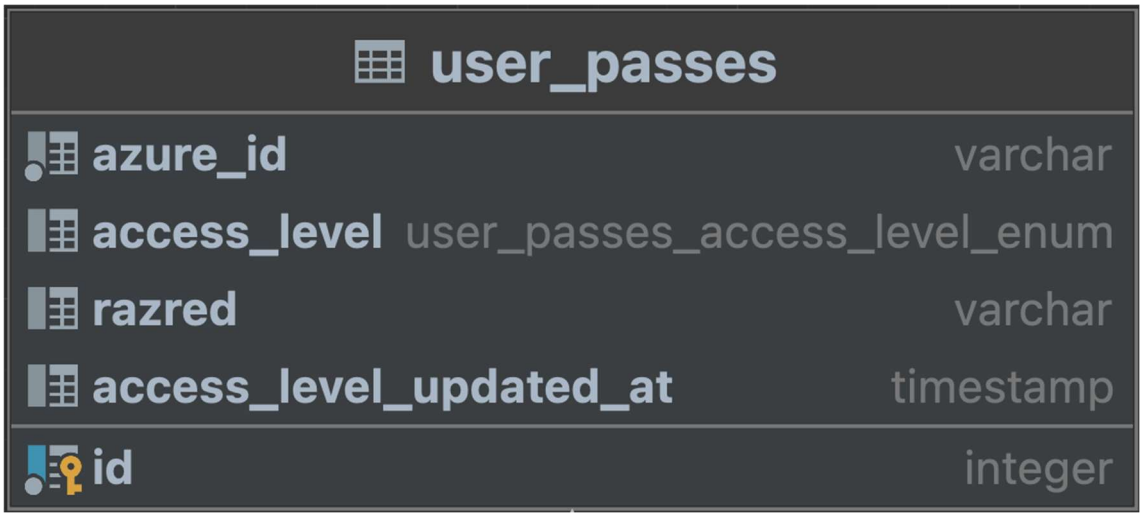
- »id« je tip inkrementalnega celoštevilskega identifikatorja;
- »name_id« je tipa znakovnega niza variabilne dolžine (*angl*. varchar) in je v njej vpisno ime učilnice;
- »minimum_allways_access_level« je prilagojen tip, namenjen za določanje varnostnih stopenj dostopa v to učilnico;

- »code« je tip znakovnega niza variabilne dolžine (*angl*. varchar), namenjen za indentifikacijo vrat;
- »access_secret« je tip znakovnega niza variabilne dolžine (*angl*. varchar), namenjen za avtentikacijo in avtorizacijo do določenih podatkov.

Entiteta vsebuje tudi 1 tuji ključ:

- »controler_id«, ki povezuje to entiteto z entiteto »pass_controlers«.

Naslednja entiteta je »uporabnik« in je namenjena za indentifikacijo uporabnika.

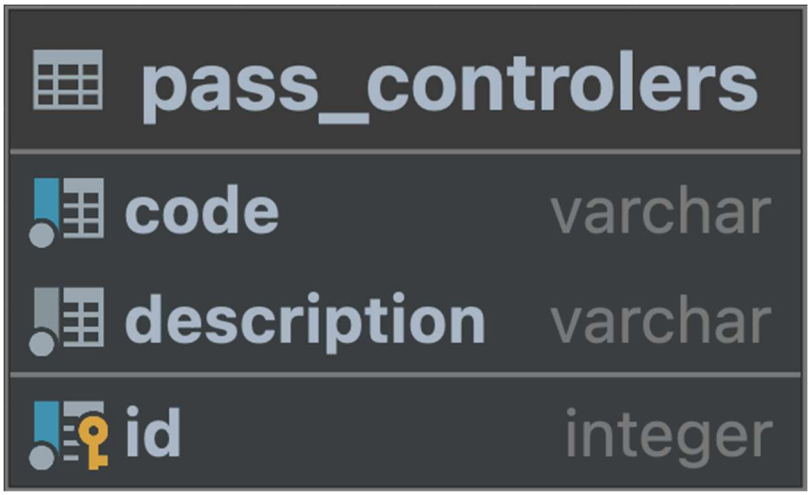


Slika 21: Entiteta za uporabnika (Vir: lasten)

Entiteta vsebuje 5 atributov, in sicer:

- »id«, ki je tip inkrementalnega celoštevilskega identifikatorja;
- »azure_id« je tip znakovnega niza variabilne dolžine (*angl*. varchar), ki ga uporabljamo za primerjavo uporabnika iz microsofta in ga s tem tudi identificiramo;
- »access_level« je prilagojen tip, s katerim povemo, katero varnostno stopnjo ima uporabnik;
- »access_level_update_at« je tip časovne oznake (*angl*. timestamp) in nam pove, kdaj je bil uporabnikov »access_level« nazadnje osvežen.

Naslednja entiteta je »krmilnik«, ki bo uporabljena, če bomo imeli več vrat, ki jih bo nadziral en krmilnik.



Slika 22: Entiteta za vrata (Vir: lasten)

Entiteta vsebuje 3 atribute, in sicer:

- »id«, ki je tip inkrementalnega celoštevilskega identifikatorja;
- »code«, ki je tip znakovnega niza variabilne dolžine (*angl*. varchar), namenjen za indentifikacijo tega krmilnika;
- »description«, ki je tip znakovnega niza variabilne dolžine (*angl*. varchar), namenjen za opis tega krmilnika (npr. »Krmilnik za ucilnici C502 in C503«).

Naslednja entiteta je »aktivnost«, ki bo uporabljena za beleženje uporabnikove aktivnosti.

Slika 23: Entiteta za beleženje aktivnosti (Vir: lasten)

Entiteta ima 3 atribute, in sicer:

- »id«, ki je tip inkrementalnega celoštevilskega identifikatorja;
- »status«, ki je prilagojen tip, namenjen, da vemo, če je bil vstop uspešen ali ne;
- »created_at«, ki je časovne oznake (*angl*. timestamp) in je namenjen, da vemo, kdaj je uporabnik vstopil.

Vsebuje še 2 tuja ključa, in sicer:

- »door_pass_id«, ki povezuje to entiteto z entiteto »door_passes«;
- »user_pass_id«, ki povezuje to entiteto z entiteto »user_passes«.

Entitete so ustvarjene, zdaj pa je treba ustvariti API, ki bo uporabljen v mobilni aplikaciji in krmilniku za vrata.

V API je treba, da vsebuje »endpoint« za pridobitev informacij o vratih, za njihov vrat, urejanje, izbris in ustvarjanje (*op. a.* za zadnje tri omenjene ima dostop samo administrator).

Da bomo ustvarili API, bomo uporabili takšnega, ki ga že imamo za ŠCVApp – samo nadgradili ga bomo.

Začeli smo tako, da smo v NestJS-u ustvarili nov modul, ki smo ga poimenovali »pass«. Modulu smo dodali še krmilnik in servis.

V krmilnik smo zapisali, na katere poti naj aplikacije »kličejo« API, kot je prikazano na Slika 24 (*op. a.* V tem primeru bo API dostopen na ».../pass/get_door/koda_vrat«; zadnji del poti je dinamičen, kar pomeni, da lahko uporabnik na koncu poti linka vnese poljubno število znakov, kar uporabimo, da preverimo, ali velja.).

```
@Get('get_door/:code')
@HttpCode(200)
async getDoorPasses(@Param('code', ValidationPipe) code: string) {
```

Slika 24: Prikaz funkcije v kontrolerju v NestJS (Vir: lasten)

V servise smo zapisali funkcije, ki jih uporabljamo v krmilniku. Funkcije v servisu po navadi komunicirajo z bazo podatkov. (Primer: Slika 25 prikazuje, kako s kodo vrat dobimo specifično entiteto vrat.)

```
async getDoorWithCode(code: string) {
  if (!code) {
    return null;
  }
  return await this.doorPassRepository.findOne({
    where: { code: code },
  });
}
```

Slika 25: Prikaz funkcije v servisih v NestJS (Vir lasten)

3.2.7 NADGRADNJA MOBILNE APLIKACIJE

Nadgradnja zalednega dela je končana, zdaj je treba nadgraditi mobilno aplikacijo tako, da bo delovala z novim, prenovljenim sistemom. Za razvijanje mobilne aplikacije smo uporabili ogrodje Flutter.

Naša prva naloga je bila, da lahko mobilno aplikacijo odpreva z nekakšno globalno povezavo oz. kot jo imenujejo razvijalci – »universal link« ali »deep link«. Da to dosežemo, je treba vstaviti najprej v posebni datoteki (»AndroidManifest.xml« za operacijski sistem Android in »Info.plist« za operacijski sistem iOS) nek poseben zapis, ki sistemu pove, s katero povezavo lahko odpre našo aplikacijo. Tukaj je prikazano, kaj je bilo treba vstaviti v ti dve datoteki: Slika 26 in Slika 27.

```
<intent-filter>
    <action android:name="android.intent.action.VIEW" />
    <category android:name="android.intent.category.DEFAULT" />
    <category android:name="android.intent.category.BROWSABLE" />
    <!-- Accepts URIs that begin with YOUR_SCHEME://YOUR_HOST -->
    <data
        android:scheme="scvapp"
        android:host="app.scv.si"
        android:pathPrefix="/open_door" />
</intent-filter>
```

Slika 26: Zapis za »AndroidManifest.xml«, da lahko aplikacijo odpremo z URL-jem (Vir: lasten)

```
<array>
    <dict>
        <key>CFBundleTypeRole</key>
        <string>Editor</string>
        <key>CFBundleURLName</key>
        <string>app.scv.si</string>
        <key>CFBundleURLSchemes</key>
        <array>
            <string>scvapp</string>
        </array>
    </dict>
</array>
```

Slika 27: Zapis za »Info.plist«, da lahko aplikacijo odpremo z URL-jem (Vir: lasten)

Sistem lahko odpre aplikacijo s posebnim URL-naslovom, zdaj pa je še treba narediti, da bo aplikacija izvedela, katera povezava je odprla aplikacijo. Da to dosežemo, bomo uporabili Flutter-knjižnico »uni_links«, ki deluje tako, da ko je bila aplikacija prvič zagnana, pokličemo funkcijo, ki nam vrne povezavo, s katero je bila aplikacija zagnana (Slika 28).

```
final initialURI = await getInitialUri(); // Dobimo povezavo s katero je bila aplikacija zagnana
```

Slika 28: Koda, kako dobimo povezavo, s katero je bila aplikacija zagnana (Vir: lasten)

Če je bila aplikacija s povezavo odprta iz ozadja, pa s funkcijo iz knjižnice »poslušamo«, kdaj se to zgodi, in ob odprtju dobimo povezavo, prikazano na: Slika 29.

```
universalLinkSubscription = uriLinkStream.listen((Uri uri) {
  universalLink = uri.toString();
}, onError: (Object err) {});
```

Slika 29: Koda, kako dobimo povezavo, s katero je bila aplikacija odprta iz ozadja (Vir: lasten)

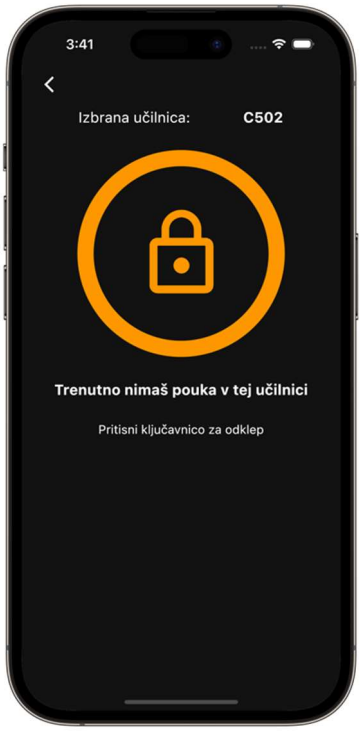
Zdaj v aplikaciji izvemo, s katero povezavo je bila aplikacija odprta, in to uporabimo, da izvemo, katera vrata je imel uporabnik namen odkleniti. Ker je povezava sestavljena iz dveh delov, je prvi, ki je statičen »scvapp://app.scv.si/open_door/«, in drugi, ki je koda (omenjena zgoraj pri entitetah v bazi podatkov), in je za vsaka vrata različna.

Nazadnje pa je bilo treba narediti prikaz za uporabnika in funkcijo, ki bo poklicala API za odklep vrat.

Želeli smo narediti uporabniku preprost in zanimiv prikaz. Da smo to dosegli, smo uporabili različne barve in ikone za različna stanja (napaka, ni dovoljenja, odklenjeno ...) To je razvidno iz slik: Slika 30, Slika 31, Slika 32 in Slika 33.

Slika 30: Prikaz za uporabnika, ko je uspešno odklenil vrata. (Vir: lasten in AppleFrames 3.0)

Slika 31: Prikaz za uporabnika, ko so se vrata zaklenila. (Vir: lasten in AppleFrames 3.0)

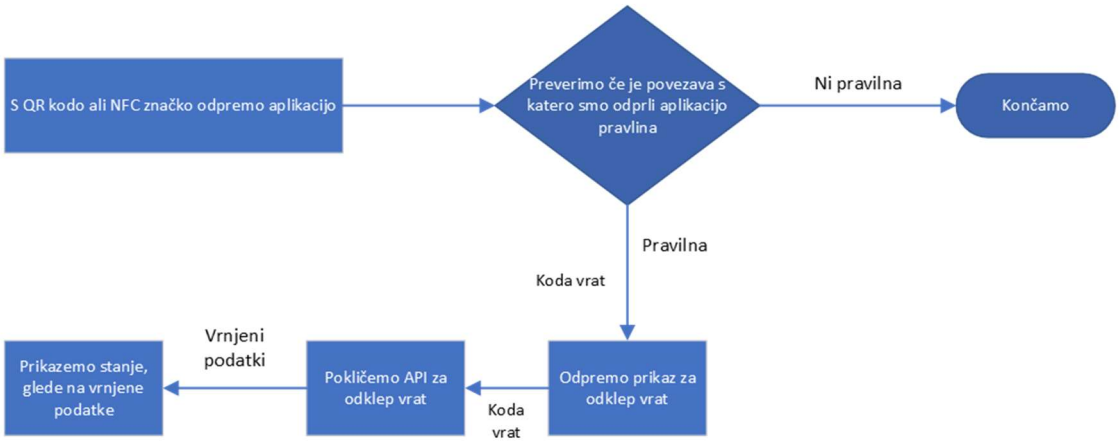


Slika 32: Prikaz za uporabnika, kadar nima dovoljenja vstopiti v učilnico. (Vir: lasten in AppleFrames 3.0)

Slika 33: Prikaz za uporabnika, kadar se zgodi napaka. (Vir: lasten in AppleFrames 3.0)

Da sprožimo odklep vrat, ob prikazu pokličemo funkcijo, ki »pokliče« API, da odklene vrata in nam sporoči stanje, ki ga potem prikažemo.

Zdaj je mobilna aplikacija nadgrajena in lahko z pomočjo QR-kode ali NFC-značke odklenemo vrata. Za lažje razumevanje poteka delovanja aplikacije je prikazan diagram. (Slika 34)

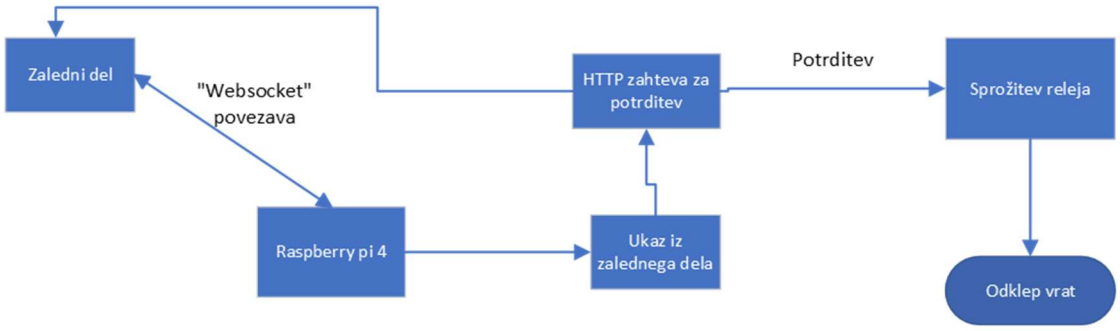


Slika 34: Prikazuje diagram poteke za delovanje mobilne aplikacije (Vir: lasten)

3.2.8 KONFIGURACIJA RASPBERRY PI 4

Da vrata odklenemo, je treba na krmilniku za vrata za kratek čas združiti dva kontakta – to bomo naredili s pomočjo Raspberry pi 4 in releji.

Zamislili smo si, da bo delovalo tako, da na Raspberry pi 4 teče program Python, ki je na zaledni del povezan preko »web socketa-a« in »posluša« ukaze iz zalednega dela. Ko dobi ukaz za odklep vrat iz zalednega dela, še enkrat preveri s http-zahtevkom na strežnik, da se prepriča, če je vrata res treba odkleniti. Ko se prepriča, preko GPIO sproži rele za 0,5 sekunde in tako sproži krmilnik za vrata, ki jih nato odklene. (Slika 35)



Slika 35: Prikazuje diagram poteka za delovanje Raspberry pi 4 (Vir: lasten)

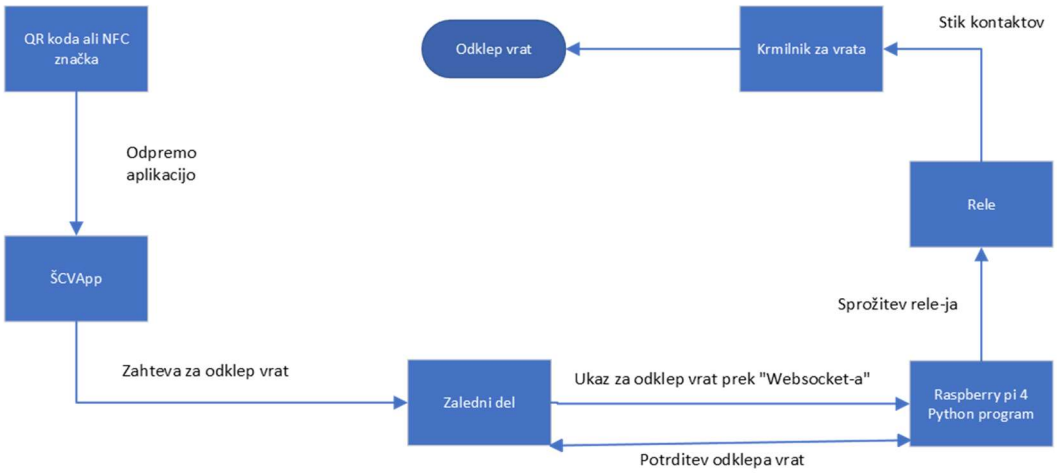
Za operacijski sistem za računalnik Raspberry pi 4 sva izbrala Ubuntu Server zaradi hitrega in robustnega delovanja.

Za program Python smo naredili, da se ob zagonu naprave zažene v ozadju in takoj prične z delovanjem.

3.2.9 KONČNO DELOVANJE SISTEMA

Ker smo želeli, da vrata odklenejo samo dijaki, ki imajo ob določeni uri pouk v določeni učilnici, je bilo treba v zaledni del dodati funkcijo, ki preveri, ali ima dijak učno uro takrat na urniku, in glede na to mu dovoli dostop. Urnik uporabnika pridobimo iz spletne strani eAsistent.

Sistem zdaj deluje tako, da dijak z QR-kodo ali z NFC-oznako odpre aplikacijo in potem aplikacija sporoči zalednemu delu, naj odklene vrata. Nato zaledni del preko »Websocket-a« sporoči programu Python na Raspberry pi 4, naj odklene vrata. Program Python se še enkrat prepriča in nato s pomočjo releja sproži kontakte na krmilniku, kar povzroči, da se vrata odklenejo. (Slika 36)



Slika 36: Prikazuje diagram poteka za celoten sistem za odklep vrat preko aplikacije. (Vir: lasten)

4 REZULTATI

4.1 ODKLEP VRAT S POMOČJO ŠCVAPP

Z odklepanjem vrat preko ŠCVApp-a ni bilo veliko težav, sistem smo preizkusili njegovo delovanje z večjim številom ljudi hkrati, kar pa ni pokazalo okvare. Težave so le z odklepanjem pravilnih vrat, saj se včasih zgodi, da se odklenejo vrata druge učilnice namesto tiste, ki naj bi se odklenila. Težava še ni bila v celoti odpravljena in je v fazi preizkušanja, sistem pa deluje pravilno.

Sistem je bil ustvarjen tako, da če dijak ali učitelj kartice nima pri sebi, lahko vrata vseeno odklene. Opazili smo, da še vedno večina dijakov in učiteljev raje uporabi kartico kot pa telefon, posledično zaradi tega, ker je ta sistem počasnejši kot kartični. Sistem bi lahko pospešili, če bi uporabljali NFC-bralnike in bi telefon oddajal samo NFC-signal, ki bi ga bralnik nato prebral.

4.1.1 TEŽAVE PRI PROGRAMU PYTHON

Pri pripravljenem programu v jeziku Python smo imeli na začetku velike težave, saj se je program po nekaj urah ugasnil, kar je zahtevalo ročni ponovni zagon ob vsakem takem dogodku. Po veliko poskusih reševanja tega problema in pisanja kode na novo, smo bili vedno neuspešni, zato smo se odločili, da bomo spremenili način delovanja programa, kar je naposled delovalo. Program smo spremenili tako, da »Websocket« deluje na primarni niti in da časovniki in sprožitev relejev delujejo na drugih nitih. V prejšnjih različicah programa je »Websocket« deloval na več nitih in zaradi tega je program po nekaj urah prenehal z delovanjem.

4.2 ODKLEP VRAT GLEDE NA DIJAKOV URNIK

Odklep vrat glede na dijakov urnik deluje odlično, saj lahko dijak v učilnico vstopi 5 minut pred začetkom ure.

Ena izmed posodobitev, ki bi bila možna za ta sistem, je, da bi pridobila urnik za vsakega dijaka posebej, saj ko se dijaki v razredu razdelijo v skupine, lahko vsak dijak iz tega razreda dostopa do vseh učilnic, ki so zajete za pouk teh skupin.

4.2.1 IZBOLJŠAVA KARTIČNEGA SISTEMA Z APLIKACIJO

Ker smo želeli, da kartični sistem deluje tako, da imajo dijaki dostop do učilnice takrat, ko imajo v njej pouk, smo v program, s katerim upravljamo krmilnik za kartični sistem,

podatke vnesli ročno, torej smo vnesli, kdaj ima kateri razred pouk. Pri tem so nastale težave, in sicer, če imajo dijaki na urniku nadomeščanje, potem v učilnico nimajo vstopa oziroma če imajo kakšno odpadlo uro ali uro v drugih učilnicah, lahko vanje vstopijo, čeprav nimajo v njih pouka.

Zato smo naredili aplikacijo, ki bo na vsako uro, kadar je pouk, osvežila dostope do učilnic. To pomeni, da bodo dijaki glede na urnik lahko vstopali v učilnico, program pa bo upošteval nadomeščanja, odpadle ure ipd.

4.3 VARNOST SISTEMA

Ko smo razvijali sistem, smo razmišljali tudi o varnosti, zato smo krmilnik za vrata in računalnik, s katerim lahko upravljamo krmilnik, postavili na svoje omrežje.

Da smo se lahko prijavili v strežnik Ubuntu, smo uporabili način prijave s privatnim in javnim ključem.

Za varnost naše aplikacije smo poskrbeli tudi tako, da program Python še enkrat preveri s HTTP-zahtevo, če je res treba odpreti vrata. Tudi mobilna aplikacija je varna, saj uporablja prijavo z Microsoftovim računom.

Edina varnostna težava je pri našem sistemu kopiranje drugih kartic. To pomeni, da lahko nekdo prekopira kartico od učitelja, ki ima poln dostop, in si ga s tem omogoči tudi sam (primer tega je orodje Flipper Zero). To težavo smo rešili z mobilno aplikacijo, pri kateri ni mogoče ničesar kopirati, prav tako pa si lahko mobilno aplikacijo si zaščitimo z biometričnim odklepanjem.

4.3.1 VARNOST UPORABNIKOVIH PODATKOV

Varnost uporabnikovih podatkov je danes zelo pomemben dejavnik pri vseh sistemih zaradi GDPR.

Pri našem sistemu hranimo podatke o uporabnikih na lokalnem strežniku, ki se nahaja v šoli. Strežnik je pod ključem v kabinetu, kamor imajo dostop samo določeni profesorji in skrbniki omrežja. Postavljen je v strežniški omari, ki ima svoj ključ. V aplikaciji ŠCVApp hranimo le Microsoftov »ID« uporabnika in razred uporabnika, za ostale uporabnikove podatke pa skrbi Microsoft.

5 RAZPRAVA

Hipoteze so bile sledeče:

1. hipoteza: Z uporabo aplikacije ŠCVApp in kartice kot ključa bo olajšano dostopanje do učilnic in drugih prostorov v šoli.

V raziskavi smo ugotovili, da je bil sistem v mesecu januarju uporabljen 1429-krat, kar pomeni, da je bil povprečno uporabljen 68-krat dnevno, če štejemo samo delovne dni.

V okviru opazovanja uporabe sistema smo v dveh urah ugotovili, da je bilo 28 uporabnikov, ki so vrata odprli s kartico, 2 uporabnika, ki so uporabili mobilno aplikacijo, medtem ko noben uporabnik ni vrata odprl s ključem.

Graf 1: Prikaz tipov odklepanja vrat

Našo hipotezo potrjujemo.

2. hipoteza: Bralnike kartic in krmilnik bo možno povezati z mobilno aplikacijo ŠCVApp, kar bo posledično omogočalo odklepanje preko aplikacije.

Opazili smo, da je bil sistem za odklepanje preko mobilne aplikacije prvotno zamišljen kot sekundarni način odklepanja, ki je namenjen le uporabnikom, ki so pozabili svojo kartico. To se odraža v rezultatih naše raziskave, kjer sta od 30 uporabnikov le 2

uporabnika uporabila odklepanje z mobilno aplikacijo, medtem ko so ostali uporabili kartico (glej Graf 1). Našo hipotezo potrjujemo.

3. hipoteza: Sistem bo omogočal dostop dijakom v času, ko imajo na urniku pouk v učilnici.

Glavni cilj te raziskovalne naloge je bila uporabnost. Želeli smo ustvariti sistem, ki bi bil koristen in uporaben. Tako smo si zamislili, da bi dijakom dovolili dostop do učilnic samo ob urah, ob katerih imajo pouk. To smo dosegli tako, da smo preverjali dijakov urnik ter se tako prepričali, ali ima dostop. Hipotezo potrjujemo, saj sistem uspešno deluje in dijakom omogoča dostop .

6 POVZETEK

Ob prvem vstopu v ŠC Velenje dijaki in učitelji dobijo kartico, ki služi kot evidenca delovnega časa oz. prisotnosti v šoli na sploh ter kot jamstvo za prevzem šolskih malic. Pri tem smo prišli do ideje pametnega odklepanja učilnic. Uporabniki bodo lahko odklepali vrata ali z uporabo kartice, ki jo bodo nosili s seboj, ali z aplikacijo ŠCVApp na svojih pametnih telefonih. Sistem bo nastavljen tako, da bo v učilnico omogočeno vstopiti samo dijaku, ki bo imel v tistem trenutku pouk v njej. To bo omogočilo boljši nadzor nad tem, kdo je v določenem prostoru, posledično pa bo to prispevalo k večji varnosti v šoli. Prednosti uporabe te tehnologije bodo večja varnost, manjša možnost izgube ključa in enostavnejše dostopanje do prostorov.

7 ZAKLJUČEK

V prvem delu raziskovanje smo se osredotočili na postavitev sistema bralnikov kartic. To je vključevalo montažo bralnikov, montažo krmilnika in vzpostavljanje sistema Raspberry Pi s sistemom Ubuntu Server. Začeli smo z montažo bralnikov v učilnicah in se nato prestavili na montažo bralnikov v kabinetu. Po uspešni montaži in preizkusu sistema smo vzpostavili strežnik s sistemom Windows Server, na katerega smo naložili programsko opremo za upravljanje krmilnika. Po opravljenem smo v programsko opremo uvozili dijake in učitelje ter jim ročno dodali pravice za dostop.

V drugem delu smo se osredotočili na kodo. To vključuje aplikacijo za sistem Windows, ki v podatkovno bazo vpisuje podatke, in sicer kateri razred ima v kateri izmed teh učilnic pouk. Ta del vključuje tudi programiranje aplikacije ŠCVApp, v kateri smo dodali možnost odklepa vrat preko NFC- ali QR-kode. Celoten sistem je torej ustvarjen tako, da glede na urnik dijakov pregleda razrede, ki imajo v učilnici pouk, in na osnovi tega dostop dovoli ali prepove.

Celoten sistem smo preizkusili z večjo skupino dijakov in dosegli pozitivne rezultate, saj se sistem ni okvaril.

8 ZAHVALA

Vesela smo, da smo z raziskovalno nalogo dosegli, kot smo si zadali. Najprej bi se zahvalili mentorjema Urošu Remenihu in Samo Železniku za vso pomoč pri delu. Zahvaljujemo se tudi Iztoku Osredkarju in Mestni občini Velenje za podarjen strežnik, učiteljici Sonji Lubej za lektoriranje raziskovalne naloge ter učiteljici Klementini Selčan za lektoriranje angleške verzije avtorskega izvlečka in lektoriranje raziskovalne naloge. Zahvaljujemo se tudi vsem preizkuševalcem, ki so nama pomagali pri razvojni različici aplikacije.

9 PRILOGE

```
def main():
    lgpio.gpio_claim_output(gpio_chip, 22)
    lgpio.gpio_write(gpio_chip, 22, 1)
    controlers:list[Controler] = getControllersFromENVFile()
    if len(controlers) == 0:
        print('No controlers found')
        return

    sio = socketio.Client()

    @sio.event
    def connect():
        print('connection established')

    @sio.event
    def disconnect():
        print('disconnected from server')

    @sio.on('open_door')
    def pass_open_door(data):
        for controler in controlers:
            if controler.code == data:
                controler.run()
                return "ok"


    sio.connect(API_URL,headers={'code': controlers[0].code,
'secret': controlers[0].api_secret})
    sio.wait()

if __name__ == '__main__':
    while True:
        try:
            main()
        except Exception as e:
            print(e)
            time.sleep(5)
```

Slika 37: Python program za Raspberry Pi

```
API_URL:str = 'https://backend.app.scv.si'
NUMBER_OF_DOORS:int = 2
gpio_chip = lgpio.gpiochip_open(0)

def getControllersFromENVFile():
    controlers:list[Controler] = []
    list_of_pins:list[int] = [23,24]
    interval:int = 1
    with open('.env', 'r') as f:
        controler_code:str = None
        controler_secret:str = None
        for line in f.readlines():
            line = line.strip()
            if
line.startswith('CONTROLER{}_CODE'.format(interval)):
                controler_code = line.split('=')[1].strip()
            elif
line.startswith('CONTROLER{}_SECRET'.format(interval)):
                controler_secret = line.split('=')[1].strip()
            if controler_code and controler_secret:
                controler:Controler = Controler(API_URL,
controler_code,
controler_secret,gpio_chip,list_of_pins[interval-1])
                controlers.append(controler)
                interval += 1
                if interval > NUMBER_OF_DOORS:
                    break
                controler_code = None
                controler_secret = None
    return controlers
```

Slika 38: Funkcija in spremenljivke za Python program

```
class Controler:
    timout_between_commands:float = 3
    time_lock_open:float = 1
    tread:threading.Thread = None
    door_is_in_timeout:bool = False
    gpio_chip=None
    pin_number:int=None

    def __init__(self, api_url:str, code:str,
api_secret:str,gpio_chip,pin_number:int):
        self.api_url:str = api_url
        self.code:str = code
        self.api_secret:str = api_secret
        self.tread =
threading.Thread(target=self.treadFunction)
        self.gpio_chip = gpio_chip
        self.pin_number = pin_number

    def run(self):
        if self.door_is_in_timeout:
            return
        self.tread =
threading.Thread(target=self.treadFunction)
        self.tread.start()

    def treadFunction(self):
        if self.door_is_in_timeout:
            return
        self.door_is_in_timeout = True
        url = self.api_url + '/pass/door/is_opened'
        res =
requests.get(url,headers={"door_code":self.code,"access_s
ecret":self.api_secret})
        if res.status_code == 200:
            self.hardware_open_door()
            time.sleep(self.time_lock_open)
            self.hardware_close_door()
            time.sleep(self.timout_between_commands)
            self.door_is_in_timeout = False

    def hardware_open_door(self):
        print('open door hardware')
        lgpio.gpio_write(self.gpio_chip, self.pin_number,
1)

    def hardware_close_door(self):
        print('close door hardware')
        lgpio.gpio_write(self.gpio_chip, self.pin_number,
0)
```

Slika 39: Razred za Python program

Povezava do kode za python program:
https://github.com/SCVApp/DoorOpenerRaspberryPI

Povezava do kode za mobilno aplikacijo: https://github.com/SCVApp/SCVAppFlutter

Povezava do kode za zaledni del: https://github.com/SCVApp/SCVAppBackend

10 VIRI IN LITERATURA

(1) https://www.postgresql.org/about/ (3. 1. 2023)

(2) https://ubuntu.com/server/docs (18. 2. 2023)

(3) https://www.techrepublic.com/article/ubuntu-server-the-smart-persons-guide/ (16. 1. 2023)

(4) https://www.python.org/doc/essays/blurb/ (18. 2. 2023)

(5) https://docs.nestjs.com (18. 2. 2023)

(6) https://www.flutter.dev/ (18. 2. 2023)

(7) https://medium.com/flutter (18. 2. 2023)

(8) https://dev.to/alvarotorresc/nestjs-introduction-ekd (18. 2. 2023)

(9) https://upload.wikimedia.org/wikipedia/commons/thumb/c/c3/Python-logo-notext.svg/1200px-Python-logo-notext.svg.png (18. 2. 2023)

(10) https://si.farnell.com/buy-raspberry-pi (18. 2. 2023)

(11) https://www.raspberrypi.com/products/raspberry-pi-4-model-b/ (18. 2. 2023)

(12) https://leoss.si/strokovnjak_svetuje/29/radiofrekvencna_identifikacija_rfid (22.2.2023)

(13) https://priporocam.si/nfc-prvic-slisim-za-to/ (23. 2. 2023)

(14) https://www.nomtek.com/blog/what-are-nfc-tags (23 .2. 2023)

(15)https://www.elektrospoji.si/info-tocka/produktne-novice/aktualne-novice/varnostni-releji-s-prisilno-vodenimi-kontakti-serije-safeseries2 (23. 2. 2023)