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

MLADI RAZISKOVALCI ZA RAZVOJ SAŠA REGIJE

RAZISKOVALNA NALOGA

RAZVOJ IN IMPLEMENTACIJA PAMETNIH OMARIC ZA ŠOLSKE DIJAKE

Tematsko področje: Računalništvo

Avtorji:

Miha Šafranko, 4. letnik
Blaž Herman, 3. letnik
Tin Keserin, 3. letnik

Mentorja:

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

Velenje, 2024

Raziskovalna naloga je bila opravljena na ŠC Velenje, Elektro in računalniška šola, 2024.

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

Datum predstavitve: marec 2024

KLJUČNA DOKUMENTACIJSKA INFORMACIJA

ŠD Elektro računalniška šola Velenje, šolsko leto 2023/2024

KG kartični dostop / IOT / pametne omarice

AV ŠAFRANKO, Miha / HERMAN, Blaž / KESERIN, Tin

SA ŽELEZNIK, Samo / REMENIH, Uroš

KZ 3320 Velenje, Trg mladosti 3

ZA ŠC Velenje, Elektro in računalniška šola, 2024

LI 2024

IN **Razvoj in implementacija pametnih omaric za šolske dijake**

TD Raziskovalna naloga

OP

IJ SL

JI sl / en

AI V današnjem digitalnem svetu, kjer tehnologija prevladuje v vsakdanjem življenju, se pojavljajo inovativne rešitve tudi v šolskem okolju. Eden izmed najnovejših projektov so pametne omarice, ki predstavljajo odličen primer uporabe tehnologije za izboljšanje šolske izkušnje. Pametne omarice so namenjene ne le varnemu shranjevanju šolskih pripomočkov, ampak tudi omogočajo avtomatiziran dostop dijakov do svojih stvari, kar bistveno olajša vsakodnevno šolsko rutino.

KEY WORDS DOCUMENTATION

ND Elektro računalniška šola Velenje, šolsko leto 2021/2022

CX card access / IOT / smart lockers

AU ŠAFRANKO, Miha / HERMAN, Blaž / KESERIN, Tin

AA ŽELEZNIK, Samo / REMENIH, Uroš

PP 3320 Velenje, Trg mladosti 3

PB ŠC Velenje, Elektro in računalniška šola, 2024

PY 2024

TI **Development and implementation of smart lockers for school students**

DT Raziskovalna naloga

NO

LA SL

AL sl / en

AB In today's digital world, where technology predominates in everyday life, innovative solutions are also emerging in the school environment. One of the newest projects is smart lockers, which represent an excellent example of using technology to enhance the school experience. Smart lockers are designed not only for safe storage of school supplies but also to enable automated access for students to their belongings, significantly easing the daily school routine.

KAZALO VSEBIN

1 Uvod 1

2 Hipoteze 2

3 Pregled stanja tehnike in objav 3

3.1 Programska orodja 3

3.1.1 Arduino IDE 3

3.1.2 Sprint Layout 4

3.2 Iskanje podobnih rešitev na trgu 5

3.3 Odprtokodne rešitve 5

4 Material in metode dela 7

4.1 Osnovna zasnova projekta 7

4.2 Digitalno upravljanje omaric 7

4.3 Izdelava distančnika 10

4.4 Izdelava šablone za montiranje 10

4.5 Montiranje ključavnic 11

4.6 Prilagojena PCB plošča 13

4.7 Povezovanje komponent 15

5 Rezultati 15

5.1 Težave 16

6 Razprava 16

7 Rezultati 18

8 Zaključek 18

9 Zahvala 20

10 Priloge 21

KAZALO SLIK

Slika 1: Prikaz logotipa mikrokontrolerja Arduino 3
Slika 2: Logotip programske opreme Sprint Layout 4
Slika 3: Skica postavitve komponent sistema omaric 7
Slika 4: Diagram poteka, za logiko kartic 8
Slika 5: Diagram poteka, za logiko omaric 8
Slika 6: Diagram poteka, za logiko izpisov uporabniku 9
Slika 7: Logika mikrokontrolerja za upravljanje omaric 9
Slika 8: Elektronske ključavnice z distančnikom 10
Slika 9: Šablona za montiranje ključavnic 11
Slika 10: Montirana električna ključavnica 12
Slika 11: Kanal za skrivanje kabla ključavnice 12
Slika 12: Držalo za zaklepanje vrat 12
Slika 13: Digitalna slika rezkanega vezja, za priklop komponent sistema omaric 13
Slika 14: PCB plošča z dodanimi kontakti 14
Slika 15: PCB plošča z povezanim mikrokontrolerjem 14
Slika 16: Končna vezava komponent 15
Slika 17: Definicije priključkov v kodi 21
Slika 18: Primer določanja priklopa relayu in drugih definicij 22
Slika 19: Glavna funkcija sistema 23
Slika 20: Funkcija, ko uporabnik prvič skenira kartico 24
Slika 21: Funkcija, ko uporabnik drugič skenira kartico 25
Slika 22: Izpis uporabniku, če so vse omarice zasedene 25

SEZNAM OKRAJŠAV IN SIMBOLOV

IDE - Integrated development enviroment

Relay – elektronsko upravljeno stikalo

PCB – Printed Circuit Board (plošča tiskanega vezja)

1 UVOD

V današnjem hitro razvijajočem se svetu, kjer tehnologija igra ključno vlogo pri vsakodnevnih opravilih in procesih, se pojavlja vedno večja potreba po inovativnih rešitvah, ki bi olajšale naše življenje in izboljšale našo produktivnost. Ena od področij, kjer se to kaže, je tudi v izobraževalnih ustanovah, kjer se soočamo s številnimi izzivi v vsakodnevnem delovanju.

Na naši šoli se je pojavil precejšen problem, povezan z nošenjem težke in občutljive opreme, ki jo dijaki potrebujejo za svoje izobraževalne dejavnosti. Slednje postane še posebej problematično, ko se premikajo iz enega konca šole na drugega, se udeležujejo različnih aktivnosti ali pa odhajajo na odmore. Pogosto se zgodi, da morajo težke nahrbtnike ali druge torbe prenašati po stopnicah ali hodnikih, kar ne le obremenjuje dijake, ampak lahko tudi poveča tveganje za poškodbe ali izgubo dragocene opreme.

Z željo po izboljšanju tega stanja smo se odločili, da bi lahko naši dijaki shranjevali svoje stvari v omaricah, ki bi bile nameščene v šoli. Vendar pa smo želeli to osnovno idejo še nadgraditi in modernizirati. Zavedali smo se, da bi lahko uporaba sodobnih tehnoloških rešitev prinesla dodatne koristi in olajšave tako dijakom kot tudi šoli.

Tako smo se odločili raziskati možnosti vzpostavitve sistema pametnih omaric na naši šoli. Pametne omarice bi omogočale varno shranjevanje osebnih stvari dijakov, obenem pa bi bile opremljene s tehnološkimi funkcijami, ki bi olajšale njihovo uporabo in zagotovile dodatno udobje ter varnost. S tem bi želeli prispevati k izboljšanju izkušenj dijakov na šoli ter hkrati povečati učinkovitost in funkcionalnost našega izobraževalnega okolja.

2 HIPOTEZE

1. Hipoteza

S pomočjo mikrokontrolerja Arduino, lahko upravljamo veliko količino omaric.

2. Hipoteza

Omarice bodo delovale z dijaškimi karticami za identifikacijo.

3. Hipoteza

Omarice se varne iz programskega in fizičnega vidika.

4. Hipoteza

V primeru napak, se te lahko hitro odpravijo zaradi modularnosti sistema.

3 PREGLED STANJA TEHNIKE IN OBJAV

3.1 PROGRAMSKA ORODJA

3.1.1 Arduino IDE

Arduino IDE je osnovna programska oprema, ki je ključna za razvoj aplikacij za mikrokontrolerje Arduino. Ta orodja omogočajo programerjem, da pišejo, preverjajo in nalagajo kodo na Arduino naprave. Arduino IDE zagotavlja intuitivno uporabniško izkušnjo in podpira široko paleto funkcij za olajšanje razvoja in debugiranja. S svojo preprosto in enostavno uporabo je Arduino IDE priljubljena izbira med ustvarjalci projektov s platformo Arduino. To smo uporabili, ker je priporočeno orodje za kodiranje na Aduiono platformi in smo se o njej že učili.

Slika 1: Prikaz logotipa mikrokontrolerja Arduino

3.1.2 Sprint Layout

Sprint Layout je specializirano orodje za načrtovanje tiskanih vezij, ki omogoča uporabnikom, da ustvarijo in urejajo vezja na računalniški platformi. Ta programsko orodje ponuja napredne funkcije in intuitiven vmesnik, ki olajšuje oblikovanje in optimizacijo tiskanih vezij za različne elektronske projekte. Sprint Layout se pogosto uporablja v elektronski industriji zaradi svoje zmožnosti natančnega načrtovanja in hitrega iteriranja pri razvoju tiskanih vezij. Zanj smo se odločili, ker smo se že učili o njegovi uporabi in smo v njem bili že spretni.

Slika 2: Logotip programske opreme Sprint Layout

3.2 ISKANJE PODOBNIH REŠITEV NA TRGU

Pred dejansko implementacijo pametnih funkcij v omarice smo temeljito raziskali že obstoječe rešitve, ki se uporabljajo na trgu. Pri tem smo opazili, da različna kopališča in fitnesi že uporabljajo podobno tehnologijo, ki deluje po podobnih principih. Ti objekti uporabljajo pametne omarice, ki se odklepajo s čipi na zapestnicah ali karticah, opremljenih s senzorji za tovrstne metode, hkrati pa imajo tudi senzorje za zaznavo, ali so omarice odprte ali zaprte.

Kljub opaženim rešitvam na trgu smo se odločili za drugačen pristop k zaklepanju omaric. Želeli smo uporabiti metodo, ki bi bila bolj enostavna, z manj možnostmi za napake in bi obenem zagotovila optimalno uporabniško izkušnjo. S tem smo se izognili zapletenim postopkom odklepanja in zaklepanja ter olajšali uporabo omaric za naše dijake.

Vredno je omeniti, da so podobne sisteme, kot jih uporabljajo fitnesi in bazeni, cenovno ocenili na približno 30 tisoč evrov. Kljub temu smo se odločili za drugačen pristop, ki je bil bolj ekonomičen in hkrati učinkovit, ter se je bolje prilagajal našim potrebam in zahtevam projekta. Ta odločitev je bila sprejeta na podlagi celovite analize obstoječih rešitev na trgu, stroškovnih vidikov in prednostnih nalog našega projekta.

3.3 ODPRTOKODNE REŠITVE

Za dosego čim večje razpoložljivosti virov in poenostavitev povezovanja med različnimi deli našega sistema, smo sprejeli odločitev, da bomo uporabili široko paleto odprtokodnih rešitev. Med najbolj vplivnimi orodji, ki smo se odločili vključiti v naš projekt, so Arduino mikrokontrolerji. Ti mikrokontrolerji nam omogočajo izjemno prilagodljivost pri izdelavi in nadzoru različnih funkcij našega sistema.

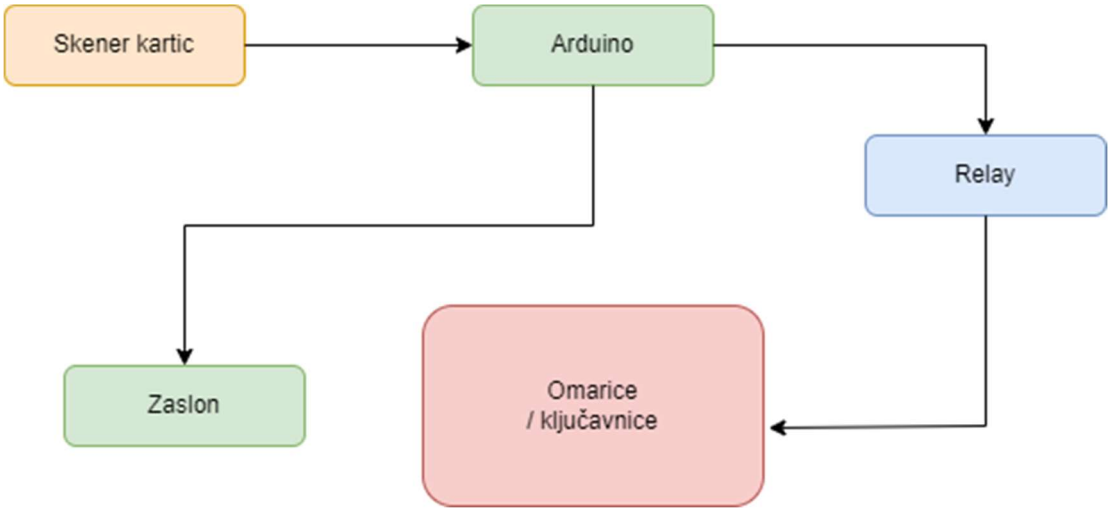
Uporaba Arduino mikrokontrolerjev prinaša več prednosti. Prvič, nudijo nam izjemno preprosto in intuitivno možnost povezovanja s številnimi senzorji, aktuatorji in drugimi napravami, ki jih potrebujemo v našem sistemu. Z njihovo pomočjo lahko enostavno vzpostavimo dokumentirane vezave za naše priključke, kot so zasloni za izpise informacij, skenerji kartic za identifikacijo uporabnikov ter releji za proženje ključavnic omaric.

Poleg tega so Arduino mikrokontrolerji široko podprti v skupnosti in imajo obsežno dokumentacijo ter veliko število že razvitih knjižnic in primerov uporabe. To nam omogoča, da hitro in učinkovito implementiramo želene funkcionalnosti brez potrebe po odpravljanju zapletenih tehničnih težav.

4 MATERIAL IN METODE DELA

4.1 OSNOVNA ZASNOVA PROJEKTA

Na samem začetku, smo najprej rabili ugotoviti, kako bi sploh dodali pametno funkcionalnost našim omaricam. Zato smo potrebovali osnovni načrt, kakšne sisteme in naprave potrebujemo, za takšno integracijo.



Slika 3: Skica postavitve komponent sistema omaric

Sistem naj bi po principu deloval tako, da uporabik skenira kartico, mikrokontroler Arduino prikaže informacijo o omarici na zaslonu, nato pa sproži relay, da omogoči tok elektrike do ključavnice za omarico, da se odpre.

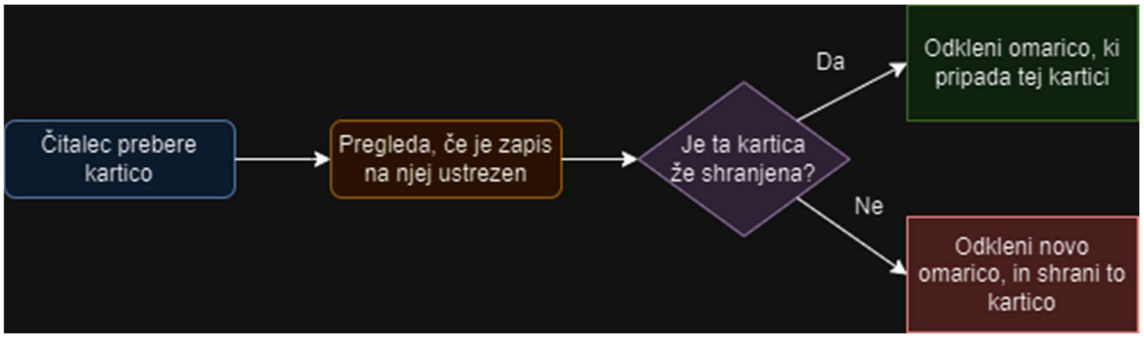
Tak pristop se nam je zdel najbolj optimalen, ker ima najmanj premikajočih delov in vmesnikov, ki bi lahko povzročali napake, sistem pa je dovolj modularen, da se v primeru napake, te komponente lahko zamenjajo.

4.2 DIGITALNO UPRAVLJANJE OMARIC

Ko smo dokončno oblikovali logiko delovanja za naše pametne omarice, smo se soočili z izzivom programiranja mikrokontrolerjev, ki so bili ključni del našega sistema. V našem primeru smo imeli 72 omaric, medtem ko je bilo največje število priključkov na enem mikrokontrolerju omejeno na 36. Da bi lahko učinkovito upravljali vse omarice, smo se odločili uporabiti 2 mikrokontrolerja, pri čemer vsak nadzoruje polovico omaric.

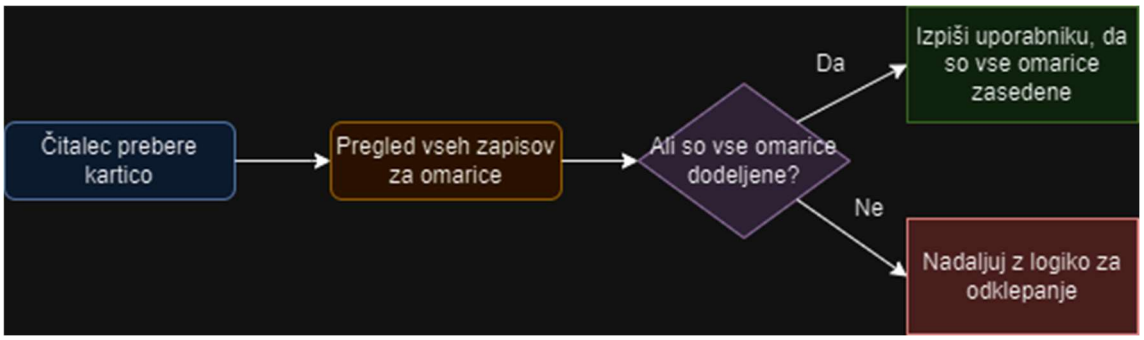
Programiranje mikrokontrolerjev je zahtevalo izdelavo kompleksne logike, ki je zajemala več ključnih funkcij:

1. **Odklepanje omaric:** Razviti smo morali algoritem za prepoznavanje uporabnikov ter ustreznega odklepanja omaric po uspešni identifikaciji. To je vključevalo uporabo podatkov iz skenerjev kartic ter aktiviranje ustrezne ključavnice za odklepanje omarice.



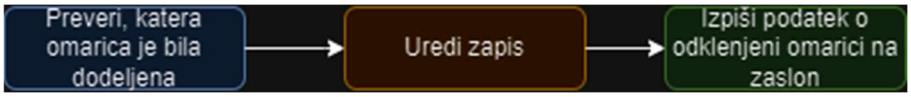
Slika 4: Diagram poteka, za logiko kartic

2. **Prepoznavanje stanja omaric:** Vsak mikrokontroler je moral biti sposoben zaznati, ali je posamezna omarica trenutno zasedena ali prosta. To smo dosegli z algoritmom, ki pregleda stanje vseh omaric, in nato nadaljuje z primerno logiko.



Slika 5: Diagram poteka, za logiko omaric

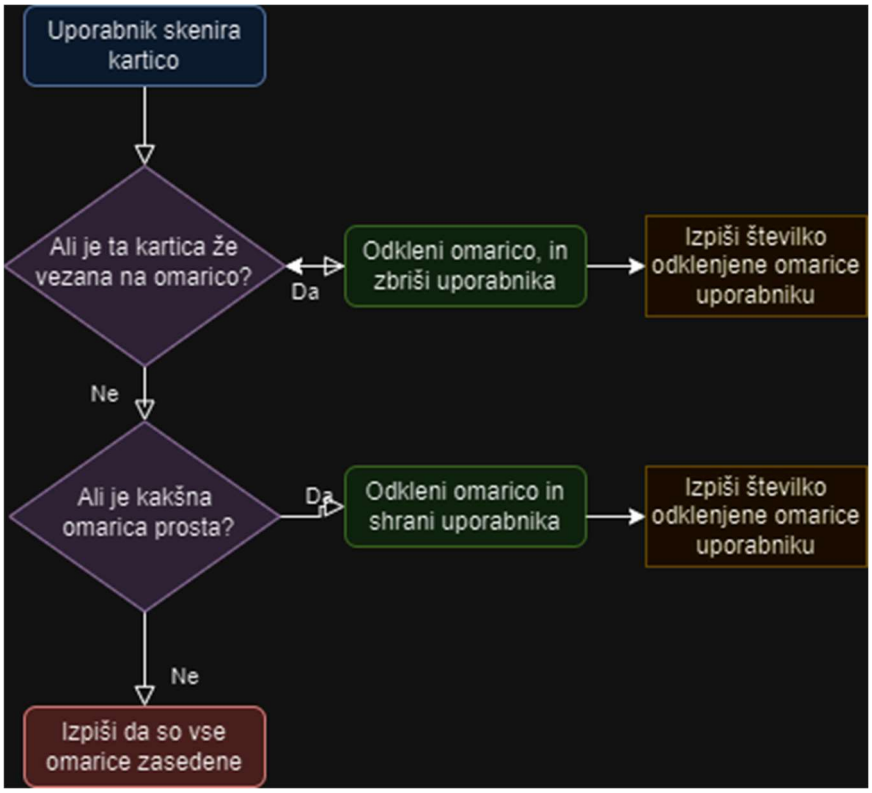
3. **Izpisovanje informacij uporabniku:** Na koncu smo bili odgovorni tudi za izdelavo logike za izpisovanje informacij uporabniku. To vključuje informacije o uspešnem odklepanju omarice, opozorila v primeru zasedenosti, prikazovanje in morebitne druge pomembne informacije na zaslonu.



Slika 6: Diagram poteka, za logiko izpisov uporabniku

V celoti je bilo programiranje mikrokontrolerjev zahteven proces, ki je zahteval temeljito načrtovanje, testiranje in optimizacijo. Kljub temu smo uspeli razviti učinkovito programsko rešitev, ki je omogočila nemoteno in zanesljivo delovanje našega sistema pametnih omaric, pri čemer smo izkoristili vse prednosti, ki jih ponuja uporaba mikrokontrolerjev v takšnih aplikacijah.

Končna logika je potekala v prikazanem smislu:



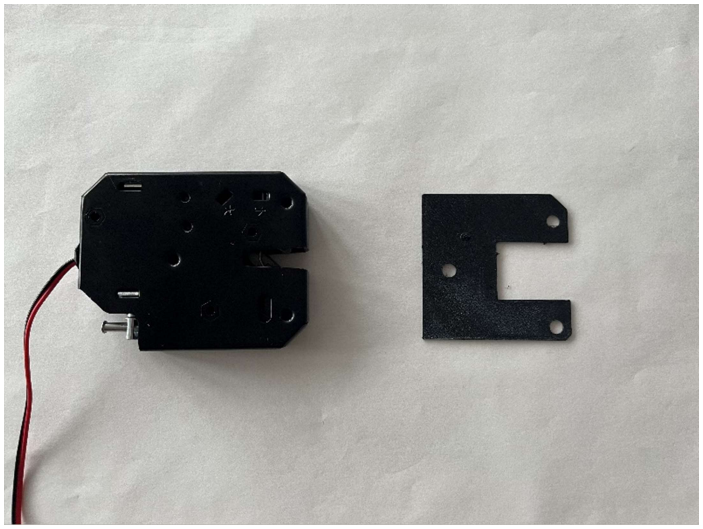
Slika 7: Logika mikrokontrolerja za upravljanje omaric

Končna koda je oddana v prilogi. Deljena je tudi na platformi Github.[1]

[1] https://github.com/safkom/DijakiOmare/blob/main/Koda/OmaraDijaki.ino

4.3 IZDELAVA DISTANČNIKA

Po opravljeni meritvah na omaricah in ključavnicah smo ugotovili, da moramo pred montažo omaric na ključavnice namestiti distančnik, da bo omarice mogoče zapirati in odpirati brez zatikanja. Ker imamo na naši šoli 3D tiskalnike smo se odločili da je to najbolj optimalno. V programu FUSION 360 smo najprej zmodelirali distančnik. Nato smo ga s programom ULTIMAKER CURA prenesli v jezik (.gcode), ki ga razume 3D tiskalnik. Nato smo natisnili distančnik. Posameznik distančnik je za tiskanje potreboval 33 minut.

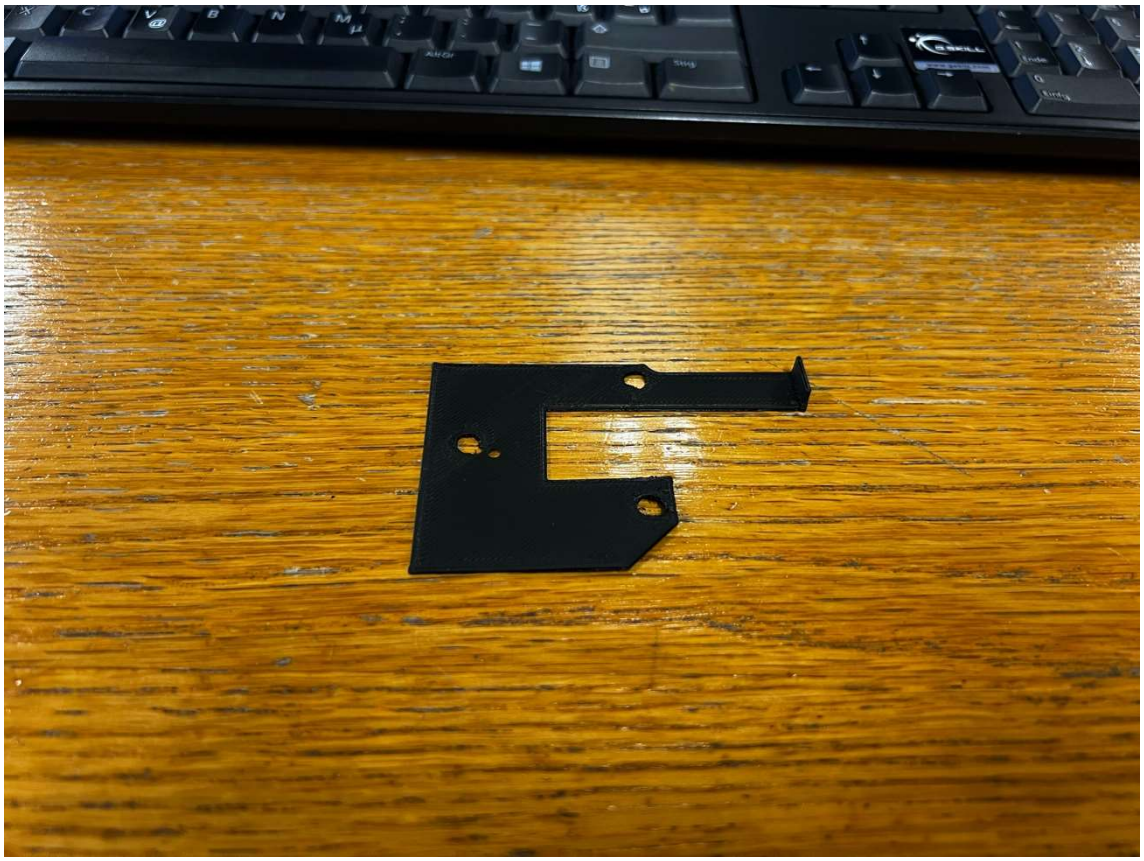


Slika 8: Elektronske ključavnice z distančnikom

4.4 IZDELAVA ŠABLONE ZA MONTIRANJE

Pri montiranju testne ključavnice, smo opazilo, da naš distančnik ni čisto ob robu, in bi morali ključavnico namestiti malo bolj od roba omarice. Da ne bi te meritve bile različne na vsaki omarici, smo naredili še šablono za ključavnico, da smo pri montiranju imeli

točno določene razdalje za montiranje, kar je omogočalo zaklepanje omaric brez zatikanja.



Slika 9: Šablona za montiranje ključavnic

4.5 MONTIRANJE KLJUČAVNIC

Ko smo stestirali vmesnik na prvi omarici smo na naslednjih 72 omaric namestili držala na vrata, potem pa smo zvrtali luknje in vstavili električne ključavnice z distančnikom. Da smo skrili kable, smo dodali še kanale. S tem smo poskrbeli, da so omarice varne in urejene. Pri uporabi ključavnic smo izberali med elektro mehanskimi ključavnicami in med elektro magnetnimi ključavnicami. Na koncu smo se odločili za elektronsko mehanične ključavnice, ker so bolj cenovno ugodne, omogočajo opravljanje vseh omaric iz enega mesta in s tem lahko lažje nadgaradimo sistem, so bolj učinkovite in omogočajo enostavnejše upravljanje.

Slika 10: Montirana električna ključavnica

Slika 11: Kanal za skrivanje kabla ključavnice

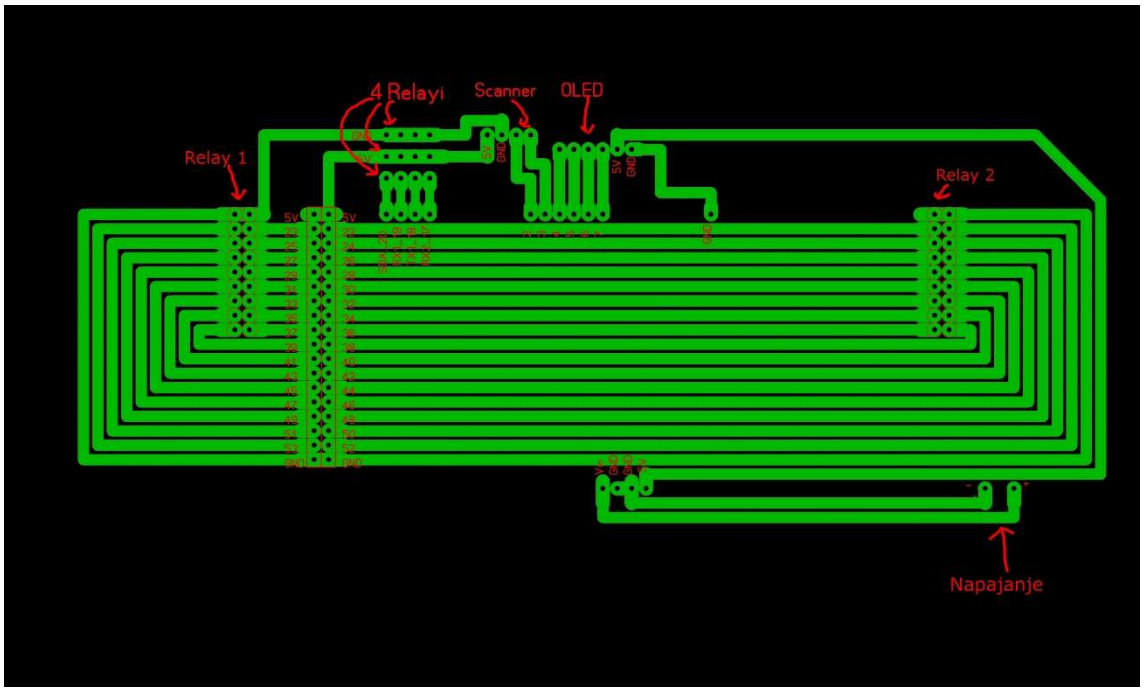
Slika 12: Držalo za zaklepanje vrat

4.6 PRILAGOJENA PCB PLOŠČA

Pri sestavljanju vseh komponent smo naleteli na težavo. Da bi vse komponente povezali med sabo, bi potrebovali veliko žic, kar pa bi lahko pripeljalo do kratkih stikov, slabih kontaktov, ali pa celo do prekinitve povezave med kritičnimi komponentami.

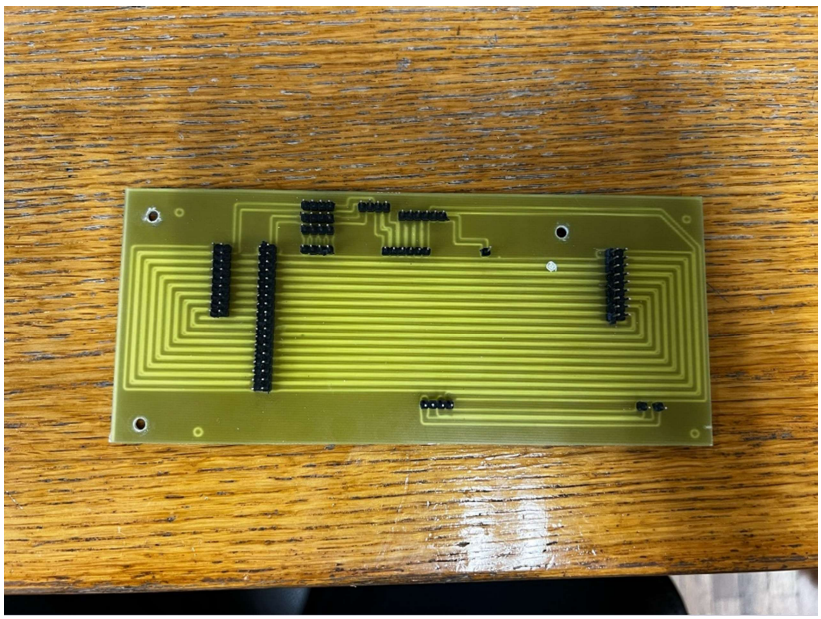
Zato smo se odločili, da bomo naredili svojo PCB ploščo, ki bo ustrezala našim zahtevam. Hoteli smo nanjo postaviti vse fiksne komponente in priključke za ostale komponente, ki niso fiksne oz. so dlje oddaljene od mikrokontrolerja Arduino (ključavnice, bralec kartic, zaslon). To nam je tudi pomagalo pri boljši organizaciji povezav, ker smo na plošči lahko naredili oddelke za določene povezave, in jih tudi označili.

Samo PCB ploščo smo narisali v programu Sprint Layout, ki smo ga že obvladali, in nam je ponujal najboljšo preglednost in kompatibilnost z našim rezkarjem za plošče. Imel je tudi že vgrajene mere za naš mikrokontroler, in smo z tem lažje urejali povezave za vsako komponento.

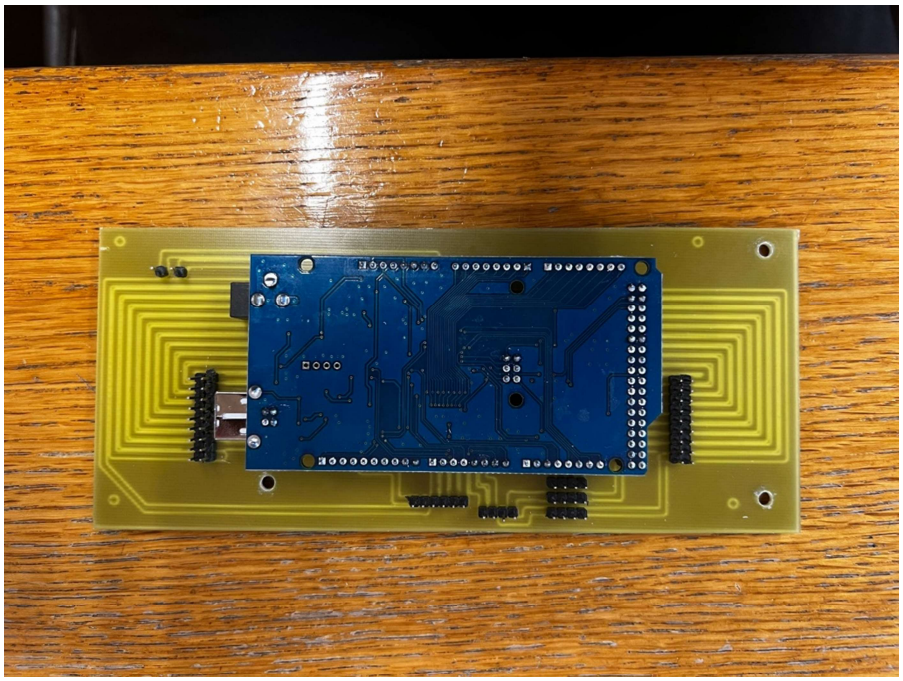


Slika 13: Digitalna slika rezkanega vezja, za priklop komponent sistema omaric

Ko je bila plošča izrezkana, smo nanjo dodali kontakte, s katerimi smo lahko nataknili mikrokontroler, in ga povezali z drugimi komponentami.



Slika 14: PCB plošča z dodanimi kontakti

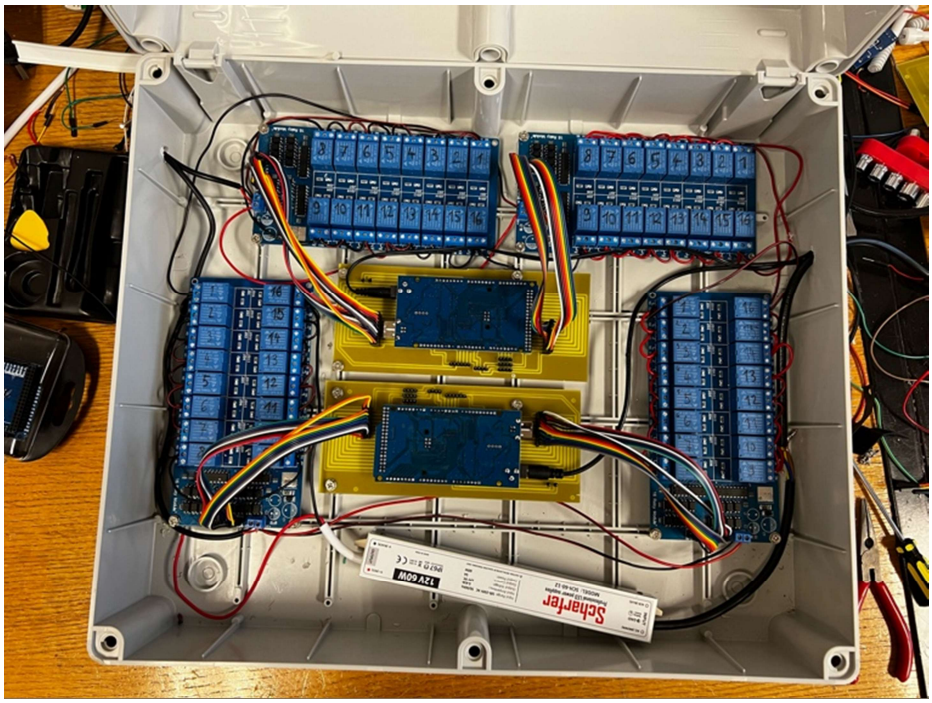


Slika 15: PCB plošča z povezanim mikrokontrolerjem

4.7 POVEZOVANJE KOMPONENT

Ko smo preizkusili vse kontakte na naši PCB plošči, smo povezali vse komponente za testiranje. Pri tem koraku smo odpravili še končne napake, in testirali funkcionalnost sistema. Končno postavitev smo dali v škatlo, ki bo na omaricah, kjer ne bo na dosegu uporabnikom, ampak na voljo za popravila in nadgradnje.

Končna postavitev je imela v škatli povezane mikrokontroler, relaye, in napajanje za te komponente. Nato smo še povezali ključavnice z relayi in je bila naša vezava dokončana.



Slika 16: Končna vezava komponent

5 REZULTATI

Končni rezultat so bile omarice, na voljo dijakom za uporabo. Dijaki lahko prosto uporabljajo omarice kadar jih potrebujejo. Dijak samo skenira svojo kartico in se mu odpre določena omarica. Izpiše se mu tudi številka omarice, da jo lahko lažje najde. Ko jo rabi odpreti, samo spet skenira kartico in se mu ta odpre. Naš končni sistem je preprost in lahek za uporabo.

V prilogi je prikazana demonstracija delovanja omaric.

Video

Tako izgleda naš končen izdelek:

5.1 TEŽAVE

Naš trenutni sistem ima še majhne napake. Najpogostejša, ki se nam je pojavljala, je bil problem, ko se nam so omarice zatikale pri odpiranju. Ta problem smo popravili tako, da smo spremenili mesto določenih ključavnic, da se te niso zatikale.

Drug problem je bil, če se je dijak zaklenil ven iz svoje omarice. V tem primeru smo ročno odlenili omarico in pojasnili dijaku kako uporabljati sistem. Da se ta problem ne bi ponavljal, smo še dodatno natisnili navodila za uporabo, da ne bi prihajalo do uporabniških napak.

6 RAZPRAVA

Kot smo predvidevali, so naši rezultati potrdili naše hipoteze.

1. hipoteza

S pomočjo mikrokontrolerja Arduino, lahko upravljamo veliko količino omaric.

Čeprav smo imeli probleme z količino omaric, smo vseeno uspešno integrirali mikrokontrolerje arduino v naš sistem. Ravno ta preprost sistem nam je omogočil veliko količino upravljanja omaric. Ker nam je ta rešitev uspela, lahko to hipotezo potrdimo,

2. hipoteza

Omarice bodo delovale z dijaškimi karticami za identifikacijo.

Integracija dijaških kartic za identifikacijo v sistem upravljanja omaric lahko prinese številne prednosti in možnosti za izboljšanje učinkovitosti ter nadzora nad dostopom do omaric in njihovo vsebino.

Povečana varnost: Z uporabo dijaških kartic za identifikacijo lahko omejimo dostop do omaric le na pooblaščene uporabnike, kar zmanjšuje možnosti kraje ali nepooblaščenega dostopa do vsebine omaric. Poleg tega omogoča sledenje uporabe omaric, kar lahko pomaga pri ugotavljanju morebitnih nepravilnosti ali zlorab.

Preprostost in udobje: Namesto uporabe klasičnih ključev ali gesel, ki jih je mogoče izgubiti ali pozabiti, so dijaške kartice za identifikacijo priročen in preprost način preverjanja identitete uporabnikov. To povečuje udobje in olajša uporabo sistema za uporabnike.

Povezava z obstoječimi sistemi: Ker se dijaške kartice pogosto že uporabljajo za druge namene, kot so identifikacija pri vstopu v šolo ali knjižnico, njihova integracija v sistem upravljanja omaric omogoča sinergijo med različnimi funkcijami in sistemskimi rešitvami v šolskem okolju.

Glede na podane argumente, lahko potrdimo tudi to hipotezo.

3. hipoteza

Omarice se varne iz programskega in fizičnega vidika.

Omarice so varne iz programskega in fizičnega vidika, kar zagotavlja zaščito vsebine pred nepooblaščenim dostopom ter fizičnimi poškodbami ali krajo. Ta dvojna varnostna zaščita omogoča visoko stopnjo zaupanja in zanesljivosti sistema upravljanja omaric.

Varovanje programskega vidika: S pravilno zasnovanim programskim okoljem in uporabo naprednih varnostnih protokolov je mogoče zagotoviti, da so podatki, povezani z uporabo omaric, ustrezno zaščiteni pred nepooblaščenim dostopom ali vdori v sistem. To vključuje uporabo močnih avtentikacijskih mehanizmov, šifriranja podatkov ter redno posodabljanje programske opreme za odpravljanje morebitnih varnostnih ranljivosti.

Fizična varnost: Poleg zaščite na programski ravni je pomembno tudi fizično varovanje omaric pred nepooblaščenim dostopom, vandalizmom ali krajo. Naše ključavnice fizlično držijo kljuko, tako da je fizični vdor zelo otežen.

S tem lahko potrdimo tudi to hipotezo.

4. hipoteza

V primeru napak, so težave hitro popravljive.

V primeru napak, hitra in učinkovita odprava težav je ključnega pomena za zagotavljanje nemotenega delovanja sistema upravljanja omaric ter preprečevanje morebitnih motenj ali zastojev v uporabi. Zato je potrebno sistematično načrtovanje ter izvajanje postopkov za odpravljanje napak, ki omogočajo takojšnje ukrepanje in minimalno motenje delovanja sistema.

Te hipoteze nismo morali ne potrditi ne zavreči, ker se nam niso pojavile ogromne napake, kjer bi potrebovali menjati ključne komponente sistema.

7 REZULTATI

Naša raziskava in razvoj pametnih omaric je prinesel več pozitivnih rezultatov. Sistem upravljanja z mikrokontrolerjem Arduino se je izkazal za izjemno zanesljivega in prilagodljivega, medtem ko je integracija dijaških kartic olajšala uporabo omaric ter povečala stopnjo varnosti. Dodatno smo razvili in implementirali tudi mehanizme dvojne varnostne zaščite omaric, kar je prispevalo k celoviti zaščiti pred morebitnimi zlorabami.

8 ZAKLJUČEK

Razvoj in implementacija pametnih omaric za šolske dijake predstavljata inovativno rešitev, ki prinaša številne koristi za izobraževalno okolje. S tem projektom smo uspešno dokazali, da je s pravilnim pristopom ter uporabo sodobne tehnologije mogoče bistveno

izboljšati šolsko izkušnjo dijakov, obenem pa spodbujati njihovo zanimanje za računalništvo in tehnologijo, kar je ključno za prihodnost izobraževalnega sistema.

9 ZAHVALA

Najlepša hvala mentorjema, Samu Železniku in Urošu Remenihu, za njuno neprecenljivo podporo, strokovno usmerjanje ter nesebično deljenje znanja in izkušenj med izvajanjem raziskovalne naloge. Njuna predanost in mentorstvo sta bila ključna pri uspehu tega projekta. Hvaležni smo za vso njuno pomoč in vzpodbudo ter za njuno zavzetost pri vodenju naše raziskovalne poti.

Miha Šafranko, Blaž Herman, Tin Keserin

10 PRILOGE

Prikaz delovanja omaric: Link - https://cdn.discordapp.com/attachments/797882228771651628/1227905729101430885/IMG_2272.MOV?ex=662a1b53&is=6617a653&hm=bf9624e990d296f12123b4b3a8950cffbc88a7eca075c095fa0b0f4476642dab&

```
#include <Wiegand.h>
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
WIEGAND wg;

// Velikost zaslona
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

// Določitev pin-ov za zaslon:
#define OLED_CLK   4 // SCL
#define OLED_RESET 5 // RES
#define OLED_MOSI   6 // SDA
#define OLED_DC    7 // DC
#define OLED_CS    0 // ni važno, sam more bit
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT,
OLED_MOSI, OLED_CLK, OLED_DC, OLED_RESET, OLED_CS);

//Določitev relayov in čas sprostitve
int relays[36]; //
bool relayActive[36];
unsigned long relayActivationTime[36];
unsigned long displayTurnOffTime = 0;
const unsigned long displayTurnOffDelay = 5000;
```

Slika 17: Definicije priključkov v kodi

```
// Določanje relayov digitalnim pinom

relays[1] = 23;
pinMode(relays[1], OUTPUT);
digitalWrite(relays[1], HIGH);
relayActive[1] = false;
```

Slika 18: Primer določanja priklopa relayu in drugih definicij

```
void loop() {
  long skeniranaKartica;
  //Preverim, če je bila kartica skenirana
  if (wg.available()) {
    Skenirana = false;
    Serial.print("ID kartice = "); // Izpišem ID kartice
    skeniranaKartica = wg.getCode();
    Serial.println(skeniranaKartica);

    OdstraniOmarico(skeniranaKartica); //Najprej oddstranim kartico, če je že v arrayu

    if (cardCount == 36) { // Če ni bila kartica v arrayu, in je array poln, to izpišem
      DisplayFull();
    }

    if(!Skenirana){ // Če array ni poln in kartica ni v arrayu, jo dodam
      DodajOmarico(skeniranaKartica);
    }

    PrintArray(); // Izpišem cel array
  }

  // Preverim če je že minilo 5 sec, da ugasnem zaslon
  if (millis() >= displayTurnOffTime) {
    display.clearDisplay(); // Clear the display if not FULL and timeout
  }

  // Preverim če je že minilo 5 sec, da ugasnem relay
  for (int x = 1; x < stKartic; x++) {
    if (relayActive[x] && millis() - relayActivationTime[x] >= 500) {
      digitalWrite(relays[x], HIGH);
      relayActive[x] = false;
    }
  }

  display.display(); //Osvežim zaslon
}
```

Slika 19: Glavna funkcija sistema

```
// Funkcija za dodajanje kartice v array, in odklepanje omarice
void DodajOmarico(long skeniranaKartica){
    for (int x = 1; x < stKartic; x++) {
        if (kartice[x] == 0) {
          kartice[x] = skeniranaKartica;
          cardCount++;
          Serial.print("Dal kartico v omarico: ");
          Serial.println(x);
          digitalWrite(relays[x], LOW);
          relayActivationTime[x] = millis();
          relayActive[x] = true;
          Skenirana = true;
          displayTurnOffTime = millis() + displayTurnOffDelay;
          display.clearDisplay();
          if(x  >= 10){
          display.setTextSize(6); // Prilagodite velikost besedila po potrebi
          display.setTextColor(SSD1306_WHITE);
          display.setCursor(15,0);
          display.println(x);
          display.display();
          }
          else{
          display.setTextSize(8); // Prilagodite velikost besedila po potrebi
          display.setTextColor(SSD1306_WHITE);
          display.setCursor(40,0);
          display.println(x);
          display.display();
          }

          Serial.println("Uspešno skenirano. Vpisal v array.");
          break;
        }
      }
}
```

Slika 20: Funkcija, ko uporabnik prvič skenira kartico

```
// Funkcija za odstranitev kartice iz array-a, in odklepanje omarice
void OdstraniOmarico(long skeniranaKartica){
  for (int x = 1; x < stKartic; x++) {
    if (kartice[x] == skeniranaKartica) {
      Serial.print("Našel kartico za omarico: ");
      Serial.println(x);
      digitalWrite(relays[x], LOW);
      relayActivationTime[x] = millis();
      relayActive[x] = true;

      Skenirana = true;
      displayTurnOffTime = millis() + displayTurnOffDelay;
      display.clearDisplay();
      if(x >= 10){
          display.setTextSize(6); // Prilagodite velikost besedila po potrebi
          display.setTextColor(SSD1306_WHITE);
          display.setCursor(18,2);
          display.println(x);
          display.display();
          }
          else{
          display.setTextSize(8); // Prilagodite velikost besedila po potrebi
          display.setTextColor(SSD1306_WHITE);
          display.setCursor(40,0);
          display.println(x);
          display.display();
          }
      kartice[x] = 0; // Reset the card in the array
      cardCount--; // Decrement the card count
      break;
}

  }
```

Slika 21: Funkcija, ko uporabnik drugič skenira kartico

```
// Funckija za izpis na zaslonu, če je array poln.
void DisplayFull(){
  display.clearDisplay();
  display.setTextSize(3);
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(0, 0);
  display.println(F("Vse omarice so zasedene!"));
  display.display();
  Serial.println("Polno!");
  displayTurnOffTime = millis() + displayTurnOffDelay;
}
```

Slika 22: Izpis uporabniku, če so vse omarice zasedene