**Šolski center Novo mesto   
Srednja elektro šola in tehniška gimnazija   
Šegova ulica 112   
8000 Novo mesto**

**ESP32 s kamero**(Izdelek oziroma storitev in zagovor)

Avtor: Žan Vencelj, R4b

Mentor: Rajko Premović, prof.

Novo mesto, april 2023

**POVZETEK**

Izdelal sem esp32 z kamero, ki z časovnim zamikom zajema fotografije in jih pošilja na strežnik, kjer si lahko fotografije ogledamo na spletni strani z mapo. Opisal bom uporabo (konfiguracija in nalaganje kode) in postopek izdelave mojega izdelka.

**Ključne besede:** esp, esp32cam, Arduino

**KAZALO**

[1 UVOD 1](#_Toc134554283)

[2 NAVODILA ZA UPORABO 2](#_Toc134554284)

[2.1 Konfiguracija kode 2](#_Toc134554285)

[2.2 Nalaganje kode 3](#_Toc134554286)

[2.3 Spletna stran 4](#_Toc134554287)

[3 IZDELAVA 5](#_Toc134554288)

[3.1 ESP koda 5](#_Toc134554289)

[3.1.1 Zajemanje slike 5](#_Toc134554290)

[3.1.2 Nalaganje slike 5](#_Toc134554291)

[3.1.3 Spletno upravljanje 6](#_Toc134554292)

[3.2 Spletna stran 7](#_Toc134554293)

[4 NADALJNJE RAZVIJANJE 8](#_Toc134554294)

[5 VIRI IN LITERATURA 9](#_Toc134554295)

**KAZALO SLIK**

[Slika 1 ESP32-Cam 1](#_Toc134554274)

[Slika 3 Board manager 3](file:///C:\Users\Vanted\Desktop\esp32cam-dokumentacija.docx#_Toc134554275)

[Slika 4 Board in port 3](#_Toc134554276)

[Slika 5 Login stran 4](#_Toc134554277)

[Slika 6 Dodajanje točk 4](#_Toc134554278)

[Slika 7 Spletna stran 7](#_Toc134554279)

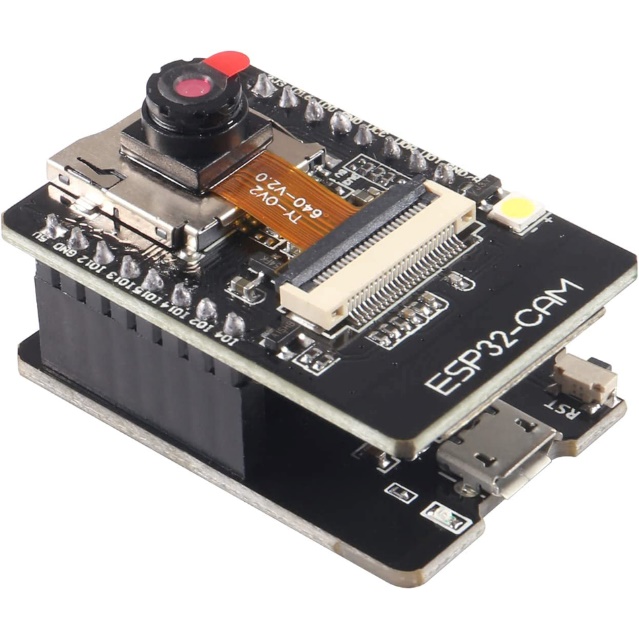
# UVOD

Cilj te seminarske naloge je predstaviti projekt, ki temelji na ESP32 napravi z vgrajeno kamero. Naprava omogoča zajemanje fotografij in njihovo prikazovanje na spletni strani. Gre za izjemno uporabno aplikacijo, saj omogoča oddaljen dostop in nadzor lokacije.

V prvem delu bom opisal kako se pripravi, uporablja napravo in spletno stran. V drugem delu pa, kako sem izdelal kodo oz. kako se jo prilagaja. Na koncu bom pisal še o nadaljnjih idejah o razvijanju projekta.

Seznam komponent uporabljenih za izdelavo projekta:

* ESP32-Cam z OV2640 modulom



Slika 1 ESP32-Cam

# NAVODILA ZA UPORABO

Dodajanje knjižnic

Za delovanje potrebujemo v Arduino IDE-ju naložene vse priložene bližnjice. To naredimo tako da jih z »Sketch > Include Library > Add .ZIP library…« dodamo.

## Konfiguracija kode

Najprej v datoteki nastavimo Wi-Fi na katerega se bo esp povezal ob zagonu. To nastavimo z spremenljivkama »ssid« in »password«.

Potem nastavimo še podatke FTP strežnika na katerega želimo pošiljati slike z spremenljivkami »ftp\_server« za ip, »ftp\_username« za FTP uporabnika in »ftp\_password« za geslo uporabnika.

Potem z spremenljivko »name« nastavimo ime naprave oz. ime datoteke, ki bo poslana na FTP stežnik in določimo način zajemanja slik – če je spremenljivka »timed« true bo pošiljanje samodejno izvedeno po določenem časovnem zamiku ali ob kliku na tipko »Capture photo« če je spremenljivka false (v tem primeru moramo še omogočiti spletno stran, tako da odkomentiramo kodo, kjer se strežnik prižge – na koncu setup() funkcije).

// Replace with your network credentials

const char\* ssid = "Zan Wi-Fi";

const char\* password = "sodd1234";

char ftp\_server[] = " <ip strežnika> ";

char ftp\_username[] = " <uporabnik> ";

char ftp\_password[] = " <geslo> ";

const String name = " <ime datoteke> ";

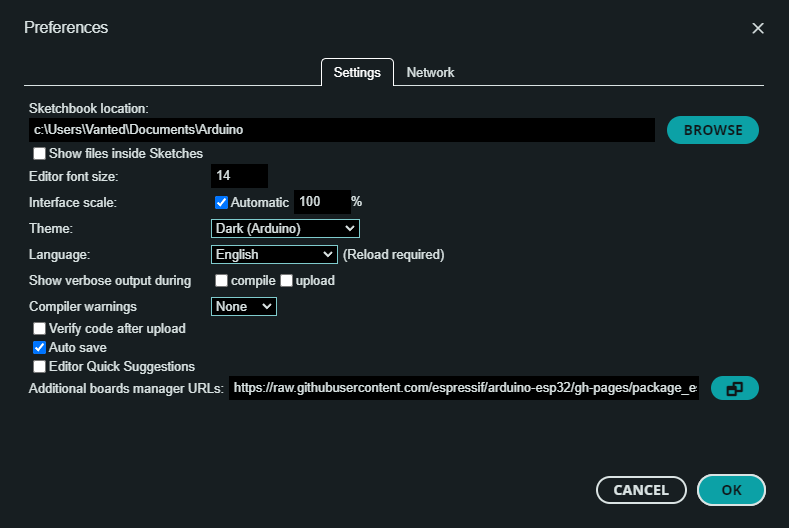
bool timed = true;

Z tem smo zaključili vso konfiguracijo, ki jo moramo opraviti znotraj kode, preden jo naložimo na napravo.

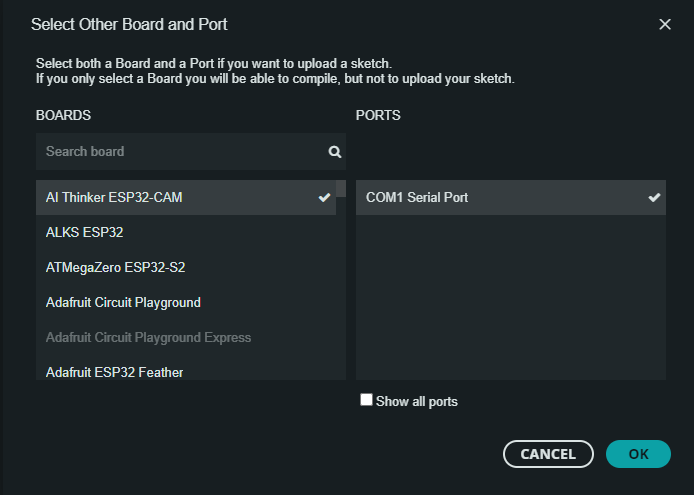
## Nalaganje kode

Najprej na napravi naložimo vse priložene knjižnice in dodamo board manager za ESP32-Cam. To naredimo tako da v File>Preferences>Additional boards manager URLs prilepimo povezavo: »https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package\_esp32\_index.json«.

Slika 3 Board manager



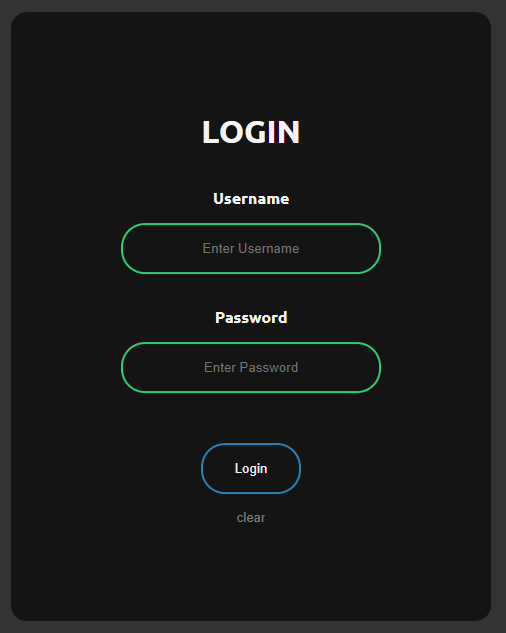
Nato med napravami izberemo »AI Thinker ESP32-CAM« in port na katerem imamo povezan esp. Kodo lahko nato preverimo in naložimo.



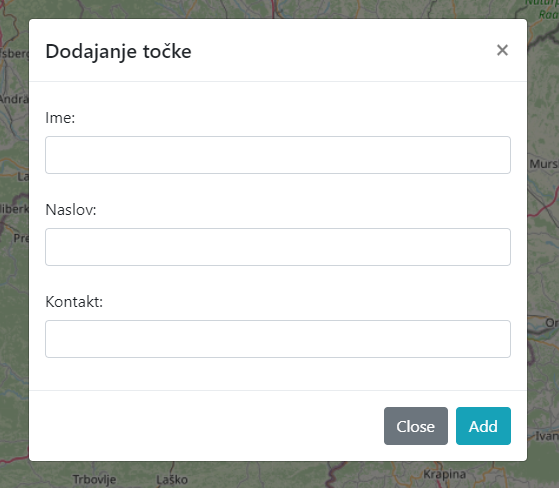
Slika 4 Board in port

## Spletna stran

Na spletni strani dodamo točke tako, da se najprej prijavimo na podstrani »/login«. In izberemo točko na mapi, kjer želimo novo lokacijo. V okno ki se odpre vpišemo vse zahtevane podatke in dodamo točko.



Slika 5 Login stran



Slika 6 Dodajanje točk

Brišemo jih tako, da tiščimo še tipko »Shift« in pritisnemo na točko, ki jo želimo zbrisati.

# IZDELAVA

## ESP koda

### Zajemanje slike

  fb = esp\_camera\_fb\_get();

  if (!fb) {

    Serial.println("Camera capture failed");

    return;

  }

Zgornja koda zajame fotografijo in izpiše napako v primeru, da zajemanje ni uspelo.

  File file = SPIFFS.open(FILE\_PHOTO, FILE\_WRITE);

Ustvarimo datoteko, v katero se bo zapisala slika.

  file.write(fb->buf, fb->len);

  Serial.print("The picture has been saved in ");

  Serial.print(FILE\_PHOTO);

  Serial.print(" - Size: ");

  Serial.print(file.size());

  Serial.println(" bytes");

V datoteko vnesemo zajete podatke (bite).

  file.close();

Na koncu mora biti datoteka obvezno zaprta.

  ok = checkPhoto(SPIFFS);

Z to kodo preveriš če je slika pravilno zajeta in shranjena.

### Nalaganje slike

ftp.OpenConnection();

Serial.println("FTP Client connected");

Vzpostavi se povezava z ftpjem.

ftp.ChangeWorkDir("httpdocs/esp/express/uploads");

Nastavimo ciljno mapo na strežniku.

ftp.InitFile("Type I");

ftp.NewFile("esp-test.jpg");

ftp.WriteData( fb->buf, fb->len );

ftp.CloseFile();

Nastavimo vrsto zapisovanja (Type I je za binarno), ime datoteke, podatke zapišemo iz bufferja in zapremo(shranimo) datoteko.

### Spletno upravljanje

Če želimo slike namesto z časovnim zamikom zajemati ob kliku na spletni strani, ki nam omogoča tudi ogled slike po zajetju moramo dodati še spodnje korake.

server.begin();

Z begin() funkcijo, ki je zakomentirana na koncu setup funkcije zaženemo spletno stran preko katere bomo imeli vpogled v sliko in možnost aktiviranja funkcij z klikom namesto z časovnim zamikom.

const char index\_html[] PROGMEM = R"rawliteral()rawliteral";

V to spremenjivko vstavimo html spletne strani, ki bo na esp-ju

server.on("/", HTTP\_GET, [](AsyncWebServerRequest \* request) {

    request->send\_P(200, "text/html", index\_html);

  });

  server.on("/capture", HTTP\_GET, [](AsyncWebServerRequest \* request) {

    takeNewPhoto = true;

    request->send\_P(200, "text/plain", "Taking Photo");

  });

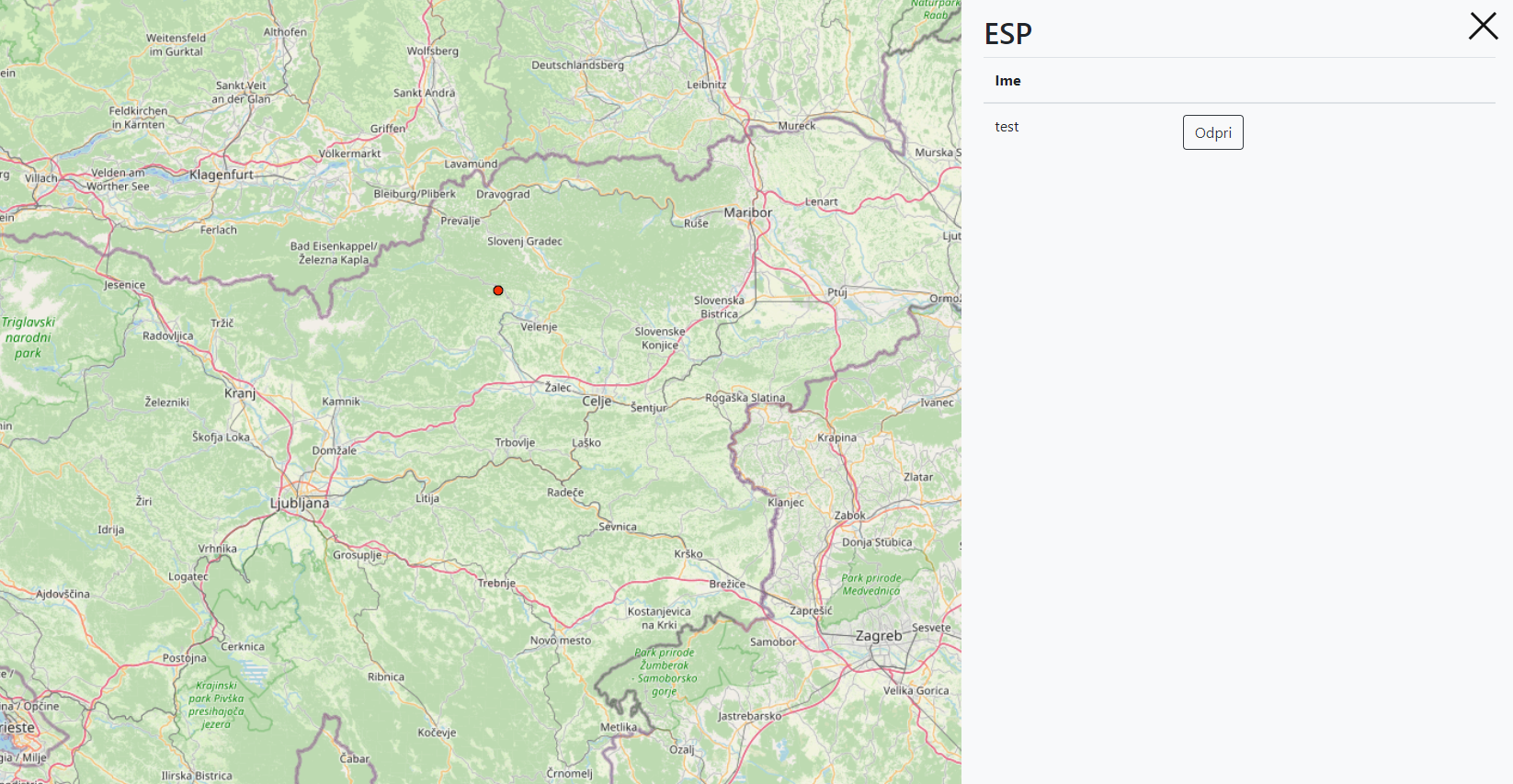
  server.on("/saved-photo", HTTP\_GET, [](AsyncWebServerRequest \* request) {

    request->send(SPIFFS, FILE\_PHOTO, "image/jpg", false);

  });

Za tipke nastavimo funkcije, ki bodo izvedene ko esp prejme http GET request. Uporabljajo se za odpiranje strani, zajemanje slike in pridobivanje slike z spomina na spletno stran.

## Spletna stran

Na indexu je zemljevid z točkami, kjer imamo postavljene esp-je. Na desni strani je seznam vseh lokacij, ki jih lahko odpremo in si ogledamo fotografije. 

Slika 7 Spletna stran

Mapa je narejena z javascript knjižnico OpenLayers.

Markerje se dodaja z funkcijo addMarker(ime, x, y). Mapo osvežimo z updateMap(). Pozicijo pogleda pa nastavimo z setCameara(x, y).

Če je uporabnik prijavljen se preveri z piškotki, tako preverimo tudi uporabniško ime z $\_COOKIE[username].

# NADALJNJE RAZVIJANJE

V prihodnje bi spremenil najprej to, da bo spletna stran narejena z Vue.js namesto z PHP, ker Vue ponuja hitrejše delovanje in je bolj prilagodljiv. Mapa bi bila narejena z Googlovim zemljevidom namesto OpenLayers.

Namesto pošiljanja slike preko SFTP povezave bi lahko naredil pošiljanje preko POST requesta na express strežnik, ki sem ga že pripravil v Node.js-u.

# VIRI IN LITERATURA

1. Random nerd tutorials. Pridobljeno 2022 s <https://randomnerdtutorials.com/esp32-cam-take-photo-display-web-server/>
2. Github. Pridobljeno 2022 s <https://github.com/ldab/ESP32_FTPClient>
3. Github. Pridobljeno 2022 s <https://github.com/espressif/esp32-camera>
4. OpenLayers. Documentation. Pridobljeno 2022 s <https://openlayers.org/doc/>