Srednja šola za kemijo, elektrotehniko in računalništvo

Pot na Lavo 22 3000 Celje

Studijske RGB luči

raziskovalna naloga

Avtorji: Anže Glavač, Žan Gradišnik, Lan Firer

Mentor: Andraž Pušnik

Mestna občina Celje, Mladi za Celje
Celje, 2023

Mentor Andraž Pušnik v skladu z 20. členom Pravilnika o organizaciji mladinske raziskovalne dejavnosti »Mladi za Celje« Mestne občine Celje, zagotavljam, da je v raziskovalni nalogi z naslovom Studijske RGB luči, katere avtorji so Lan Firer, Anže Glavač in Žan Gradišnik:

- besedilo v tiskani in elektronski obliki istovetno,
- pri raziskovanju uporabljeno gradivo navedeno v seznamu uporabljene literature,
- da je za objavo fotografij v nalogi pridobljeno avtorjevo dovoljenje in je hranjeno v šolskem arhivu,
- da sme Osrednja knjižnica Celje objaviti raziskovalno nalogo v polnem besedilu na knjižničnih portalih z navedbo, da je raziskovalna naloga nastala v okviru projekta Mladi za Celje,
- da je raziskovalno nalogo dovoljeno uporabiti za izobraževalne in raziskovalne namene s povzemanjem misli, idej, konceptov oziroma besedil iz naloge ob upoštevanju avtorstva in korektnem citiranju,
- da smo seznanjeni z razpisni pogoji projekta Mladi za Celje.

Celje, 7. 4. 2023

žig šole

ŠOLSKI CENTER CELJE
ŠOLA ZA KEMIJO, ELEKTROTEHNIKO IN RAČUNALNIŠTVO
SREDNJA ŠOLA 1
CELJE, Pot na Lavo 22

Podpis mentorja

Podpis odgovorne osebe

Zahvala

Za strokovno pomoč, nasvete in predloge se zahvaljujemo gospodu Andražu Pušniku, ki nas je s svojimi napotki usmerjal in tako pomagal pri nastajanju raziskovalne naloge.

Povzetek

V naši raziskovalni nalogi smo preučevali uporabo RGB luči pri fotografiranju in načine, kako lahko preko mikrokrmilnika ESP32 krmilimo osvetlitev. Predstavili smo različne načine za komunikacijo med mikrokrmilnikom in RGB lučmi ter podrobno opisali, kako deluje komunikacija med njima. Raziskali smo tudi uporabo led luči in mikrokrmilnika ESP32 ter opisali, kako ju lahko uporabimo za krmiljenje RGB osvetlitve pri fotografiranju. Skozi raziskavo smo pridobili dragocene informacije o uporabi RGB luči in mikrokrmilnikov ESP32, ki bodo v pomoč pri nadaljnjih projektih s področja fotografije in avtomatizacije.

V praktičnem delu naše raziskovalne naloge smo izdelali letvico, na kateri smo namestili RGB luči. Krmiljenje luči smo izvedli preko mikrokrmilnika ESP32, ki smo ga povezali z omrežjem preko WebSocket protokola. Opisali smo postopek izdelave letvice in povezavo mikrokrmilnika z omrežjem ter podrobno opisali delovanje krmiljenja RGB luči preko ESP32. Naše praktično delo nam omogoča vpogled v možnosti uporabe RGB luči in mikrokrmilnikov ESP32 pri fotografiranju in drugih področjih, kjer je potrebno avtomatizirati proces krmiljenja osvetlitve.

Ključne besede: RGB, WebSocket, JSON, ESP32, HTTP, access point

Abstract

In our research project, we studied the use of RGB lights in photography and the ways in which we can control the lighting via the ESP32 microcontroller. We presented different ways to communicate between the microcontroller and RGB lights and described in detail how the communication between them works. We also explored the use of LED lights and the ESP32 microcontroller and described how they can be used to control RGB lighting in photography. Through the research, we obtained valuable information about the use of RGB lights and ESP32 microcontrollers, which will be helpful in further projects in the field of photography and automation.

In the practical part of our research task, we made a strip on which we installed RGB lights. We controlled the lights via an ESP32 microcontroller, which we connected to the network via the WebSocket protocol. We described the process of making the strip and the connection of the microcontroller to the network and described in detail the operation of the RGB light control via ESP32. Our practical work gives us insight into the possibilities of using RGB lights and ESP32 microcontrollers in photography and other areas where it is necessary to automate the lighting control process.

Keywords: RGB, WebSocket, JSON, ESP32, HTTP, access point

Vsebina

Kazalo slik

Kazalo grafov

1 Uvod

V svetu se vedno bolj uveljavljajo različne tehnologije, ki omogočajo enostavnejšo in učinkovitejšo upravljanje različnih naprav. Ena takih tehnologij je brezžično krmiljenje naprav preko mobilnega telefona. V zadnjem času se tudi v domačih in profesionalnih okoljih vse bolj uporabljajo LED svetila, zato smo se odločili, da v okviru te raziskovalne naloge preučimo možnost krmiljenja RGB LED svetil preko mobilnega telefona za studijsko uporabo pri fotografiranju.

ESP32 je mikrokrmilnik, ki je namenjen za uporabo v različnih aplikacijah na področju internetne povezljivosti in avtomatizacije. Gre za naprednejšo različico mikrokrmilnika ESP8266, ki je priljubljena izbira pri razvoju IoT naprav. ESP32 je opremljen z dvojedrnim procesorjem, ki omogoča hitro in učinkovito obdelavo podatkov, ter z vgrajenim WiFi in Bluetooth modulom za brezžično povezljivost. Poleg tega omogoča tudi uporabo drugih protokolov in vmesnikov, kot so na primer SPI, I2C, UART, PWM in ADC, kar mu omogoča širok nabor uporabe.

Pri izdelavi RGB luči, ki jih krmili ESP32, smo se srečali s kompleksnostjo in zahtevnostjo projekta. Bili smo v dilemi, ali bomo lahko uspešno izvedli nalogo, saj smo se spraševali, ali imamo dovolj znanja in izkušenj in ali bo mogoče izvesti projekt z minimalnimi kapitalskimi sredstvi. Kljub vsem izzivom smo se odločili, da se bomo lotili naloge, saj nas je vodila strast do programiranja, elektronike in ustvarjanja. S tem projektom smo pridobili nove izkušnje in se naučili reševati različne probleme, ki so se pojavili med izdelavo.

1.1 Hipoteze

1. Predvidevamo, da bomo imeli težave predvsem z utripanjem LED diod pri zajemanju s kamero.

   Zaradi razlike med hitrostjo zaslonke in osveževanja naših LED luči, lahko pride do “flicker” efekta.

2. Mikrokrmilnik ESP32 bo zadostil osnovnim funkcijam, pri naprednih zahtevah pa ne bo dovolj zmogljiv.

   Skrbelo nas je, česa vse je zmožen krmilnik in kakšna komunikacija lahko poteka.

3. Končni izdelek načrtujemo izdelati za manj kot 50 eur.

1.2 Cilji naloge

Cilj je bil izdelati barvne luči za fotografiranje, ki bi bile bolj dostopne kot profesionalne luči in jih je mogoče upravljati preko telefona. Poleg tega smo želeli zagotoviti, da so te luči hitrejše in enostavnejše za uporabo kot luči, ki jih je treba upravljati fizično. S tem smo si prizadevali zagotoviti, da bodo ljudje imeli dostop do kakovostnih luči za fotografiranje, ki bodo stale največ 100€ ter omogočiti večjo fleksibilnost in nadzor nad njihovo uporabo. Zadali smo si časovno obdobje, da bomo izdelek naredili v roku treh mesecev.

1.3 SWOT analiza

Ustvarili smo SWOT analizo, ki nam pomaga, da bolje razumemo, kje so prednosti našega izdelka kje obstajajo pomanjkljivosti ter kakšne priložnosti in grožnje obstajajo na trgu:

Moči:

- Cenovno ugoden izdelek
- Tehnološko napreden izdelek
- Enostaven za uporabo

Šibkosti:

- Občutljivost LED traku, kar lahko povzroči hitro okvaro izdelka
- Visoka poraba energije, ki vodi do dodatnih stroškov napajanja

Priložnosti:

- Izboljšanje izdelka z opcijo analognega upravljanja
- Baterijsko napajanje
- Zamenjava LED traku z boljšo verzijo

Grožnje:

- Velika konkurenca s strani velikih podjetij z ogromnim kapitalom

2 Postopek raziskovanja

2.1 Področje in problem

Razsvetljava je ena najpomembnejših sestavin pri fotografiranju, saj ima velik vpliv na kakovost in estetiko fotografije. Pravilna uporaba različnih svetlobnih virov lahko ustvari dramatično različne učinke in občutke v fotografiji.

Barvne luči igrajo posebno vlogo pri fotografiranju, saj lahko dodajo dramatične učinke in ustvarijo različna razpoloženja v fotografiji. Barvne luči se lahko uporabljajo za poudarjanje ali poudarjanje določenih barv ali tonov v fotografiji. Na primer uporaba rdeče barvne luči lahko ustvari občutek toplote in strasti, medtem ko lahko modra barvna luč ustvari občutek hladnosti in miru.

Problem naše naloge je bil, kljub ponudbi barvnih luči na trgu, ki so namenjene bolj profesionalnim uporabnikom in so zato dražje na trgu ni podobnih luči, ki bi bile ugodnejše in bi bile dostopne uporabnikom. S tem smo hoteli ponuditi ugodnejše luči, ki bi bile dostopne večini začetnikom in domačim uporabnikom.

2.2 Metodologija

Z raziskovanjem smo začeli tako, da smo preučili načine, kako bi lahko izvedli našo idejo.

Sledila je raziskava konkurence na trgu. Ugotavljali smo, kako lahko za pretirano nižjo ceno ustvarimo izdelek, ki se bo lahko primerjal z izdelki naših konkurentov. Iskali smo načine, kje smo lahko boljši od konkurentov in kako lahko zmanjšamo stroške izdelave.

Nato smo preučili potrebne komponente, kako delujejo in kako med seboj funkcionirajo. Namenili smo precej časa preučevanju njihovega delovanja, da smo lahko dobro poznali komponente, kar nam je dalo veliko prednost pri ugotavljanju problemov, ki smo jih imeli. Po preučevanju smo spoznali vse slabosti in prednosti komponent ter na podlagi tega izbrali in se odločili, katere komponente bi bile najboljše za naš projekt. Po vseh zbranih podatkih smo postavili hipoteze za našo raziskovalno nalogo.

Poleg tega pa smo povprašali fotografe, če bi bili zainteresirani za uporabo naše luči in kakšne lastnosti bi potrebovali v takšnem izdelku.

3 ESP32

ESP32 je mikrokrmilnik, ki ga je razvilo podjetje Espressif Systems. Prvič je bil izdan leta 2016 in je naslednik njihovega priljubljenega modela ESP8266. ESP32 je bil razvit kot bolj zmogljiva različica svojega predhodnika in ponuja več funkcionalnosti ter večjo zmogljivost. (Espressif, Espressif, 2023)

ESP32 temelji na arhitekturi Xtensa LX6 in ima dva 32-bitna procesorja, ki delujeta pri frekvenci do 240 MHz. Vgrajen ima tudi Wi-Fi in Bluetooth povezljivost, kar mu omogoča brezžično povezavo s številnimi napravami in omrežji. Poleg tega ima tudi številne druge vgrajene funkcije, kot so senzorji, PWM izhodi, I2C, SPI, UART in še več.

Je majhen in zmogljiv mikrokrmilnik, ki se uporablja za izvajanje različnih nalog na različnih napravah, kot so mikrokontrolerji, pametni telefoni, pametne ure, senzorji, IoT naprave in drugi sistemi.

ESP32 ima vgrajeno Wi-Fi in Bluetooth povezljivost, kar ga naredi primernega za uporabo v napravah, ki potrebujejo brezžično povezavo s svetovnim spletom in drugimi napravami. Podpira tudi druge brezžične protokole, kot so LoRa, Sigfox in Zigbee, kar povečuje njegovo uporabnost za širok spekter IoT aplikacij.

ESP32 ima veliko pomnilnika, ki omogoča shranjevanje velike količine podatkov in programov. Poleg tega ima tudi številne vmesnike, kot so SPI, I2C, UART in CAN, ki omogočajo povezavo z drugimi napravami in senzorji.

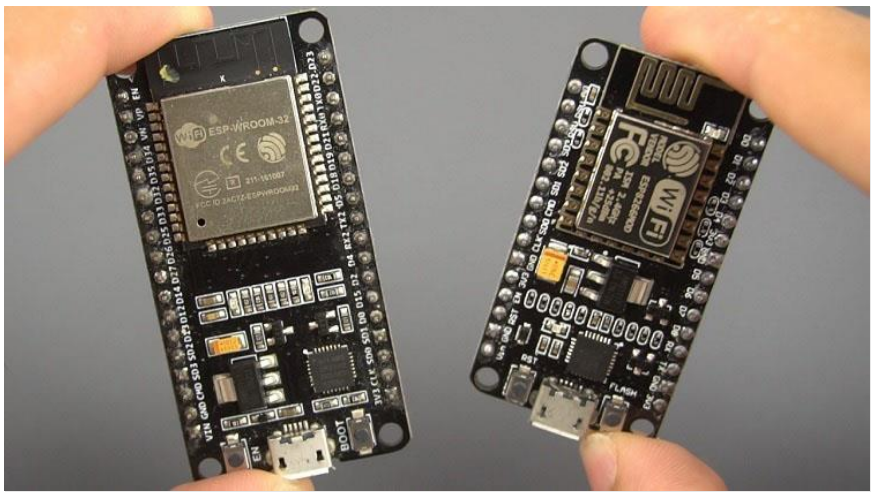
Zaradi svoje majhnosti, nizke porabe energije in močnih zmogljivosti je ESP32 postal zelo priljubljen med razvijalci, ki ga uporabljajo za izdelavo različnih naprav, kot so roboti, brezžične senzorske mreže, pametni domovi in drugi pametni sistemi. Poleg tega pa je

njegova programska oprema odprtokodna, kar omogoča razvijalcem, da prilagodijo in razširijo funkcionalnost tega mikrokrmilnika.

Poleg tega, da ima ESP32 številne vgrajene funkcije, je na voljo tudi bogat nabor orodij za razvoj programske opreme. Na voljo so številni programski jeziki, vključno s C in C++, Python, JavaScript in drugimi, ter različna orodja za razvoj, kot so Arduino IDE, ESP-IDF in drugi. To omogoča razvijalcem, da izberejo najbolj primerno orodje za svoje potrebe in da hitro in učinkovito razvijejo kodo.

ESP32 ima tudi veliko skupnost razvijalcev, ki delijo svoje izkušnje, nasvete in primere uporabe. To omogoča lažje učenje in razvoj z uporabo tega mikrokrmilnika. Poleg tega pa obstaja tudi veliko knjižnic in razširitev, ki razvijalcem omogočajo hitro in enostavno integracijo z drugimi napravami in senzorji.

Nekatere od prednosti, ki jih ima ESP32 v primerjavi z drugimi mikrokrmilniki, so tudi nizka cena, enostavnost uporabe in izjemno nizka poraba energije. To omogoča uporabo ESP32 tudi v napravah, ki so odvisne od baterij in imajo omejene vire energije.



Slika 1 - ESP32

Vir: https://randomnerdtutorials.com/getting-started-with-esp32/

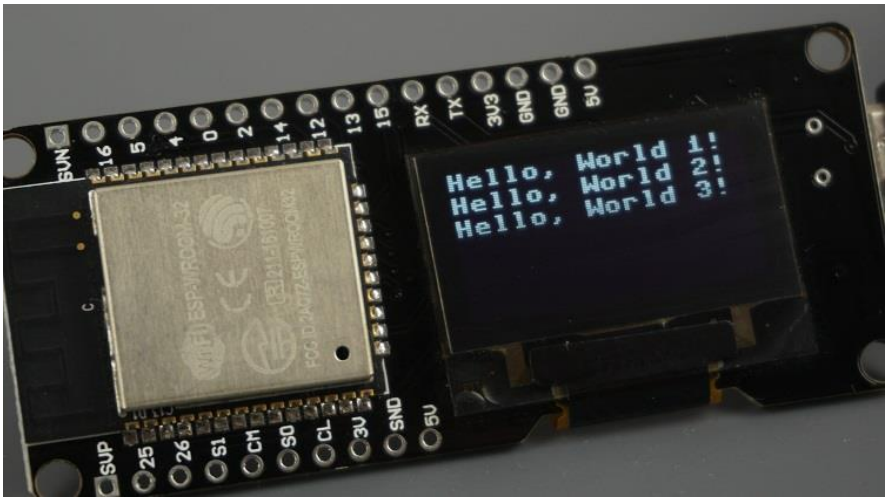
3.1 ESP32 OLED

ESP32 OLED je razvojni komplet, ki vključuje mikrokrmilnik ESP32 in OLED zaslon. ESP32 je zmogljiv mikrokrmilnik, ki se pogosto uporablja za različne projekte na področju IoT (Internet of Things). OLED zaslon, ki je vgrajen v ta razvojni komplet, pa je majhen, nizkoenergijski zaslon, ki ima visoko ločljivost in dobro vidljivost tudi v svetlih prostorih.

Razvojni komplet ESP32 OLED je odlična izbira za različne projekte, ki zahtevajo brezžično povezljivost in vizualno prikazovanje podatkov. Mikrokrmilnik ESP32 omogoča povezovanje z internetom prek Wi-Fi ali Bluetooth povezave, z OLED zaslonom pa lahko uporabnik vizualno prikaže podatke, kot so senzorski podatki, sporočila, grafikoni in drugo.

OLED zaslon v razvojnem kompletu ESP32 OLED je 0,96-palčni zaslon z ločljivostjo 128x64 pik. Zaslon ima samosvetlečo tehnologijo, kar pomeni, da ni potrebno odzadnje osvetljevanje in ima zato nižjo porabo energije. Zaslon ima tudi visok kontrast in dobro vidljivost v širokem spektru kotov gledanja.

Razvojni komplet ESP32 OLED je odprtokodna platforma, kar pomeni, da ima uporabnik dostop do vira kode in lahko programira mikrokrmilnik po svojih željah. Na voljo je veliko knjižnic in primerov, ki olajšajo uporabo in programiranje različnih funkcij.



Slika 2 - OLED zaslon na ESP32

Vir: https://randomnerdtutorials.com/esp32-built-in-oled-ssd1306/

4 Izbira razvojnega okolja

Za razvoj smo izbrali Arduino IDE, saj smo z njim že seznanjeni in se z njim dobro znajdemo. Poleg tega je Arduino IDE zelo priljubljen med razvijalci, kar pomeni, da je na voljo veliko pomoči in podpore s strani skupnosti. Obstaja tudi ogromno knjižnic, ki so bile napisane za ta IDE, kar zelo olajša in pospeši proces programiranja.

4.1 Arduino IDE

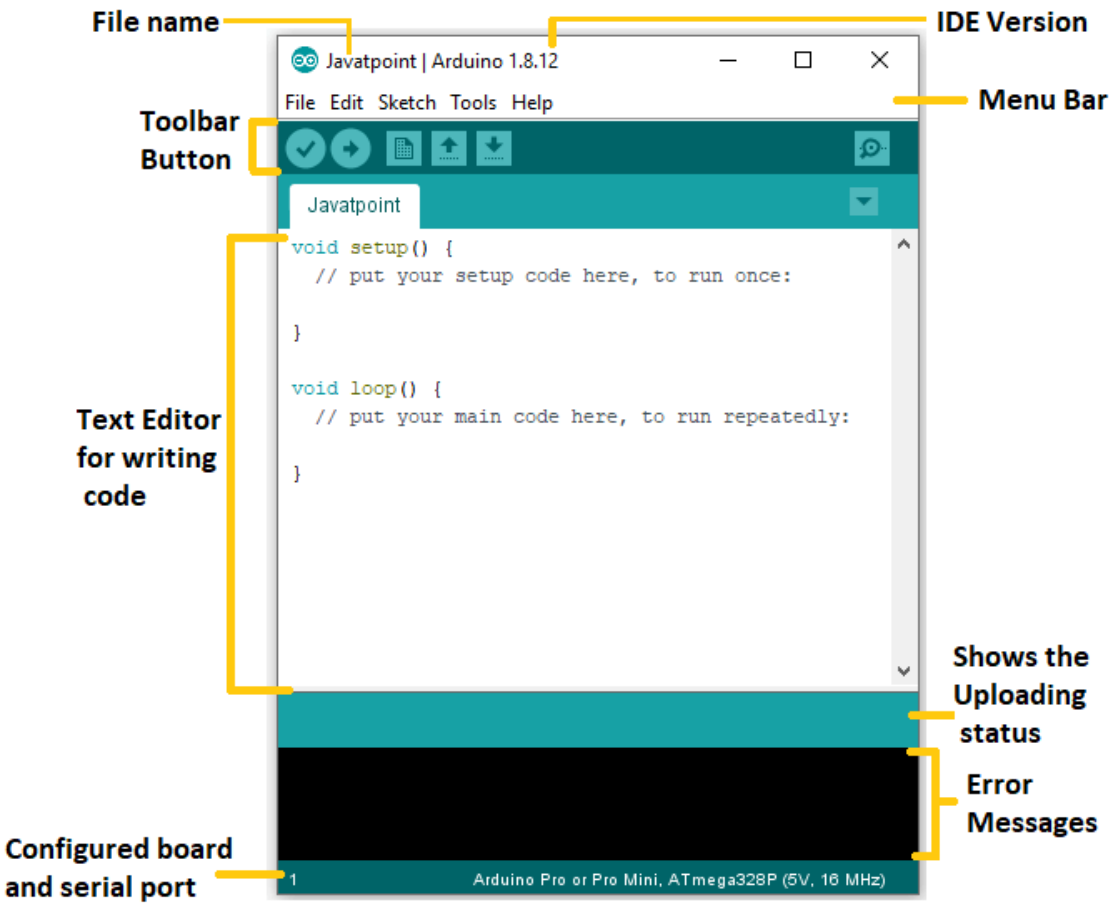
Arduino IDE je razvojno okolje (IDE), ki se uporablja za programiranje mikrokrmilnikov Arduino. Mikrokrmilniki Arduino so majhni računalniški sistemi, ki se lahko uporabljajo za različne namene, kot so nadzor robotov, samodejno zalivanje rastlin, avtomatizacija domačih naprav in še veliko več. Platforma Arduino temelji na odprtokodnem programskem jeziku in različnih komponentah, kot so senzorji, motorji, LED diode itd. (Arduino, 2023)

Arduino IDE je zasnovan tako, da omogoča programerjem, da pišejo, nalagajo in urejajo kodo za mikrokrmilnike Arduino. Koda, napisana v Arduino IDE, je nato prevedena v strojno kodo, ki jo lahko mikrokrmilnik Arduino izvaja.

Arduino IDE je zelo preprost za uporabo in je primeren tudi za tiste, ki nimajo veliko izkušenj s programiranjem. IDE vključuje preprost urejevalnik besedil, ki ga lahko uporabite za pisanje vašega programa. IDE vključuje tudi funkcije, kot so izbira plošče (board), izbor vrste priklopnih točk (port), branje serijskih podatkov, pošiljanje serijskih podatkov in še veliko več.

Arduino IDE omogoča uporabo različnih programskih jezikov, kot so C, C++, Java in Python, in vključuje knjižnice za uporabo različnih komponent, kot so senzorji, motorji, LED diode itd. Programerji lahko uporabijo te knjižnice, da bi skrajšali čas razvoja in poenostavili svoje delo. Poleg tega IDE vključuje tudi orodja za preverjanje in razhroščevanje kode, kar omogoča programerjem, da ugotovijo, ali njihov program deluje pravilno.

Arduino IDE je na voljo za različne operacijske sisteme, kot so Windows, macOS in Linux, in je na voljo za prenos s spletnega mesta Arduino. Poleg tega je IDE odprtokoden, kar pomeni, da ga lahko programerji prilagodijo svojim potrebam in ga uporabljajo za razvoj različnih aplikacij. Arduino IDE je brezplačen in je na voljo vsakomur, ki želi razviti svoje projekte z mikrokrmilniki Arduino.



Slika 3 - Arduino IDE

Vir: https://www.javatpoint.com/arduino-ide

4.2 Knjižnice

Knjižnice (libraries) pri mikrokrmilnikih ESP32 so sklopi kode, ki zagotavljajo že napisane funkcije za uporabo v vaših projektih. Knjižnice lahko vsebujejo funkcije za uporabo različnih komponent, kot so senzorji, motorji, brezžični moduli in še veliko več.

Pri programiranju mikrokrmilnika ESP32 lahko uporabite obstoječe knjižnice ali pa napišete lastne funkcije za uporabo v svojem projektu. Obstajajo tudi knjižnice, ki so na voljo na spletu in se jih lahko prenese ter uporabi v projektu.

Knjižnice lahko v veliki meri olajšajo razvoj vašega projekta, saj vam omogočajo uporabo že napisanih funkcij za uporabo komponent, ki jih želite vključiti v svoj projekt. To lahko zmanjša čas in trud, ki ga porabite za pisanje kode za uporabo teh komponent.

Pri uporabi knjižnic je pomembno, da preverite, ali knjižnica vsebuje funkcije, ki jih potrebujete za vaš projekt, ter ali je knjižnica združljiva z vašim mikrokrmilnikom ESP32. Poleg tega lahko uporaba knjižnic vpliva na velikost vaše kode. Če uporabljate več knjižnic, se lahko velikost vaše kode poveča, kar lahko vpliva na zmogljivost vašega mikrokrmilnika ESP32.

Knjižnice se lahko uporabljajo na različne načine. Ko prenesete knjižnico, jo morate najprej vključiti v vaš projekt. To storite tako, da knjižnico dodate v vaš projekt v Arduino IDE, kar bo omogočilo dostop do funkcij in kod v knjižnici.

Uporaba funkcij v knjižnici je preprosta. Ko vključite knjižnico v vaš projekt, lahko uporabite funkcije, ki jih vsebuje knjižnica, s klicanjem imena funkcije v vašem programu. Na primer, če knjižnica vsebuje funkcijo za vklop LED diode, lahko to storite z enim samim klicem funkcije v vašem programu.

Knjižnice lahko bistveno olajšajo razvoj vašega projekta in vam omogočajo hitrejše ustvarjanje vaših idej. Vendar pa je pomembno, da preverite, ali knjižnica vsebuje funkcije, ki jih potrebujete, in ali bo uporaba knjižnice vplivala na velikost vaše kode in zmogljivost mikrokrmilnika ESP32.

5 Uporabljene knjižnice

Pri našem projektu bomo uporabili več knjižnic, ki bodo omogočale uporabo različnih komponent in funkcij. Med njimi bodo Adafruit NeoPixel, Adafruit_GFX, Adafruit_SSD1306, AsyncTCP, ESPAsyncWebServer, SPIFFS in WiFi.

Adafruit NeoPixel knjižnica omogoča upravljanje z NeoPixel LED trakovi in nudi funkcije za upravljanje s posameznimi LED diodami, vključno s spreminjanjem barve in svetlosti.

Adafruit_GFX knjižnica je grafična knjižnica, ki ponuja funkcije za risanje različnih oblik, besedil in slik na zaslonu.

Adafruit_SSD1306 knjižnica omogoča uporabo OLED zaslona SSD1306 in ponuja funkcije za upravljanje z besedilom in slikami na zaslonu.

AsyncTCP in ESPAsyncWebServer sta knjižnici, ki omogočata uporabo brezžične povezave WiFi za oddajanje in sprejemanje podatkov prek protokolov TCP in HTTP.

SPIFFS knjižnica omogoča uporabo datotečnega sistema SPI Flash, ki lahko shranjuje podatke in konfiguracijske datoteke

Knjižnica WiFi omogoča uporabo brezžične povezave WiFi za povezovanje z internetom in uporabo spletnih storitev.

Uporaba teh knjižnic nam bo omogočila enostavno uporabo različnih komponent in funkcij v našem projektu.

5.1 Adafruit NeoPixel

Knjižnica Adafruit NeoPixel ponuja različne funkcije, ki omogočajo upravljanje z NeoPixel LED trakovi. Nekatere pomembne funkcije so:

1. begin(): Ta funkcija inicializira NeoPixel trak in omogoči komunikacijo med mikrokrmilnikom in trakom.
2. show(): Funkcija posodobi stanje vseh LED diod v traku glede na trenutne nastavitve barve in svetlosti.
3. setPixelColor(): S to funkcijo lahko nastavimo barvo in svetlost posamezne LED diode v traku. Funkcija sprejme dva argumenta − indeks diode in RGB vrednost.
4. fill(): Ta funkcija nastavi barvo in svetlost vseh LED diod v traku na podano vrednost.
5. clear(): S to funkcijo lahko izklopite vse LED diode v traku in s tem ustvarite črn zaslon.
6. setBrightness(): S to funkcijo lahko nastavimo svetlost celotnega traku LED diod.
7. numPixels(): Ta funkcija vrne število LED diod v traku.
8. getPixelColor(): S to funkcijo lahko pridobimo trenutno nastavljeno barvo in svetlost posamezne LED diode v traku.
9. gamma32(): Ta funkcija se uporablja za izravnavo barvne krivulje in izboljšanje kakovosti barv na LED diodah.

Te funkcije omogočajo enostavno upravljanje z LED trakovi in prilagajanje barve in svetlosti posameznih LED diod v traku.

5.2 Adafruit_SSD1306

Knjižnica Adafruit_SSD1306 je grafična knjižnica, ki omogoča upravljanje OLED zaslona SSD1306. Nekatere pomembne funkcije knjižnice so:

1. begin(): Ta funkcija inicializira OLED zaslon in omogoči komunikacijo med mikrokrmilnikom in zaslonom.

2. display(): Funkcija posodobi stanje zaslona glede na trenutne nastavitve. To pomeni, da se vsi podatki, ki so bili izrisani na zaslon, dejansko izpišejo na zaslonu.

3. clearDisplay(): S to funkcijo lahko izbrišete celoten zaslon in s tem ustvarite črn zaslon.

4. drawPixel(): S to funkcijo lahko izrisujete posamezne pike na zaslonu z določeno barvo in svetlostjo. Funkcija sprejme tri argumente − koordinati x in y ter barvo pike.

5. setTextSize(): S to funkcijo lahko nastavite velikost besedila, ki ga boste nato izpisali na zaslon.

6. setTextColor(): S to funkcijo lahko nastavite barvo besedila, ki ga boste nato izpisali na zaslon.

7. setCursor(): S to funkcijo lahko nastavite položaj kazalca besedila na zaslonu, kamor boste nato izpisali besedilo.

8. print(): S to funkcijo lahko izpišete besedilo na zaslon v trenutno nastavljeni barvi, velikosti in položaju.

Te funkcije omogočajo enostavno upravljanje z OLED zaslonom in izrisovanje različnih grafičnih elementov, kot so besedilo, pike in slike. Z uporabo teh funkcij lahko ustvarimo različne vizualne elemente na zaslonu, kot so gumbi, grafikoni in indikatorji.

6 ESPAsyncWebServer

ESPAsyncWebServer je knjižnica, ki omogoča enostavno ustvarjanje spletnih strežnikov na platformi ESP32. Ta knjižnica temelji na knjižnici AsyncWebServer in dodaja podporo za asinhrono HTTP zahtevke na platformi ESP32. To pomeni, da se lahko naš ESP32 odzove na HTTP zahtevke z minimalnim vplivom na delovanje naprave.

ESPAsyncWebServer podpira tako HTTP kot tudi HTTPS protokole in zagotavlja veliko funkcij za enostavno upravljanje in ustvarjanje spletnih vmesnikov. Nekatere od teh funkcij vključujejo:

Podpora za asinhrono obdelavo HTTP zahtevkov, ki omogoča hkratno obsluževanje več uporabnikov in boljšo odzivnost naprave.

- Podpora za HTTP in HTTPS protokole.
- Podpora za avtentikacijo uporabnikov s standardno HTTP Basic Auth avtentikacijo.
- Podpora za statično spletno vsebino, kot so slike, CSS datoteke in podobno.
- Podpora za streženje dinamične vsebine, kot so HTML datoteke, ki se ustvarjajo na podlagi podatkov, ki jih prejme vaš ESP32.
- Podpora za samodejno upravljanje pomnilnika za spletni strežnik, tako da se lahko izognete težavam s pomnilnikom, povezanimi s preveliko uporabo pomnilnika.
- Podpora za branje in pisanje podatkov v pomnilniku SPIFFS, kar omogoča hiter in enostaven dostop do vaših datotek.

ESPAsyncWebServer je zelo priljubljena knjižnica med razvijalci in uporabniki ESP32, saj omogoča enostavno upravljanje in ustvarjanje spletnih vmesnikov na tej platformi. S to knjižnico lahko hitro ustvarite svoj spletni vmesnik za nadzor vaših naprav, kot so senzorji ali motorji, in tako nadzorujete vaše naprave prek spleta ali intraneta.

7 SPIFFS

SPIFFS (Serial Peripheral Interface Flash File System) je majhen datotečni sistem, ki je optimiziran za uporabo s flash pomnilniškimi napravami, kot so naprave na osnovi NOR in NAND flash pomnilnikov. To je knjižnica, ki jo je mogoče uporabiti na platformi ESP32 in drugih podobnih mikrokrmilnikih, ki imajo vgrajeno podporo za SPI (Serial Peripheral Interface) komunikacijski protokol. (Espressif, Espressif, 2023)

SPIFFS omogoča enostavno upravljanje datotek v flash pomnilniških napravah in omogoča shranjevanje datotek na pomnilniški napravi, kot so konfiguracijske datoteke, slike, datoteke z besedilom itd.

8 WIFI manager

WifiManager je knjižnica za arduino, ki je namenjena upravljanju WiFi povezav na napravah, ki temeljijo na mikrokrmilniku ESP32. Namen knjižnice je poenostaviti postopek povezovanja naprave z omrežjem WiFi in omogočiti samodejno preverjanje kakovosti povezave ter ponovno povezovanje ob izgubi signala.

Kot verjetno že veste, je ESP32 mikrokrmilnik, ki je namenjen predvsem za IoT projekte in vključuje tako WiFi kot Bluetooth povezljivost. Vendar pa lahko povezovanje s WiFi omrežjem včasih predstavlja izziv, še posebej če uporabljamo mikrokrmilnik v okoljih, kjer se pogosto spreminja dostopna točka ali signal ni vedno na voljo.

Z uporabo knjižnice WifiManager pa se ta izziv zelo olajša. Knjižnica namreč omogoča uporabniku, da predhodno nastavi ime in geslo WiFi omrežja, s katerim se bo naprava povezala, ter spremlja stanje povezave. V primeru izgube povezave, se naprava avtomatsko ponovno poveže z omrežjem, kar uporabniku prihrani čas in izognemo se morebitnim težavam z ročnim ponovnim povezovanjem naprave.

Poleg tega pa ima WifiManager tudi vgrajeno funkcijo za samodejno prikazovanje konfiguracijskega portala, kar nam omogoča, da spremenimo nastavitve WiFi-ja, tudi če se naprava že nahaja v omrežju. Na ta način se izognemo težavam z ročno spremembo nastavitev v kodi naprave, kar je lahko zamudno in zahtevno.

WifiManager je zelo uporabna knjižnica, s katero bi v primeru, da ni omrežja, bi ESP32 sam poskrbel, da bi se postavil v način dostopne točke **(acces point)** in deloval kot Router.

9 Delovanje ESP32

ESP32 je zmogljiv mikrokrmilnik, ki omogoča povezovanje z brezžičnimi omrežji preko WiFi povezave. Pri WiFi povezovanju ima ESP32 dve glavni možnosti: delovanje kot Access Point (AP) in Station (STA).

9.1 AP dostopna točka

AP ali Access Point pri ESP32 je brezžični način delovanja, ki omogoča, da se ESP32 naprava deluje kot brezžična dostopna točka. To pomeni, da lahko druge naprave, kot so pametni telefoni, tablice, prenosniki itd., brezžično povežejo z ESP32 in uporabljajo njegovo brezžično omrežje.

Uporaba AP pri ESP32 je koristna v primerih, ko ni na voljo dostopa do obstoječega omrežja ali ko želite ustvariti lastno brezžično omrežje. AP na ESP32 se lahko uporablja tudi v primerih, ko želite uporabiti ESP32 kot brezžični usmerjevalnik, ki lahko posreduje povezavo do interneta.

AP na ESP32 deluje tako, da se ESP32 naprava pretvori v brezžično dostopno točko, tako da oddaja brezžični signal, ki ga lahko zaznajo druge naprave v bližini. Ko druge naprave zaznajo signal ESP32, lahko vzpostavijo brezžično povezavo z njim.

Pri uporabi AP na ESP32 lahko uporabnik nastavi različne parametre, kot so ime in geslo dostopne točke, omrežni kanal, omrežno varnost in druge nastavitve, ki jih AP podpira. To omogoča, da lahko uporabnik prilagodi delovanje AP na ESP32 glede na svoje potrebe.

AP na ESP32 deluje kot brezžični usmerjevalnik tako, da omogoča drugim napravam dostop do interneta. To pomeni, da lahko naprave, ki so povezane z AP na ESP32, dostopajo do interneta brezžično.

Za delovanje AP na ESP32 mora biti ESP32 povezan na napajanje, lahko pa ga uporabnik tudi programira za izvajanje določenih nalog. Uporaba AP na ESP32 lahko na primer omogoča, da se naprava uporabi kot prenosni Wi-Fi usmerjevalnik za druge naprave, kot so pametni telefoni ali tablice, kadar ni na voljo drugega dostopa do interneta.

10 STA

STA ali Station pri ESP32 je brezžični način delovanja, ki omogoča, da se ESP32 naprava poveže na obstoječe brezžično omrežje kot uporabnik. To pomeni, da se lahko ESP32 uporablja kot brezžična naprava za povezavo z drugimi brezžičnimi omrežji. (Kashif, 2023)

Uporaba STA na ESP32 je koristna, kadar potrebujete povezavo z obstoječim brezžičnim omrežjem. To lahko vključuje vaš domači Wi-Fi, Wi-Fi v pisarni ali javne brezžične omrežje v kavarni ali hotelu. S STA na ESP32 lahko vzpostavite brezžično povezavo z obstoječim omrežjem in dostopate do interneta.

STA na ESP32 deluje tako, da se ESP32 naprava poveže z obstoječim brezžičnim omrežjem in prejme IP naslov, ki omogoča dostop do interneta. Ko je ESP32 povezan z obstoječim omrežjem, lahko naprave, ki so povezane z ESP32, dostopajo do interneta preko tega omrežja.

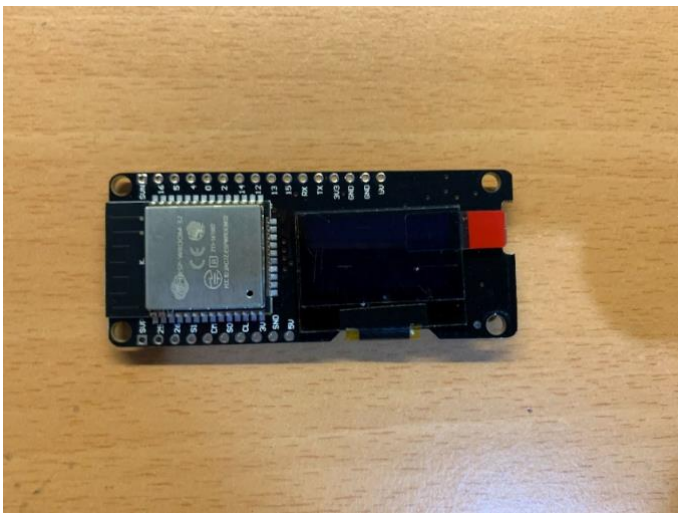
Pri uporabi STA na ESP32 lahko uporabnik nastavi različne parametre, kot so ime in geslo brezžičnega omrežja, vrsta varnosti, omrežni kanal itd. To omogoča, da lahko uporabnik prilagodi delovanje STA na ESP32 glede na potrebe omrežja, ki ga želijo povezati.

STA na ESP32 omogoča tudi hitro in preprosto povezavo z drugimi brezžičnimi napravami, kot so pametni telefoni, tablice, prenosni računalniki itd., ki so že povezane z obstoječim brezžičnim omrežjem.

V primerjavi z drugimi načini delovanja ESP32 uporaba STA omogoča uporabnikom, da hitro in enostavno vzpostavijo povezavo z obstoječimi brezžičnimi omrežji. To omogoča tudi, da se naprave, ki so povezane z ESP32 preko STA, povežejo z drugimi napravami, ki so na istem omrežju, in si izmenjujejo podatke med seboj.

11 OLED

OLED (Organic Light Emitting Diode) zaslon na Lolin ESP32 je zaslon, ki uporablja organske snovi za oddajanje svetlobe. Zaslon je sestavljen iz posameznih pikslov, ki oddajajo svetlobo in sestavljajo sliko. OLED zasloni so zelo priljubljeni zaradi svojega visokega kontrasta in nizke porabe energije. Na Lolin ESP32, OLED zaslon omogoča prikazovanje besedila in grafičnih elementov, kot so ikone in slike. To omogoča uporabnikom, da hitro preverijo in spremljajo različne parametre, kot so čas, temperatura in senzorski podatki. Poleg tega OLED zasloni omogočajo zelo hitro osveževanje slike, kar omogoča tekoče in gladko uporabniško izkušnjo. (wikipedia, wikipedia, free encylopedia, 2023)



Slika 4 - OLED zaslon

12 RGB diode

Odločali smo se med tremi RGB LED diodami, to so WS2811, WS2815 in še WS2812B, za katerega smo se odločili, da ga bomo uporabili. Razlike med trakovi so dokaj minimalne, a so kljub temu opazne in pri natančnosti zelo pomembne.

WS2811 je prva generacija WS281x serije LED trakov. Sestavljen je iz enega kontrolerja in ene LED diode na istem ohišju. Vsaka LED dioda ima svoj WS2811 kontroler, kar omogoča natančno upravljanje barv za vsako diodo posebej. WS2811 omogoča zaporedno povezovanje več LED diod za ustvarjanje daljšega LED traku.

WS2815 je nadgradnja WS2811. Razlika je v tem, da ima vsaka LED dioda dva kontrolerja, kar zagotavlja boljšo varnost in stabilnost pri zaporednem povezovanju več diod. Če ena LED dioda odpove, lahko še vedno nadaljuje delovanje ostalih diod v verigi. Zaradi dveh kontrolerjev na diodo je ta generacija LED stripa ena izmed dražjih.

WS2812B je druga generacija WS281x serije LED trakov in ima enako funkcionalnost kot WS2811, vendar z izboljšano zmogljivostjo in boljšo energetsko učinkovitostjo. Poleg tega ima WS2812B večjo svetilnost in bogatejše barve kot WS2811.

Na splošno lahko rečemo, da sta WS2815 in WS2812B naprednejši različici WS2811, ki sta bolj varni in učinkoviti.

12.1 WS2812B

Pri našem projektu smo uporabili WS2812B diode. Tej diode rečemo tudi inteligentna dioda, ker vsebuje majhno mikrokontrolersko enoto, ki jo nadzoruje. Dioda ima tri barvne kanale: rdečo (R), zeleno (G) in modro (B), ki se lahko nastavijo v različne kombinacije, da ustvarijo poljubno barvo. Kontroler v diodi sprejme podatke o barvah preko enosmerne serijske komunikacijske povezave in jih nato razdeli na tri kanale (R, G in B). Ti podatki se nato uporabijo za določanje jakosti svetlosti vsake barve v diodi.

Podatki se prenašajo v obliki zaporedja bitov, ki predstavljajo podatke o barvah za vse diode, ki so povezane v verigo. Ta zaporedja bitov se pošljejo preko enega izhoda mikrokrmilnika, ki je povezan s prvim WS2812B čipom v verigi. Vsak naslednji WS2812B čip je povezan s prejšnjim in prejme podatke, ki jih potrebuje za določanje svoje barve.

Ko se podatki prejmejo, se barve v diodi določijo z uporabo PWM (Pulse Width Modulation) signala, ki uravnava jakost svetlosti posamezne barve. To omogoča, da se barve v diodi lahko natančno nadzorujejo in prilagajajo glede na želene barve in učinke.

WS2812B diode so zelo priljubljene zaradi svoje enostavne uporabe in programabilnosti, kar omogoča ustvarjanje različnih svetlobnih učinkov in vzorcev.

12.1.1 Prednosti in slabosti

WS2812B dioda je znana predvsem po enostavni uporabi in enostavnem programiranju za ustvarjanje različnih svetlobnih učinkov. Za nas je tudi pomembno, da imajo te diode relativno nizko porabo energije, kar je pomembno pri našem projektu, kjer bomo uporabili veliko teh diod. So pa tudi zelo pogosto uporabljene, kar pomeni, da so zelo dostopne in jih je mogoče kupiti po zelo ugodni ceni. Imajo pa tudi nekaj slabosti, na primer te diode so občutljive na napetost, kar lahko vodi do prenehanja delovanja ob prekomerni ali nestabilni napetosti. Zaradi nizke porabe energije pa tudi pomeni, da so manj svetle kot novejši modeli.

Imajo tudi težave s sinhronizacijo, kar lahko vodi do neenakomernega osvetljevanja in drugih neželenih učinkov.

12.2 WS2815

WS2815 je vrsta RGB LED diode, ki je podobna WS2812B, vendar ima nekaj izboljšav in razlik. WS2815 je tudi digitalna LED dioda, ki se lahko nadzira posamično po vsakem barvnem kanalu, kar omogoča veliko prilagodljivost pri ustvarjanju svetlobnih učinkov.

Ena izmed največjih razlik med WS2815 in WS2812B diodo je, da ima WS2815 dva napajalna pina (VDD in GND) ter en signalni pin (DI), medtem ko ima WS2812B samo en napajalni pin in en signalni pin. To omogoča, da WS2815 dioda bolj učinkovito obvladuje napajanje in zmanjša možnost, da se celotna veriga diod izklopi, če napajanje ne deluje pravilno.

Druga pomembna izboljšava pri WS2815 diodi je, da je njeno vezje odporno na nizko napetostno stanje, kar omogoča, da dioda še vedno deluje, tudi če je napetost na signalnem pinu nižja od zahtevane napetosti za delovanje. To zmanjša možnost napak in neželenih učinkov na svetlobni učinek, ki bi se lahko pojavili v primeru, da signalni signal ni popolnoma sinhroniziran.

WS2815 dioda se lahko uporablja tudi z višjimi napetostmi kot WS2812B dioda, kar je lahko koristno pri projektih, ki zahtevajo večje število diod ali pa projektov, ki uporabljajo daljše trakove.

Ta dioda ima tudi enako število bitov za vsako barvo (8 bitov na kanal) kot WS2812B dioda in omogoča do 16,8 milijona barvnih kombinacij. WS2815 dioda se pogosto uporablja v svetlobnih projektih, kot so LED zasloni in osvetlitev notranjosti, zlasti v projektih, ki zahtevajo boljše nadzorovanje napajanja in manjše tveganje za napake pri delovanju.

12.3 WS2813

WS2813 je tudi vrsta digitalne LED diode, ki je podobna WS2812B in WS2815, vendar ima nekaj izboljšav in razlik. Kot pri WS2815 ima WS2813 dva napajalna pina (VDD in GND) in en signalni pin (DI), kar omogoča bolj učinkovito obvladovanje napajanja in zmanjšuje tveganje za napake v celotni verigi diod.

Ena izmed ključnih razlik med WS2813 in prejšnjima diodama je vgrajen signalni izolator, ki zmanjšuje možnost, da se celotna veriga diod izklopi, če ena dioda ne deluje pravilno. To zagotavlja večjo zanesljivost pri delovanju večjih svetlobnih projektov, saj je manj verjetno, da bodo celotni verigi diod okvarjeni zaradi napake na eni diodi.

Poleg tega ima WS2813 dioda tudi zaščito pred obratno polariteto, kar zmanjšuje možnost napak pri priključitvi diode na napajanje.

13 OTA

OTA (Over-The-Air) posodobitev pomeni, da lahko posodobitve programske opreme izvedemo brez potrebe po neposredni povezavi z napravo, na kateri se izvaja. To je uporabno pri napravah, ki so nameščene na težko dostopnih mestih ali so vgrajene v procese, ki jih ni mogoče ustaviti. (TechTarget, 2023)

Pri ESP32 lahko OTA posodobitev uporabimo za posodabljanje programske opreme na daljavo, brez potrebe po fizičnem dostopu do modula. To omogoča enostavno posodabljanje programske opreme, tudi če je modul že nameščen in deluje.

Uporaba OTA pri ESP32 je precej preprosta. Obstajajo knjižnice, kot je na primer ArduinoOTA, ki omogočajo OTA posodobitve v okolju Arduino IDE. Postopek je nekako sledeč.

Napravo, ki jo želimo posodobiti, moramo pripraviti tako, da podpira OTA posodobitve. To pomeni, da moramo v kodo programa vključiti podporo za OTA posodobitve. Knjižnica ArduinoOTA omogoča enostavno implementacijo te funkcionalnosti.

V omrežje, v katerem se nahaja ESP32, moramo namestiti OTA strežnik. To je računalnik ali strežnik, ki bo skrbel za posodabljanje programske opreme. Uporabimo lahko že obstoječi strežnik ali pa namestimo namenski OTA strežnik.

Ko sta naprava in OTA strežnik pripravljena, lahko začnemo z OTA posodobitvijo. To storimo tako, da v Arduino IDE izberemo meni "Sketch" > "Export compiled Binary" in ustvarimo binarno datoteko, ki vsebuje posodobljeno kodo. Nato to datoteko prenesemo na OTA strežnik.

Ko je datoteka na strežniku, lahko izvedemo OTA posodobitev tako, da se povežemo z napravo prek Wi-Fi omrežja in uporabimo namensko aplikacijo ali spletno stran, ki omogoča izvedbo posodobitve. OTA posodobitev se bo nato izvedla samodejno in bo trajala nekaj časa, odvisno od velikosti programske opreme in hitrosti omrežja.

OTA posodobitve so zelo uporabne pri razvoju IoT naprav, saj omogočajo enostavno posodabljanje programske opreme na daljavo, kar lahko zmanjša stroške vzdrževanja in izboljša uporabniško izkušnjo.

Slika 5 - OTA in ESP32

Vir: https://randomnerdtutorials.com/esp32-over-the-air-ota-programming/

14 JSON

JSON (JavaScript Object Notation) je format za shranjevanje in prenos podatkov, ki temelji na JavaScriptu. Uporablja se predvsem v spletnem programiranju, saj je lahek, berljiv in razumljiv tako ljudem kot strojem. Podpira različne tipe podatkov, vključno z nizi, števili, boolovimi vrednostmi, objekti in seznami. (Crockford, 2023)

Pri prenosu RGB vrednosti na ESP32 preko WebSocket bi lahko uporabili JSON za enostavno in učinkovito kodiranje in dekodiranje podatkov. Na primer, lahko bi uporabili naslednjo strukturo JSON podatkov:

```
{
  "red": 255,
  "green": 127,
  "blue": 0
}
```

V tem primeru bi "red", "green" in "blue" predstavljali vrednosti rdeče, zelene in modre barve, ki bi jih želeli prikazati na RGB LED traku. Te vrednosti bi lahko uporabnik vnesel v aplikaciji na mobilnem telefonu in jih poslal na ESP32 preko WebSocket.

Na strani ESP32 bi lahko uporabili knjižnico JSON za dekodiranje prejetih podatkov. Na primer, če bi uporabljali Arduino IDE za programiranje ESP32, bi lahko uporabili vgrajeno knjižnico ArduinoJson. S pomočjo te knjižnice bi lahko dekodirali prejete podatke in jih nato poslali na RGB LED trak preko PWM izhoda.

15 Komunikacija

ESP32 lahko komunicira z mobilnim telefonom preko WiFi-ja na več načinov, odvisno od uporabljene programske opreme in potreb. Nekateri načini komunikacije, ki so na voljo, so:

1. HTTP / REST API: ESP32 lahko deluje kot HTTP strežnik in ponuja REST API, preko katerega lahko mobilna aplikacija pošilja zahtevke za branje in posodabljanje podatkov. Ta metoda komunikacije je primerna za preprosto uporabo podatkov brez potrebe po realnem času.
2. WebSocket: ESP32 lahko uporablja WebSocket protokol za dvosmerno komunikacijo z mobilno aplikacijo. Ta metoda omogoča hitrejšo in bolj odzivno

komunikacijo v realnem času, saj lahko strežnik pošilja sporočila katerikoli čas, ne glede na to, ali je mobilna aplikacija zahtevala posodobitve.

3. MQTT: ESP32 lahko uporablja MQTT protokol za komunikacijo z mobilno aplikacijo. Ta metoda omogoča učinkovito in zanesljivo prenašanje sporočil v realnem času, kar je še posebej koristno za aplikacije, ki zahtevajo visoko hitrost komunikacije, npr. senzorske mreže.

4. UDP: ESP32 lahko uporablja UDP protokol za pošiljanje in prejemanje sporočil z mobilno aplikacijo. Ta metoda omogoča hitro in preprosto prenašanje sporočil v realnem času brez potrebe po potrditvi prejema sporočila.

5. TCP: ESP32 lahko uporablja TCP protokol za zanesljivo in varno pošiljanje in prejemanje sporočil z mobilno aplikacijo. Ta metoda je primerna za aplikacije, ki zahtevajo visoko stopnjo varnosti in zanesljivosti, npr. bančne aplikacije.

Vse zgoraj navedene metode so lahko uporabljene za vzpostavitev komunikacije med ESP32 in mobilnim telefonom preko WiFi-ja, odvisno od potreb in zahtev aplikacije.

15.1 HTTP

HTTP (Hypertext Transfer Protocol) je protokol, ki se uporablja za prenos hipertekstovnih dokumentov preko omrežja, kot je internet. V kontekstu komunikacije med ESP32 in mobilnim telefonom preko WiFi-ja se HTTP pogosto uporablja kot način komunikacije, ki omogoča preprost in učinkovit način za prenos podatkov.

HTTP komunikacija med ESP32 in mobilno aplikacijo se običajno izvaja tako, da ESP32 deluje kot strežnik HTTP. Mobilna aplikacija lahko pošlje HTTP zahtevek na ESP32 in prejme odgovor. HTTP zahtevek je običajno oblikovan kot URL naslov, ki se nanaša na določen vir na ESP32, na primer določen vir podatkov ali funkcijo.

Zahtevek HTTP ima lahko tudi določene parametre, ki se uporabljajo za prenos podatkov. Na primer, če aplikacija zahteva posodobitev določenega vira podatkov, lahko HTTP zahtevek vsebuje parametre, ki opisujejo, katere podatke je treba posodobiti in kakšne so nove vrednosti. ESP32 lahko nato uporabi te podatke za posodobitev vira podatkov in odgovoriti z odgovorom HTTP, ki vsebuje statusno kodo, ki opisuje, ali je bil zahtevek uspešen ali ne, in morebitne druge podatke, ki so zahtevani.

Poleg preprostega načina za prenos podatkov HTTP komunikacija omogoča tudi učinkovit nadzor nad dostopom do virov na ESP32. Na primer, lahko se določijo omejitve dostopa do posameznih virov ali funkcij na ESP32, kar pomeni, da lahko aplikacija dostopa do določenih virov, samo če ima ustrezna dovoljenja.

V skladu s tem HTTP komunikacija med ESP32 in mobilno aplikacijo preko WiFi-ja zagotavlja enostavno in učinkovito rešitev za prenos podatkov, ki temelji na standardiziranem protokolu, ki ga podpirajo številne platforme in naprave.

15.1.1 GET

GET je HTTP metoda, ki se uporablja za pridobivanje podatkov s spletnega strežnika. Zahteva GET pošlje zahtevek za določenim virom, ki je na voljo na spletnem strežniku, in prejme odgovor, ki vsebuje zahtevane podatke. Podatki se v odgovoru običajno vračajo v obliki HTML, JSON ali drugega formata podatkov.

GET zahteva se običajno uporablja, ko se aplikacija želi povezati z določenim virom podatkov na spletnem strežniku in dobiti informacije o tem viru, na primer, ko se aplikacija prikaže na zaslonu uporabnika ali kadar se podatki spreminjajo. Na primer, aplikacija bi lahko uporabila zahtevo GET za pridobitev trenutnih temperaturnih podatkov iz senzorja, ki je povezan z ESP32.

Pri ESP32 se lahko GET zahteva uporablja za pridobivanje podatkov iz senzorjev ali drugih naprav, ki so povezane z njim, in pošiljanje teh podatkov na spletni strežnik. S tem se omogoči, da se podatki posredujejo aplikacijam ali drugim uporabnikom preko spleta, kar zagotavlja učinkovito in preprosto rešitev za prenos podatkov.

Na primer, ESP32 bi lahko uporabil zahtevo GET za pridobitev trenutnih vrednosti temperaturnih podatkov iz senzorja in poslal te podatke na spletni strežnik. Aplikacija, ki je povezana s spletnim strežnikom, bi nato lahko pridobila te podatke in jih prikazala na zaslonu uporabnika.

Pomembno je opozoriti, da je GET zahteva javna in se lahko shranjuje v predpomnilniku brskalnika ali strežnika. Zato je priporočljivo, da se pri uporabi GET zahtev uporabi ustrezna avtentikacija in avtorizacija, da se zagotovi zaščita pred nezaželenim dostopom do virov podatkov.

15.1.2 POST

POST je HTTP metoda, ki se uporablja za pošiljanje podatkov na spletni strežnik. Zahteva POST vsebuje podatke, ki se bodo poslali na spletni strežnik, in običajno povzroči ustvarjanje novega vira na strežniku ali posodobitev obstoječega vira. Podatki se v zahtevi POST pošljejo v telesu zahteve.

Pri ESP32 se lahko zahteva POST uporablja za pošiljanje podatkov iz senzorjev ali drugih naprav na spletni strežnik. S tem se omogoči, da se podatki shranijo na spletni strežnik in so na voljo za nadaljnjo uporabo. Na primer, ESP32 bi lahko uporabil zahtevo POST za pošiljanje temperaturnih podatkov na spletni strežnik, ki bi nato shranil te podatke v bazo podatkov.

Poleg tega se lahko zahteva POST uporablja tudi za pošiljanje podatkov za obdelavo na spletni strani, kot je na primer obrazec za registracijo uporabnika ali za oddajo komentarja na spletni strani. Podatki, ki jih uporabnik vnese v obrazec, se pošljejo na spletni strežnik s pomočjo zahteve POST, ki jih strežnik obdeluje in shrani.

Pomembno je opozoriti, da se podatki v zahtevi POST ne shranjujejo v predpomnilniku brskalnika ali strežnika. To pomeni, da se lahko zahteve POST uporabljajo za pošiljanje občutljivih podatkov, kot so gesla in osebni podatki, brez tveganja, da bi jih drugi lahko

videli ali uporabili. Vendar je kljub temu priporočljivo, da se pri uporabi zahtev POST uporabi ustrezna avtentikacija in avtorizacija, da se zagotovi zaščita pred nezaželenim dostopom do virov podatkov.

15.2 WebSocket

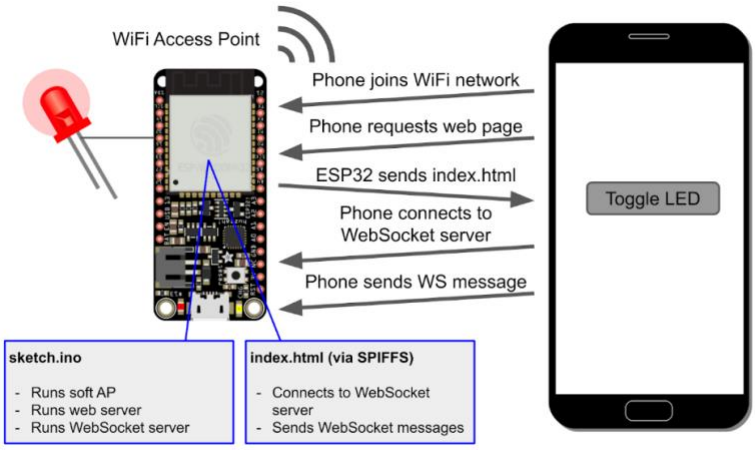
WebSocket je protokol za dvosmerno komunikacijo med odjemalcem in strežnikom preko TCP povezave. Omogoča stalno odprto povezavo med odjemalcem in strežnikom, ki omogoča pošiljanje sporočil med njima brez potrebe po ponovni vzpostavitvi povezave za vsako sporočilo posebej. (wikipedia, wikipedia, free encylopedia, 2023)

Komunikacija prek WebSocketa se začne s postopkom "ročnega stiskanja roke" (handshaking), ki se izvede s pomočjo HTTP protokola. Odjemalec pošlje zahtevo za vzpostavitev povezave prek HTTP protokola s posebnim glavičem (headerjem) "Upgrade" in "Connection". Če strežnik podpira WebSocket protokol, odgovori s pravilnim odgovorom, ki vsebuje tudi ključni glavič "Sec-WebSocket-Accept". Po tem se povezava vzpostavi in omogoči stalno dvosmerno komunikacijo.

Medtem ko HTTP deluje s pravili "zahtevanja-odgovora" (request-response), WebSocket protokol omogoča prenos sporočil v obeh smereh, ne da bi bil potreben nov zahtevek vsakič, ko se sporočilo pošlje. Odjemalec lahko kadarkoli pošlje sporočilo na strežnik, ki ga ta prejme in nanj odgovori s svojim sporočilom. Prenos sporočil je v obliki "paketov" (pakets), kjer se vsak paket sestoji iz naslova (headerja) in sporočila (payloada).

WebSocket protokol se lahko uporablja za različne aplikacije, kjer je potrebna hitra in zanesljiva dvosmerna komunikacija med odjemalcem in strežnikom. Primeri uporabe vključujejo realnočasne igre, klepetalnice (chat), spremljanje delovanja senzorjev in drugih naprav ter posodobitve v realnem času v različnih aplikacijah.

Pri uporabi WebSocket protokola s ESP32 lahko odjemalci pošiljajo sporočila prek Wi-Fi ali Ethernet povezave na strežnik, ki lahko sprejme in obdeluje ta sporočila. ESP32 lahko prejema tudi sporočila s strežnika in odgovarja nanje. ESP32 ima vgrajeno podporo za WebSocket protokol, kar omogoča enostavno implementacijo tega protokola v projektih z ESP32.



Slika 6 – WebSocket

Vir: https://shawnhymel.com/1882/how-to-create-a-web-server-with-websockets-using-an-esp32-in-arduino/

16 Ideja

Izdelek, ki smo si ga zamislili, je pametna RGB luč, ki jo lahko krmilimo s pomočjo ESP32 mikrokrmilnika preko Wi-Fi povezave s svojega pametnega telefona. Za komunikacijo med napravami uporabljamo WebSocket protokol, ki omogoča stalno dvosmerno komunikacijo med odjemalcem in strežnikom.

Uporabnik lahko na svojem telefonu izbere različne barve in svetlosti luči, kot tudi različne efekte, ki se izvajajo na lučeh. Za merjenje svetlosti in barve, ki jo želimo nastaviti, se uporablja JSON format, ki se prenaša prek WebSocketa na mikrokrmilnik ESP32. Mikrokrmilnik nato te podatke uporabi za krmiljenje RGB luči.

Glavni poudarek izdelka bi bil na natančni kontroli nad barvo in svetlostjo luči, ki jo lahko uporabnik prilagodi svojim željam. Uporaba WebSocketa omogoča hitro in zanesljivo komunikacijo med napravami, kar zagotavlja hitro odzivnost in pravilno delovanje sistema.

16.1 Tehnična izvedba

Mikrokrmilnik ESP32 komunicira z uporabnikom preko spletne strani, ki se nahaja na gostiteljskem strežniku. Uporabnik na spletni strani sporoči barvo, ki jo želi nastaviti preko WebSocket povezave.

Ob zagonu se program poveže na omrežje Wi-Fi in izpiše svoj naslov IP. Po uspešni povezavi se začne nalaganje spletne vsebine iz datoteke shranjene v spominskem pomnilniku SPIFFS. Nato se začne tudi inicializacija zaslona SSD1306 in prikaz uvodnega besedila na zaslonu.

V funkciji initWebSocket se inicializira WebSocket povezava na podani poti /ws. Na dogodke WebSocket povezave se odzove funkcija onEvent.

Ob vsakem prejetem sporočilu se v funkciji handleWebSocketMessage iz luščenega niza izloči R, G in B vrednost, ki jo uporabnik želi nastaviti.

Ko je povezava prekinjena ali vzpostavljena, se o tem izpiše obvestilo preko serijske povezave. Ko uporabnik spremeni barvo, se vrednosti RGB osvežijo in prikažejo preko serijske povezave.

16.2 Izbrana komunikacija

Za komunikacijo med napravami smo sprva uporabljali HTTP protokol, vendar smo kmalu ugotovili, da se hitrost prenosa podatkov s tem protokolom ne more kosati z našimi potrebami. Število zahtev, ki jih moramo poslati in prejeti za upravljanje luči iz več naprav hkrati, je preprosto previsoko za HTTP. Poleg tega je osveževanje stanja barve, ki se izvaja na vseh napravah, s HTTP protokolom precej počasno in ni bilo sprotnega osveževanja.

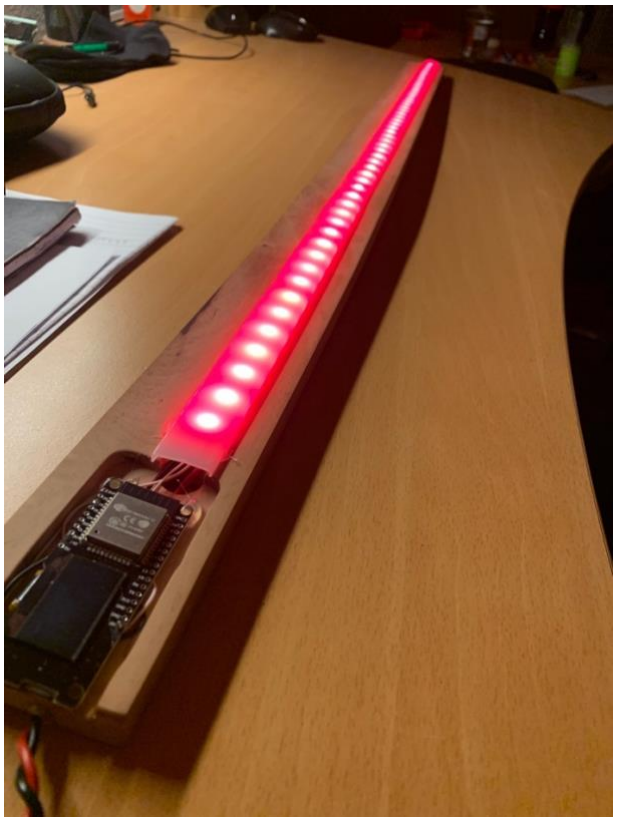
Zato smo se odločili za uporabo WebSocket protokola, saj ta protokol omogoča hitro in enostavno komunikacijo med napravami. WebSocket protokol omogoča stalno odprto povezavo med klientom in strežnikom, kar zmanjšuje število potrebnih zahtev in odgovorov, kar na koncu omogoča hitrejši prenos podatkov.

Zaradi WebSocket protokola lahko zdaj upravljamo luči iz več telefonov hkrati in osvežujemo stanje izbrane barve na vseh napravah, kar nam omogoča, da so naše aplikacije hitrejše in odzivnejše.

17 Izdelek

Za prototip smo uporabili 3D-tiskalnik in natisnili lahko stojalo za ESP32. Nato smo priklopili tri WS2812b diode na ustrezne pine in vse skupaj sestavili v eno funkcionalno enoto. S tem smo ustvarili preprost, vendar učinkovit način za testiranje delovanja diod v povezavi z ESP32.

Slika 7 - 3D Tiskanje



Slika 8 - Izdelek

17.1 Problemi pri nastajanju končnega izdelka

Med izdelavo končnega projekta smo imeli težave pri tiskanju škatlice za ESP32, saj nam je med postopkom zmanjkalo filamenta. To nas je prisililo, da smo postopek tiskanja ponovili, kar je povzročilo nekaj zamude pri izdelavi.

Ko smo končno izdelali prvo škatlico za ESP32, smo ugotovili, da je bila premajhna. Pri vstavljanju ESP32 v škatlico smo poškodovali dva kondenzatorja, ki sta se odlomila. Poskusili smo ju zamenjati, vendar smo ugotovili, da sta bila poškodovana tudi dva pina na samem ESP32, kar je pomenilo, da smo morali nadomestiti celoten modul.

Med spajkanjem diod smo preveč zagreli pine, kar je povzročilo, da so nekateri pini odstopili, kar je vodilo do poškodbe nekaterih RGB diod. Kljub tem težavam smo se naučili pomembne lekcije o tem, kako skrbno izbrati in izvesti pravilne postopke pri izdelavi končnega izdelka ter se iz teh izkušenj naučili, kako izboljšati naš proces izdelave v prihodnosti.

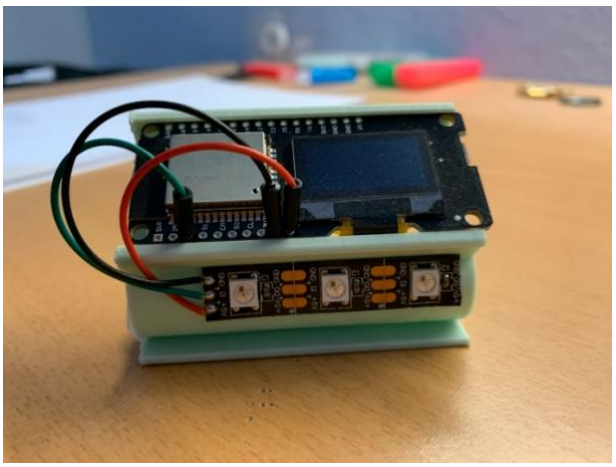
17.1.1 Utripanje luči

Med našim projektom smo bili zaskrbljeni glede počasnega osveževanja luči, saj smo ugotovili, da bi to lahko povzročilo utripanja na kameri. Preizkusili smo več različnih vrst luči in kamer, vendar smo še vedno opazili nekaj utripanja.

Sprva smo v našem projektu uporabljali WS2812 diode. Vendar smo med testiranjem ugotovili, da bi potrebovali nekaj več natančnosti in hitrosti osveževanja, da bi dosegli boljšo kakovost svetlobe. Zato smo se odločili, da bomo nadomestili WS2812 diode z WS2812b diodami.

WS2812b diode imajo nekaj prednosti, ki so nam bile pomembne. Imajo hitrejšo osvežitev kot prejšnje diode, kar je izboljšalo natančnost svetlobe. Poleg tega so tudi bolj zanesljive in učinkovite, kar nam je omogočilo boljši nadzor nad svetlobnim sistemom.

Po zamenjavi starih WS2812 diod z novimi WS2812b diodami smo bili zadovoljni z rezultati. Svetloba je bila natančnejša in bolj enakomerna, kar je izboljšalo kakovost našega projekta. Skupaj z drugimi izboljšavami smo s tem zagotovili optimalno delovanje našega svetlobnega sistema.



Slika 9 - Naš prototip

17.1.2 Baterija

Na žalost smo naleteli na omejitev pri izbiri baterij za naše RGB luči za fotografiranje. Zaradi omejenega proračuna nismo mogli uporabiti dovolj močnih baterij, ki bi zagotavljale zadostno moč za daljše obdobje uporabe. Želeli smo, da bi izdelek lahko deloval samostojno vsaj eno uro, vendar nam to ni uspelo doseči. Kljub temu smo uspeli ustvariti izdelek, ki deluje in izpolnjuje svoj namen, vendar zahteva stalno napajanje.

18 Komunikacija OLED zaslona v kodi

Za povezavo OLED zaslona uporabljamo dvonitno vezje I2C, ki je sestavljeno iz dveh signalnih linij: SDA (Data) in SCL (Clock). SDA in SCL povezujemo z digitalnima pini D5 in D4 na ESP32.

Za uporabo knjižnice Adafruit_SSD1306 vključimo knjižnico Wire.h, ki nam omogoča upravljanje z I2C komunikacijo.

Za ustvarjanje objekta zaslona (display) uporabimo spodnjo kodo:

```
25 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
26
```

Slika 10 - Komunikacija OLED zaslona v kodi

Za prikazovanje besedila na OLED zaslonu uporabljamo spodnjo kodo:

```
  display.setTextSize(2); // Draw 2X-scale text
display.setTextColor(SSD1306_WHITE);
display.setCursor(0, 5);
display.clearDisplay();
display.println("R: " + R_string);
display.println("G: " + G_string);
display.println("B: " + B_string);
display.display();      // Show initial text
delay(10);
```

Slika 11 - Koda za prikazovanje besedila na OLED zaslonu

Koda pripravi OLED zaslon tako, da:

1. Nastavi velikost pisave na 2-kratno velikost s pomočjo display.setTextSize(2);.
2. Nastavi barvo besedila na belo s pomočjo display.setTextColor(SSD1306_WHITE);.
3. Nastavi začetno pozicijo besedila na vrstico 5, stolpec 0 s pomočjo display.setCursor(0, 5);.
4. Počisti zaslon z pomočjo display.clearDisplay();.
5. Izpiše tri vrstice besedila, ki predstavljajo vrednosti za rdečo, zeleno in modro barvo, s pomočjo display.println("R: " + R_string);, display.println("G: " + G_string); in display.println("B: " + B_string);.
6. Prikazuje izpisano besedilo na zaslonu s pomočjo display.display();.
7. Počaka 10 milisekund s pomočjo delay(10);.

19 WIFI komunikacija

Mikrokrmilnik je povezan na WiFi omrežje s poverilnicami, ki so shranjene v spremenljivkah ssid in geslo.

Koda ustvari objekt AsyncWebServer na portu 80 in ga doda v strežnik. Strežnik je konfiguriran tako, da uporabnikom omogoča dostop do osnovne HTML strani (index.html) in css datoteke (style.css), ki se nahajata na datotečnem sistemu mikrokrmilnika (SPIFFS).

Funkcija handleWebSocketMessage() se kliče ob vsakem prejemu sporočila prek WebSocket povezave. Prejeta sporočila se razdelijo na posamezne R, G in B vrednosti in nato uporabijo za nastavitev barve diod.

19.1 handleWebSocketMessage()

```
56 void handleWebSocketMessage(void *arg, uint8_t *data, size_t len) {
57   AwsFrameInfo *info = (AwsFrameInfo*)arg;
58   if (info->final && info->index == 0 && info->len == len && info->opcode == WS_TEXT) {
59     data[len] = 0;
60     message = (char*)data;
61     Serial.println(message);
62
63     if(message.indexOf("R") >= 0){
64       message.remove(0, 2);
65       R_string = message;
66       R = R_string.toInt();
67       Serial.println(R);
68     }
69     else if(message.indexOf("G") >= 0){
70       message.remove(0, 2);
71       G_string = message;
72       G = G_string.toInt();
73       Serial.println(G);
74           }
75     else if(message.indexOf("B") >= 0){
76       message.remove(0, 2);
77       B_string = message;
78       B = B_string.toInt();
79       Serial.println(B);
80     }
81
82
83     |
84     //message = String(data);
85
86   }
87 }
```

Slika 12 - funkcija handleWebSocketMessage

Tukaj definiramo funkcijo handleWebSocketMessage, ki se uporablja za obdelavo sporočil WebSocket. Funkcija sprejme tri argumente: arg, data, in len. Funkcija preveri, ali je prišlo celotno sporočilo, nato sporočilo shranjuje v spremenljivko message. Če sporočilo vsebuje "R", "G" ali "B", iz sporočila izbriše prvih dva znaka in preostali niz pretvori v celo število. Na koncu funkcija izpiše vrednosti R, G in B.

19.2 Pošiljanje RGB vrednosti iz telefona na ESP

Ta koda vzpostavi WebSocket povezavo s strežnikom na naslovu ws://${window.location.hostname}/ws. Ko se povezava vzpostavi, se sprožita funkciji onOpen in initButton. Ko uporabnik klikne na gumb z ID-jem "button", se sproži funkcija toggle, ki pa ni prikazana v tem kodnem primeru. Namesto tega se sproži funkcija myFunction, ki se izvede ob kliku na gumb.

Funkcija myFunction pridobi vrednost barve iz polja z ID-jem "myColor" in jo shrani v spremenljivko x. Nato pretvori vrednost iz hex oblike v RGB obliko in posamezne vrednosti R, G in B shranjuje v spremenljivke r, g in b, ki so oblikovane v skladu s protokolom, ki ga pričakuje strežnik. Nato funkcija pošlje spremenljivke r, g in b preko

WebSocket povezave s funkcijo websocket.send(). Tako se RGB vrednost pošlje na strežnik preko WebSocket protokola.

```

    var gateway = `ws://${window.location.hostname}/ws`;
    var websocket;
    var x;
    window.addEventListener('load', onLoad);
    function initWebSocket() {
      console.log('Trying to open a WebSocket connection...');
      websocket = new WebSocket(gateway);
      websocket.onopen    = onOpen;
      websocket.onclose   = onClose;
      websocket.onmessage = onMessage; // <-- add this line
    }
    function onOpen(event) {
      console.log('Connection opened');
    }
    function onClose(event) {
      console.log('Connection closed');
      setTimeout(initWebSocket, 2000);
    }
    function onMessage(event) {

    }
    function onLoad(event) {
      initWebSocket();
      initButton();
    }
    function initButton() {
      document.getElementById('button').addEventListener('click', toggle);
    }

    function myFunction() {
    x = document.getElementById("myColor").value;
    document.getElementById("demo").innerHTML = x;

    var newStr = x.replace('#','')
    var RGB = newStr.convertToRGB()
    r = "R." + RGB[0];
    g = "G." +RGB[1];
    b = "B." +RGB[2];

  console.log(r);
  console.log(g);
  console.log(b);
    websocket.send(r);
    websocket.send(g);
    websocket.send(b);
  }
  var r;
  var g;
  var b;
  String.prototype.convertToRGB = function(){
      if(this.length != 6){
          throw "Only six-digit hex colors are allowed.";
      }

      var aRgbHex = this.match(/.{1,2}/g);
      var aRgb = [
          parseInt(aRgbHex[0], 16),
          parseInt(aRgbHex[1], 16),
          parseInt(aRgbHex[2], 16)
      ];
      return aRgb;
  }

  console.log(x.convertToRGB());
```

Slika 13 - Kode za pošiljanje RGB

20 Rezultati

Naša raziskovalna naloga se je osredotočala na razvoj sistema RGB luči za fotografiranje, ki jih krmili ESP32. V teoretičnem delu smo se posvetili predvsem preučevanju vseh komunikacij in metod, ki smo se jih morali naučiti za uspešno izvedbo projekta. Podrobno smo preučili različne načine upravljanja RGB luči, vključno z uporabo mikrokrmilnika ESP32 in različnimi načini komunikacije z njim, kot sta Wi-Fi in Bluetooth. Poleg tega smo preučili različne načine uporabe RGB luči za fotografiranje, vključno z različnimi barvnimi filtri, ki se uporabljajo v fotografski industriji.

V praktičnem delu raziskovalne naloge smo uspešno izdelali RGB luči za fotografiranje, ki jih krmili ESP32. Kljub temu da nam zaradi omejenega proračuna ni uspelo izdelati takšnega izdelka, kot smo si ga prvotno zamislili, smo zadovoljni z doseženimi rezultati. Verjamemo, da bi z več denarja in več znanja lahko uspešno izdelali našo idejo, ki smo si jo na začetku zamislili. Kljub temu smo se naučili veliko o izdelavi in programiranju RGB luči ter o uporabi ESP32 za krmiljenje luči, kar nam bo v prihodnosti zagotovo koristilo pri podobnih projektih.

21 Analiza ankete

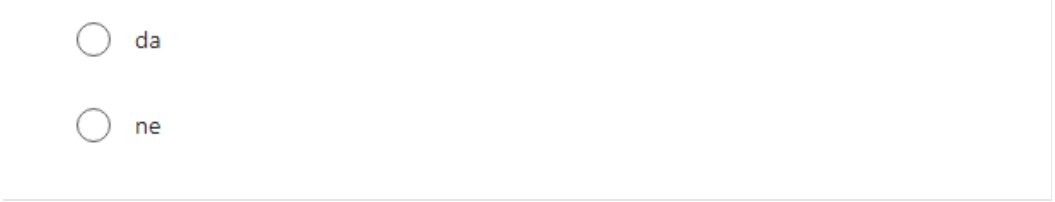
Izvedli smo spletno anketo, s katero smo želeli ugotoviti, če naše anketirance zanima nakup našega izdelka. Pridobili smo podatke, če anketiranci fotografirajo in kako pomembne so jim funkcije, ki jih ponuja naš izdelek. V raziskovalno nalogo smo vnesli vsa vprašanja, razlage, razlog, zakaj smo si zastavili ta vprašanja. Nato smo predstavili dobljene rezultate z grafikoni, ki prikazujejo odgovore v odstotkih.

21.1 Anketa – rezultati ankete

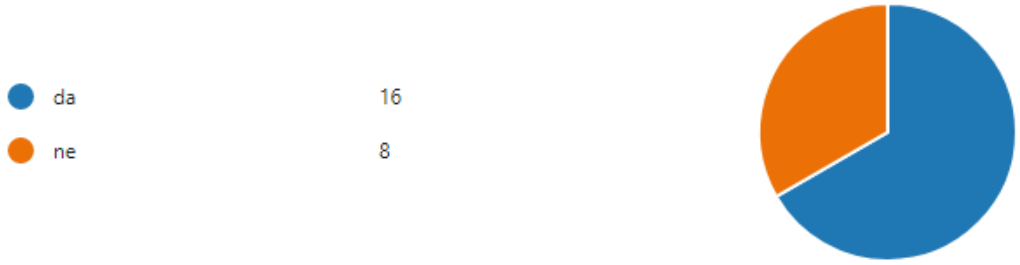
Odločili smo se izvesti spletno anonimno anketo, da bi zbrali podatke o anketirancih. Anketirali smo 24 oseb s Šolskega centra Celje, od tega je bilo 50 % anketirancev, ki so nam povedali, da fotografirajo, 13 % da ne fotografirajo in 38 %, da fotografirajo le občasno.

Ugotovili smo, da kar 67 % anketirancev uporablja osvetljavo za fotografiranje, kar pomeni, da bi potencialno zanimanje za naš izdelek bilo kar veliko.

Ali uporabljate studijske luči za fotografiranje?



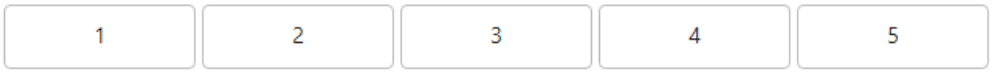
Slika 14 - Vprašanje na anketi



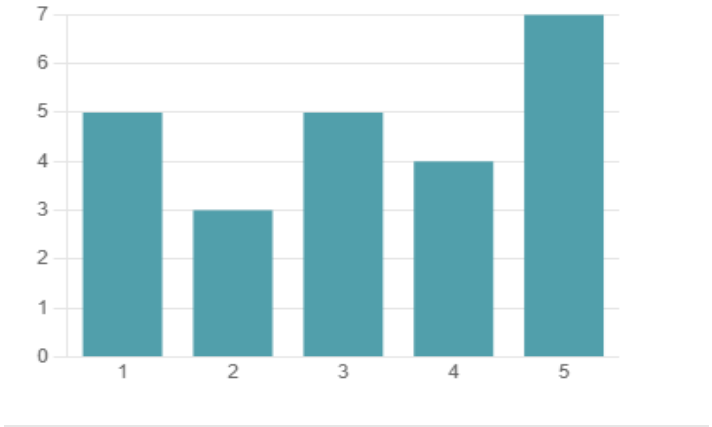
Graf 1 - Prvo vprašanje

V naslednjem vprašanju “kako pomembna vam je cena, če bi kupili studijsko luč?” nas je zanimalo, če bi naš izdelek sploh imel smisel, glede na to, da je naš cilj ustvariti cenovno ugoden izdelek. Največ jih je odgovorilo, da jim je cena zelo pomembna, povprečna ocena pomembnosti, pa je 3,21.

Kako pomembna vam je cena, če bi kupili studijsko luč? 1 - ni pomembna, 5 - zelo pomembna.



Slika 15 - Vprašanje na anketi 2



Graf 2 - Drugo vprašanje

Anketirance smo vprašali tudi, če jim je pomembna lastnost, da so luči zmogljive prikazovanja več barv. Večini anketirancev ta lastnost je pomembna, saj je z “da” odgovorilo kar 71 % anketirancev.

Ali vam je pomembno, da so luči zmogljive prikazovanja več barv?

Slika 16 - Vprašanje na anketi 3

Graf 3 - Tretje vprašanje

Zanimalo pa nas je tudi, če je našim anketirancem pomembno, da imajo luči zmožnost upravljanja z mobilno napravo. Zopet smo ugotovili, da je večini to pomembno, saj je z “da” odgovorilo 88 % anketirancev.

Ali vam je pomembno, da imajo luči zmožonst upravljanja z mobilno napravo?

Slika 17 - Vprašanje na anketi 4

Graf 4 - Četrto vprašanje

Zadnje vprašanje, ki smo ga zastavili, je “koliko bi bili pripravljeni plačati za studijsko luč, ki spreminja barve, je mobilna in ima zmožnost spreminjanja barv na mobilni napravi?”. Iz analize odgovorov smo ugotovili, da lahko za naš izdelek računamo nekje med 80−200 eur.

Koliko bi bili pripravljeni plačati za studijsko luč, ki spreminja barve, je mobilna in ima zmožnost spreminjanja barv na mobilni napravi?

Slika 18 - Vprašanje na anketi 5

1	anonymous	100€
2	anonymous	200
3	anonymous	50€
4	anonymous	80
5	anonymous	nevem
6	anonymous	500
7	anonymous	120
8	anonymous	200€
9	anonymous	80
10	anonymous	60€
11	anonymous	50
12	anonymous	80
13	anonymous	75
14	anonymous	90
15	anonymous	80

Slika 19 - Rezultati ankete

16	anonymous	120
17	anonymous	60
18	anonymous	200
19	anonymous	70
20	anonymous	90
21	anonymous	100
22	anonymous	220
23	anonymous	150
24	anonymous	140

Slika 20 - Rezultati ankete 2

22 Ovrednotenje hipotez

1. Predvidevamo, da bomo imeli težave predvsem z utripanjem LED diod pri zajemanju s kamero.

   Po opravljenih testih in opazovanjih smo delno potrdili hipotezo " Predvidevamo, da bomo imeli težave predvsem z utripanjem LED diod pri zajemanju s kamero”. Ugotovili smo, da so luči utripale, vendar smo z nadaljnjim testiranjem, izbiro pravih luči ter nastavitvijo prave frekvence uspešno odpravili utripanje in tako zagotovili kakovostno osvetlitev pri fotografiranju. Kljub temu verjamemo, da bi lahko s povečanjem proračuna za projekt ter z dodatnim znanjem na področju izdelave luči, uspešno ustvarili tudi izdelek, ki bi v celoti potrdil našo hipotezo.

2. Mikrokrmilnik ESP32 bo zadostil osnovnim funkcijam, pri naprednih zahtevah pa ne bo dovolj zmogljiv.

   Po opravljenem raziskovalnem delu smo lahko hipotezo " Mikrokrmilnik ESP32 bo zadostil osnovnim funkcijam, pri naprednih zahtevah pa ne bo dovolj zmogljiv" zavrgli, saj smo spoznali, da je ESP32 zelo zmogljiv in nam nudi več kot dovolj za uresničitev naše ideje. Poleg tega je cenovno ugoden in ima veliko virov za raziskovanje in učenje, kar je bilo ključno pri razvoju našega projekta. Zato smo se odločili, da bomo nadaljevali z uporabo ESP32 in izkoristili njegove prednosti ter funkcionalnosti za nadaljnji razvoj našega izdelka.

3. Končni izdelek načrtujemo izdelati za manj kot 50 eur.

Naša hipoteza "Končni izdelek načrtujemo izdelati za manj kot 50 eur" je bila na žalost potrjena. Kljub našim naporom in spretnostim nam je med projektom zmanjkalo sredstev, saj smo si med izdelavo privoščili nekaj napak, ki so zahtevale dodatne stroške. Zaradi tega nismo mogli dokončati našega izdelka do želenega obsega in kakovosti, kot smo si ga prvotno zamislili.

23 Zaključek

V okviru te raziskovalne naloge smo se lotili izdelave RGB luči za fotografiranje, ki jih krmili ESP32, upravljamo pa ga preko telefona. Cilj je bil ustvariti izdelek, ki bi izboljšal kakovost fotografij v slabših svetlobnih pogojih, hkrati pa bi bil cenovno ugoden in enostaven za uporabo. Pri raziskovanju smo se osredotočili na opis potrebnih komunikacij ter raziskali različne vire, ki so nam pomagali pridobiti znanje za izdelavo izdelka.

V praktičnem delu smo uspešno naredili RGB luči za fotografiranje in testirali njihovo delovanje. Naleteli smo na težave pri utripanju luči, ki smo jih uspešno rešili s testiranjem in izbiro pravih luči ter nastavitvijo prave frekvence. Žal pa nam zaradi pomanjkanja sredstev ni uspelo izdelati takšnega izdelka, kot smo si ga prvotno zamislili, kar bi nam omogočilo samostojno delovanje izdelka do ene ure. Kljub temu smo zadovoljni z doseženim in verjamemo, da bi nam z več denarja ter več znanja uspelo ustvariti našo idejo, ki smo si jo na začetku zamislili.

Upamo, da bo naš izdelek v obliki RGB luči za fotografiranje s krmiljenjem preko ESP32 mikrokrmilnika uporabnikom omogočil lažje in boljše fotografiranje ter jim omogočil izražanje svoje kreativnosti na nov način. Kljub temu da izdelek ni bil v celoti izpolnjen v naših načrtih zaradi omejenih finančnih sredstev, smo se naučili veliko novega in pridobili izkušnje na področju elektronike, programiranja in komunikacij, kar nam bo koristilo pri naših prihodnjih projektih.

24 Viri in litreratura

Santos, R. (9. januar 2023). *Random nerd tutorials*. Pridobljeno iz ESP32 Web Server – Arduino IDE: https://randomnerdtutorials.com/esp32-web-server-arduino-ide/

Santos, R. (3. Februar 2023). *Random Nerd tutorials*. Pridobljeno iz ESP32 WebSocket Server: Control Outputs (Arduino IDE): https://randomnerdtutorials.com/esp32-websocket-server-arduino/

Santos, R. (23. februar ¸ 2023). *Random Nerd Tutorials*. Pridobljeno iz ESP32 MQTT – Publish and Subscribe with Arduino IDE: https://randomnerdtutorials.com/esp32-mqtt-publish-subscribe-arduino-ide/

Santos, R. (4. marec 2023). *random nerd tutorials*. Pridobljeno iz ESP32 HTTP GET and HTTP POST with Arduino IDE (JSON, URL Encoded, Text): https://randomnerdtutorials.com/esp32-http-get-post-arduino/

Satntos, R. (16. marec 2023). *random nerd tutorials*. Pridobljeno iz ESP32/ESP8266 RGB LED Strip with Color Picker Web Server: https://randomnerdtutorials.com/esp32-esp8266-rgb-led-strip-web-server/

Santos, R. (11. Februar 20). *random nerd tutorials*. Pridobljeno iz ESP32 OLED Display with Arduino IDE: https://randomnerdtutorials.com/esp32-ssd1306-oled-display-arduino-ide/

wikipedia, f. e. (11. februar 2023). *wikipedia, free encylopedia*. Pridobljeno iz OLED: https://en.wikipedia.org/wiki/OLED

Santos, R. (13. februar 2023). *random nerd tutorials*. Pridobljeno iz ESP32 Web Server using SPIFFS (SPI Flash File System): https://randomnerdtutorials.com/esp32-web-server-spiffs-spi-flash-file-system/

Santos, R. (11. februar 2023). *random nerd tutorials*. Pridobljeno iz ESP32 I2C Communication: Set Pins, Multiple Bus Interfaces and Peripherals (Arduino IDE): https://randomnerdtutorials.com/esp32-web-server-spiffs-spi-flash-file-system/

Krkoč, P. (2014). *Arduino- programirajmo arduino z lahkoto.* Ljubljana: AX elektronika d.o.o .

wikipedia, f. e. (14. 2 2023). *wikipedia, free encylopedia*. Pridobljeno iz Wifi: https://en.wikipedia.org/wiki/Wi-Fi

Espressif. (14. 2 2023). *Espressif*. Pridobljeno iz ESP32: https://www.espressif.com/en/products/socs/esp32

Arduino. (14. 2 2023). *Arduno docs*. Pridobljeno iz IDE: https://docs.arduino.cc/software/ide-v1/tutorials/arduino-ide-v1-basics

Espressif. (14. 2 2023). *Espressif*. Pridobljeno iz SPIFFS Filesystem: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/storage/spiffs.html

Kashif. (20. 2 2023). *linuxhint*. Pridobljeno iz esp23 STA: https://linuxhint.com/esp32-both-access-station-points/

TechTarget. (20. 2 2023). *TechTarget*. Pridobljeno iz OTA update: https://www.techtarget.com/searchmobilecomputing/definition/OTA-update-over-the-air-update

Crockford, D. (23. 2 2023). *JSON*. Pridobljeno iz Intruducing JSON: https://www.techtarget.com/searchmobilecomputing/definition/OTA-update-over-the-air-update

wikipedia, f. e. (23. 2 2023). *wikipedia, free encylopedia*. Pridobljeno iz WebSocket: https://en.wikipedia.org/wiki/WebSocket#:~:text=WebSocket%20is%20a%20computer%20communications,as%20RFC%206455%20in%202011.

25 Priloge

Intervju z uporabnikom

Kakšna je vaša izkušnja z našimi RGB lučmi?

Moja izkušnja z vašimi studijskimi RGB lučmi je zelo pozitivna. Uporabljam jih za snemanje videoposnetkov in fotografiranje in opazil sem izboljšanje kakovosti slik od takrat, ko sem začel uporabljati vaše luči. Njihova prilagodljivost mi omogoča, da jih prilagodim glede na različne svetlobne pogoje in tako zagotovim, da so moje fotografije vedno enakomerno osvetljene. Njihova enostavna uporaba in preprost nadzor nad barvami omogočata, da učinkovito in hitro prilagodim svetlobo za doseganje želenega učinka. Skratka, zelo sem zadovoljen z vašimi studijskimi RGB lučmi in jih z veseljem priporočam drugim uporabnikom.

Ali bi kaj spremenli na našem izdelku?

Kljub temu, da sem zelo zadovoljen z vašimi studijskimi RGB lučmi, sem pomislil, da bi bilo zelo priročno, če bi bile luči opremljene z baterijskim napajanjem, kar bi omogočalo lažjo mobilnost in uporabo zunaj zaprtih prostorov, kjer ni stalne električne povezave. Poleg tega bi lahko dodali analogni način upravljanja, kar bi

uporabniku omogočilo fizično upravljanje, kar bi bilo zelo koristno v nekaterih situacijah.

Ali je bila svetilnost luči dovolj zadovoljiva?

S svetilnostjo RGB luči sem zadovoljen. Za moje potrebe studijske uporabe so bile luči več kot dovolj močne, da sem dosegel želeni učinek osvetlitve. Vendar pa, ko sem jih uporabil zunaj v nekoliko bolj svetlih okoljih, sem opazil, da bi lahko bile luči močnejše za boljšo izpostavljenost in doseganje boljše kakovosti slik. Kljub temu pa je to bolj odvisno od okoljskih pogojev, zato sem zelo zadovoljen s svetilnostjo vaših studijskih RGB luči za notranjo uporabo.