

Fakulteta za elektrotehniko,  
računalništvo in informatiko

Smetanova ulica 17

2000 Maribor, Slovenija

|  |  |
| --- | --- |
| ***Predmet: Projekt*** | |
| *Nosilec predmeta:* | Ernest Gungl |
| *Asistent:* | Ernest Gungl |
| *Naziv predmeta:* | **Projekt** |

|  |  |
| --- | --- |
| **Kandidat(ka)** | |
| *Vpisna številka:* | E5036871 |
| *Priimek in ime:* | Enej Unterajter |
| *Datum:* | 14.1.2020 |

Vsebina

[Uvod 5](#_Toc30295150)

[Specifikacije 6](#_Toc30295151)

[Raspberry Pi 7](#_Toc30295152)

[Priprava Raspberry Pi za projekt 7](#_Toc30295153)

[ESP8266 11](#_Toc30295154)

[Priprava ESP8266 za projekt 12](#_Toc30295155)

[MQ-135 13](#_Toc30295156)

[DHT22 15](#_Toc30295157)

[LED zaslon 16](#_Toc30295158)

[Podatkovna baza ESP8266 in senzorjev 17](#_Toc30295159)

[Izdelava okvirja 18](#_Toc30295160)

[Izdelava projekta po tednih 21](#_Toc30295161)

[Teden 1: 21](#_Toc30295162)

[Teden 2: 21](#_Toc30295163)

[Teden 3: 21](#_Toc30295164)

[Teden 4: 21](#_Toc30295165)

[Teden 5: 21](#_Toc30295166)

[Teden 6: 21](#_Toc30295167)

[Teden 7: 21](#_Toc30295168)

[Teden 8: 21](#_Toc30295169)

[Teden 9: 21](#_Toc30295170)

[Teden 10: 21](#_Toc30295171)

[Teden 11: 21](#_Toc30295172)

[Teden 12: 21](#_Toc30295173)

[Teden 14: 22](#_Toc30295174)

[Teden 15: 22](#_Toc30295175)

[Teden 16: 22](#_Toc30295176)

[Cena projekta 23](#_Toc30295177)

[Primerjava 24](#_Toc30295178)

[Funkcije pametnega ogledala projekta 24](#_Toc30295179)

[Pametno ogledalo primer 1 24](#_Toc30295180)

[Pametno ogledalo primer 2 24](#_Toc30295181)

[Slike končnega izdelka 25](#_Toc30295182)

[Koda 28](#_Toc30295183)

[Viri 32](#_Toc30295184)

**KAZALO SLIK**

[Slika 1:skica povezanih komponent in osnovno delovanje 5](#_Toc30295185)

[Slika 2:prikaz operacijskega sistema Raspbian 6](#_Toc30295186)

[Slika 3:prikaz rezervacije IP naslova RPi 6](#_Toc30295187)

[Slika 4:prikaz modula MMM-DarkSkyForecast 7](#_Toc30295188)

[Slika 5:prikaz testnega obvestila na LCD zaslonu 8](#_Toc30295189)

[Slika 6:prikaz delovanja obvestil MMM-PushBulletNotifications 8](#_Toc30295190)

[Slika 7:končen prikaz delovanja aplikacije MagicMirror 9](#_Toc30295191)

[Slika 8:prikaz mikrokmilnika ESP8266 10](#_Toc30295192)

[Slika 9:prikaz razporeditev "pinov" ESP8266 10](#_Toc30295193)

[Slika 10:prikaz rezervacije IP naslova ESP8266 11](#_Toc30295194)

[Slika 11:prikaz odzivna ESP8266 11](#_Toc30295195)

[Slika 12:prikaz povezave MQ-135 na ESP8266 12](#_Toc30295196)

[Slika 13:prikaz delovanja grafa senzorja MQ-135 12](#_Toc30295197)

[Slika 14:prikaz povezave DHT22 na ESP8266 14](#_Toc30295198)

[Slika 15:prikaz povezave LED zaslona na ESP8266 15](#_Toc30295199)

[Slika 16:skica celotne povezave 16](#_Toc30295200)

[Slika 17:prikaz podatkovne baze Firebase 16](#_Toc30295201)

[Slika 18:prikaz skice izdelave okvirja 17](#_Toc30295202)

[Slika 19:izdelava okvirja 18](#_Toc30295203)

[Slika 20:izdelava zrcala 18](#_Toc30295204)

[Slika 21:sestavljanje projekta 19](#_Toc30295205)

[Slika 22:končen sestavljen projekt 19](#_Toc30295206)

[Slika 23:prikaz celotnega končnega ogledala v temi 24](#_Toc30295207)

[Slika 24:prikaz datuma in ure 24](#_Toc30295208)

[Slika 25:prikaz vremena in vremenske napovedi 25](#_Toc30295209)

[Slika 26:prikaz obvestil iz mobilnega telefona 25](#_Toc30295210)

[Slika 27:prikaz temperature in vlage 26](#_Toc30295211)

[Slika 28:prikaz celotnega ogledala ob nočni svetlobi 26](#_Toc30295212)

# Uvod

Pri projektu smo se odločili izdelati pametno ogledalo. Ker je na trgu že nekaj pametnih ogledal, katerih cena je visoka, funkcije samega ogledala pa omejene, smo se odločili da bomo izdelali svoje ogledalo, pri čemer smo upoštevali ceno izdelave, da naredimo cenejše pametno ogledalo z vsemi funkcijami ki bi lahko bile koristne za uporabnika. Za izdelavo smo uporabili Raspberry Pi, ESP8266, 2 senzorja (MQ-135 in DHT22), mali LED zaslon, velik LCD zaslon, microSD karico, VGA-HDMI pretvornik, les in ceneno pleksi steklo ki je služilo kot zrcalo, električni razdelilec ter vijake. Danes nas komunikacijske naprave obkrožajo povsod zato smo izdelali celoten produkt pametnega ogledala za domačo uporabo.

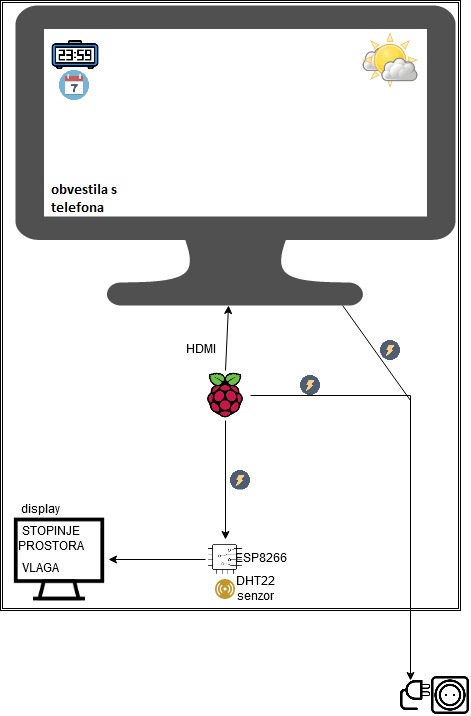
# Specifikacije

Uporabili smo Raspberry Pi 3 Model B+, NodeMCU ESP8266 – skupaj s senzorjem kvalitete zraka MQ-135, senzorjem temperature in vlage DHT22 in malim LED zaslonom ter veliki LCD zaslon.

Plošček Raspberry Pi je povezan preko HDMI kabla na velik LCD zaslon in napajanje. Prikazuje osrednje informacije pametnega zrcala (ura, datum, vreme in vremensko napoved, obvestila s telefona – operacijskega sistema android). V njega smo tudi vstavili microSD kartico (ki služi kot trdi disk oz. je namenjen shranjevanju).

Na mikrokrmnilnik ESP8266 smo povezali na Raspberry Pi (preko napajalnega kabla, za napajanje), senzor kvalitete zraka MQ-135, senzor vlage in temperature DHT22, mali LED zaslon (za prikaz vlage in temperature).

Komponente smo priključili po spodnji skici:



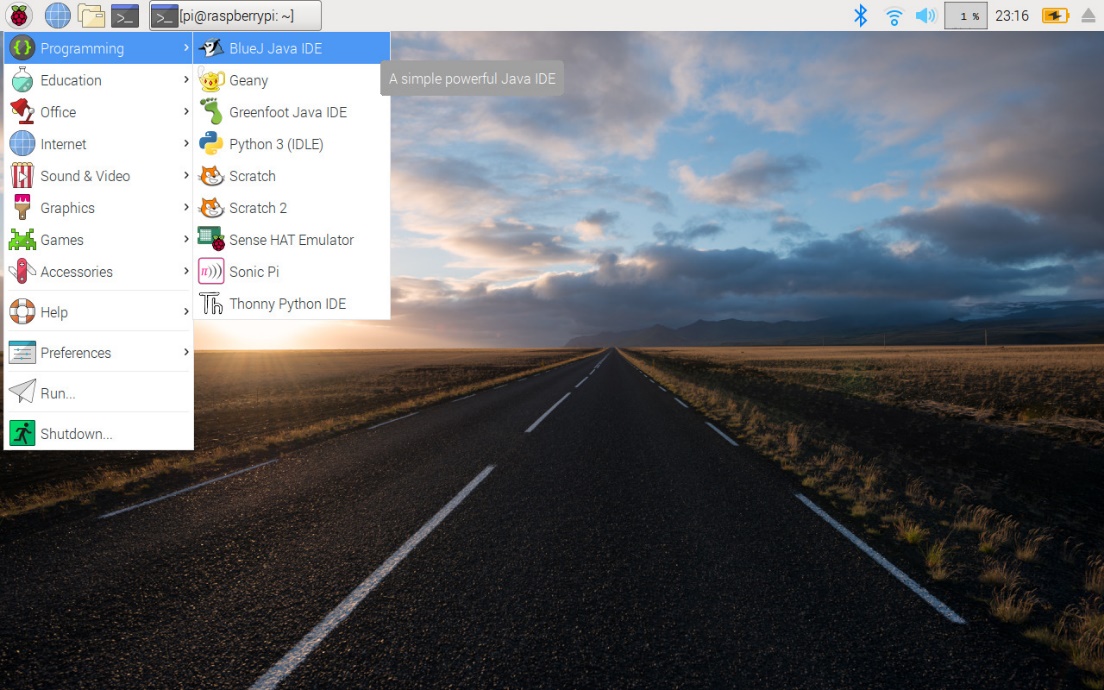
Slika 1:skica povezanih komponent in osnovno delovanje

# Raspberry Pi

Raspberry Pi je mali računalnik, razvit s strani Raspberry Pi Foundation, ki je namenjen za učenje osnov računalništva in je namenjen razvijanju različnih projektov, lahko tudi za komercialno uporabo. Raspberry Pi (RPi) je doživel velik razcvet, saj lahko nanj priključimo miško, tipkovnico, senzorje ipd.

## Priprava Raspberry Pi za projekt

Na našo MicroSD kartico velikosti 16GB smo »*fleshali«* Raspbian operacijski sistem, katerega smo prenesli iz njihove uradne spletne strani: <https://www.raspberrypi.org/downloads/raspbian/>. Postopek smo izvedli s pomočjo programa »*rufus«* katerega smo prenesli in namestili na naš računalnik iz njihove uradne spletne strani: <https://rufus.ie/>. Pripravljeno MicroSD kartico smo vstavili v Raspberry Pi in uredili osnovnega korake (določili državo, jezik, časovni pas, kreirali osnovna gesla itd.). Ko smo uredili osnovne nastavitve smo operacijski sistem posodobili s ukazoma *sudo apt-get update* in *sudo apt-get upgrade*. Za boljše in zanesljivejše delovanje smo na našem usmerjevalniku tudi rezervirali IP naslov, ko RPi povezali na naše Wi-Fi omrežje (b8:27:eb:32:05:e3| 192.168.1.10).



Slika 2:prikaz operacijskega sistema Raspbian



Slika 3:prikaz rezervacije IP naslova RPi

Nato smo še na RPi naredili oddaljen dostop, preko terminala smo na RPi pognali ukaz *»enable SSH«*, tako se lahko v lokalni mreži s katerokoli napravo povežemo na RPi preko ukaza *»ssh* [*pi@192.168.1.10«*](mailto:pi@192.168.1.10). Tako lahko naše zrcalo oddaljeno dosežemo in spreminjamo, ne da bi ga razstavljali za različne spremembe. Uporabniško ime in geslo smo pustili privzeto *(username: pi, password: raspberry)*.

Na naše zrcalo smo še prenesli in namestili MagicMirror, ki je služil kot centralen prikaz. MagicMirror je odprtokodna, brezplačna modularna platforma, ki omogoča izdelavo pametnih ogledal.

MagicMirror smo na RPi namestili s ukazom: *bash -c "$(curl -sL* [*https://raw.githubusercontent.com/MichMich/MagicMirror/master/installers/raspberry.sh)*](https://raw.githubusercontent.com/MichMich/MagicMirror/master/installers/raspberry.sh))*"*. Ko smo namestitev dokončno izvedli, smo se postavili v namestitveno mapo *(cd ~/MagicMirror)* in ga zagnali s *DISPLAY=:0 npm start*. Želeli smo še da se MagicMirror program izvede takoj, ko se vklopi RPi v napajanje in se naloži, to smo dosegli s ukazom *pm2 startup*. Za urejanje modulov v programu MagicMirror smo lahko vedno dostopali do datotek najdenih v *cd ~/MagicMirror* preko oddaljenega dostopa ali pa direktno na RPi s pomočjo tipke *ctrl* ki omogoča *minimizacijo* programa, da lahko dostopamo do datotek.

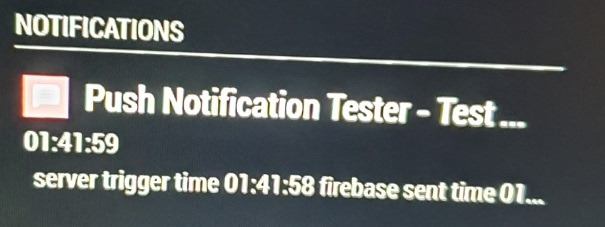
Ko smo naš program pognali, so se nam prikazale osnovne nastavitve (prednastavljene za Združene države Amerike). Videli smo njihov časovni pas (ura, datum) in koledar s njihovimi prazniki. V datotekah *MagicMirror* smo nastavili pravilen časovni pas za našo državo in izbrisali *modul calendar*. Program *MagicMirror* nam je sedaj prikazoval pravilen čas in datum za našo državo.

Na naš RPi smo še dodali dodaten modul imenovan *MMM-DarkSkyForecast*, katerega smo prenesli iz uradne spletne strani *github* na povezavi: <https://github.com/jclarke0000/MMM-DarkSkyForecast>. Modul privzeto vzame državo in časovni pas iz operacijskega sistem *Raspbian*, zato teh nastavitev ni bilo potrebno spreminjati. V datotekah modula *MMM-DarkSkyForecast* smo še spremenili ikone, da je izgled lepši in ga postavili v zgornji desni kot na velikem LCD zaslonu. Modul *MMM-DarkSkyForecast* prikazuje trenutno vreme, stopinje in vremensko napoved za ostale dni.

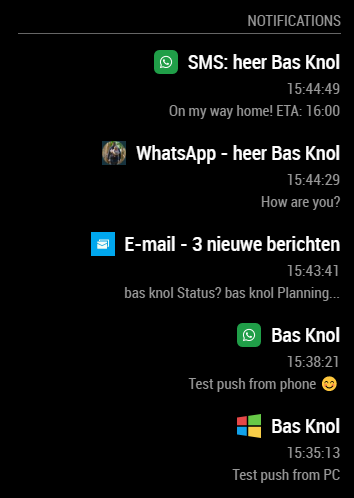


Slika 4:prikaz modula MMM-DarkSkyForecast

V projektu smo še želeli dodati obvestila telefona. Uporabili smo telefon operacijskega sistema Android, model Samsung Galaxy S10+. Ker na svojem mobilnem telefonu uporabljam aplikacijo *PushBullet* hiter prenos datotek med telefonom in računalnikom, ter shranjevanje datotek v oblaku, sočasno pa še prikazuje vsa obvestila mobilnega telefona na osebnem računalniku kadar mobilnega telefona ni v bližini, ali pa ni v uporabi, smo se odločili da bomo implementirali modul prikazovanja obvestil s pomočjo aplikacije *PushBullet*. Tako smo iz uradne spletne strani namestili nov modul za program *MagicMiroor* imenovan *MMM-PushBulletNotifications*, ki prikazuje obvestila s pomočjo *PushBullet* API-ja: <https://github.com/basknol/MMM-PushBulletNotifications>. Ko smo modul namestili smo se postavili v datoteke modula *MMM-PushBulletNotifications* in spremenili *AccessToken*, katerega smo našli v nastavitvah aplikacije *PushBullet* na mobilnem telefonu. Modul *MMM-PushBulletNotifications* smo še postavili v spodnji levi kot.

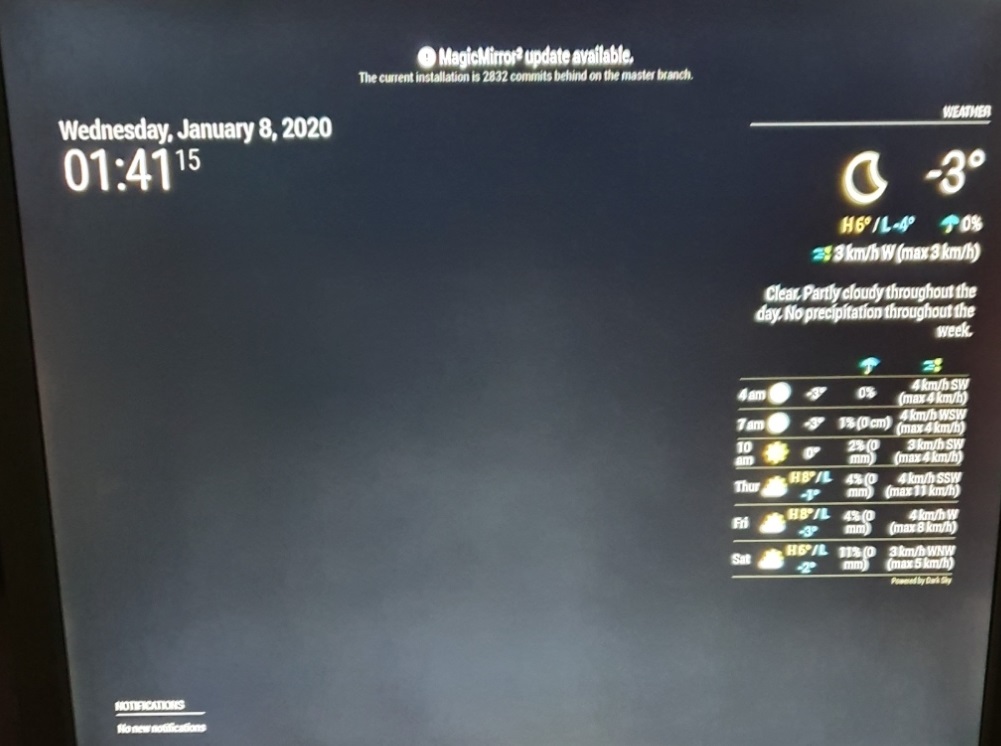


Slika 5:prikaz testnega obvestila na LCD zaslonu



Slika 6:prikaz delovanja obvestil MMM-PushBulletNotifications

S tem smo zaključili del projekta ki je vključeval RPi, na spodnji sliki je še prikaz končno delovanje RPi in programa *MagicMirror*.



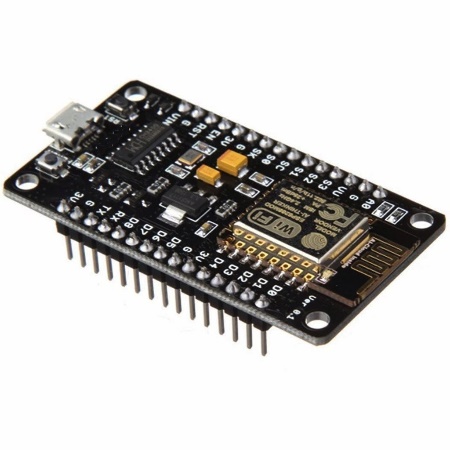
Slika 7:končen prikaz delovanja aplikacije MagicMirror

# ESP8266

NodeMCU ESP8266 je nizkocenovni mikročip/mikrokrmilnik ki omogoča povezavo na Wi-Fi kot tudi priključitev dodatnih zunanjih komponent omogoča široko uporabo in zelo dobro delovanje za ceneni mikrokrmilnik.

Osnovne specifikacije mikrokrmilnika:

* Povezava na Wi-Fi (standard 802.11 b/g/n)
* 32-bit RISC CPU
* Delovna napetost: 3V3
* delovna frekvenca: 80 MHz
* 64 KB RAM program(razširljiv z Flash)
* 96 KB RAM podatki
* Zunanji "flash" - 512 KB do 4 MB
* Strojna podpora: SPI, I2C, I²S, UART
* ADC: 1x 10-bit
* 16 GPIO priključkov



Slika 8:prikaz mikrokmilnika ESP8266



Slika 9:prikaz razporeditev "pinov" ESP8266

## Priprava ESP8266 za projekt

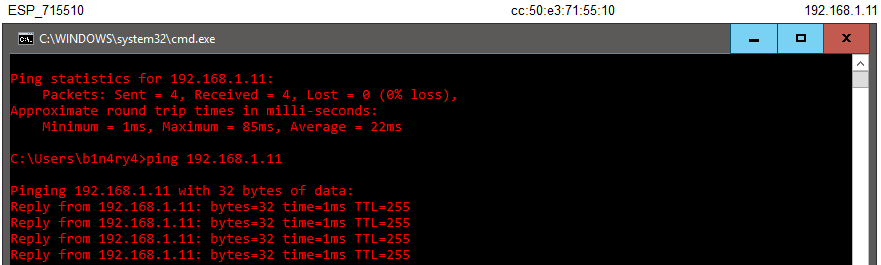
Za programiranje mikrokrmilnika ESP8266 smo uporabili Arduino. Na osebni računalnik smo namestili Arduino IDE, katerega smo prenesli iz uradne spletne strani <https://www.arduino.cc/en/main/software>. Na naložen in nameščen program smo namestili ESP8266 kot mikrokrmilnik in namestili vse potrebne knjižice. Za senzorjev smo namestili OLED knjižice *Adafruit SSD1306 library* in *Adafruit GFX library*. Namestili smo še knjižice za DHT22 senzor in MQ-135 senzor.

Za začetek smo ESP8266 povezali na Wi-Fi, na našem usmerjevalniku smo rezervirali IP naslov (cc:50:e3:71:55:10|192.168.1.11) kot je prikazano na sliki spodaj:



Slika 10:prikaz rezervacije IP naslova ESP8266

Pogledali smo na naš usmerjevalnik, ali je ESP8266 v DHCP dobil IP-naslov. Nato smo pognali test *»ping«* da smo ugotovili ali je naprava odzivna, dobili smo rezultat ki je prikazan na spodnji sliki:



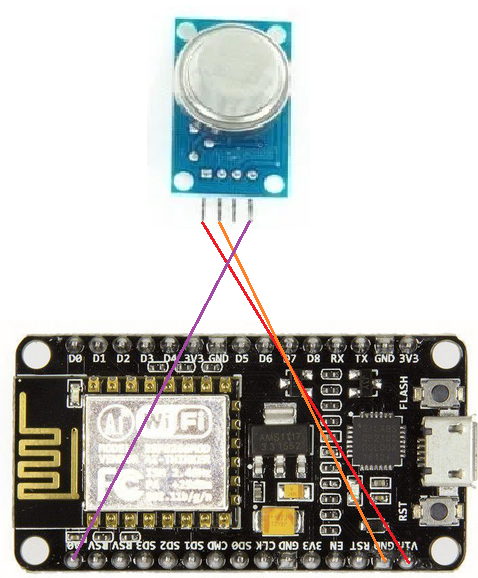
Slika 11:prikaz odzivna ESP8266

In tako zaključili da se je ESP8266 uspešno povezal v omrežje.

## MQ-135

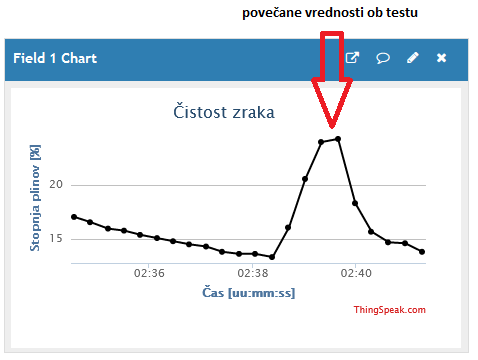
Prvo smo na EPSP8266 povezali senzor kvalitete zraka MQ-135. MQ-135 smo povezali na ESP8266 na naslednje *»pine«*: A0 na A0, GND na GND in VCC n Vin.

Povezavo smo naredili kot je prikazano na spodnji skici:



Slika 12:prikaz povezave MQ-135 na ESP8266

Ko smo vse priključili in programsko uredili smo testirali delovanje. MQ-135 senzor smo prvo testirali v terminalu, ko smo ugotovili da vse deluje smo ga povezali s spletno stranjo: <https://thingspeak.com/> ki deluje kot *»broker«* in izriše graf delovanja oz. vrednosti senzorja MQ-135. Senzor smo testirali tako, da smo s elektronsko cigareto pihali v senzor in gledali vrednosti ali narastejo (glede na gostoto dima iz elektronske cigarete morajo vrednosti narasti). Na spodnji sliki je prikazano delovanje senzorja MQ-135 in graf delovanja:



Slika 13:prikaz delovanja grafa senzorja MQ-135

MQ-135 je naprava ki zazna prisotnost plinov v ozračju. Uporablja se predvsem za zaznavanje plinov v primeru puščanja plinov in drugih emisij. V našem primeru smo senzor uporabili za zaznavanje gostote plinov v prostoru, katera narast lahko vpliva tudi na človeško zdravje. Senzor se še lahko uporablja za alarmno obveščanje prevelike gostote plinov, izklop kakšnega sistema v primeru večje gostote in podobno. MQ-135 senzor deluje tako, da prevodnost naraste, kadar se koncentracija plinov poveča.

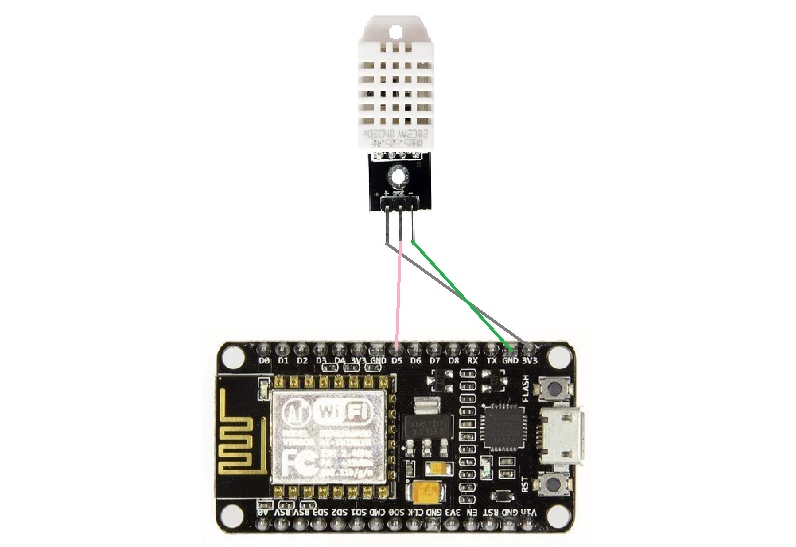
Specifikacije senzorja MQ-135:

* Zaznava plinov
* Zaznava koncentracij 10-1000ppm
* Vhodno napajanje DC 5V
* Poraba energije (tok): 150mA
* A0 izhod: 0.1-0.3V (najvišja koncentracija toka je 4V)
* Hitra odzivnost
* Lahko se uporablja kot digitalni ali analogni senzor (v našem primeru kot analogni)

## DHT22

Nato smo na ESP8266 povezali senzor temperature in vlage DHT22. Senzor DHT22 smo na ESP8266 povezali na naslednje *»pine«*: + na 3V3, OUT na D5 in – na GND.

Povezavo smo naredili kot je prikazano na spodnji skici:



Slika 14:prikaz povezave DHT22 na ESP8266

Ko smo vse priključili in programsko uredili smo testirali delovanje. DHT22 senzor smo prvo testirali v terminalu, testirali smo ga tako, da smo vanj pihali in analizirali vrednosti, ob pihanju v njega je vlaga narastla, kot tudi temperatura. Testirali smo še ga na različnih lokacijah kjer smo spremljali temperaturo. Iz rezultatov testiranja smo zaključili da je delovanje senzorja optimalno.

DHT22 je senzor temperature in vlage. Gre za osnovni senzor ki je počasen, vendar je poceni in dobro opravlja svoje delo za delovanje našega projekta. Senzor vsebuje osnovni čip, kateri omogoča analogno – digitalno pretvorbo in na izhodu poda digitalni podatek temperature in vlage. Obstajata dva različna senzorja, DHT11 in DHT22 – v našem primeru smo se odločili uporabiti malo dražji senzor DHT22 zaradi večje natančnosti in malo boljšem delovanju na daljši razdalji.

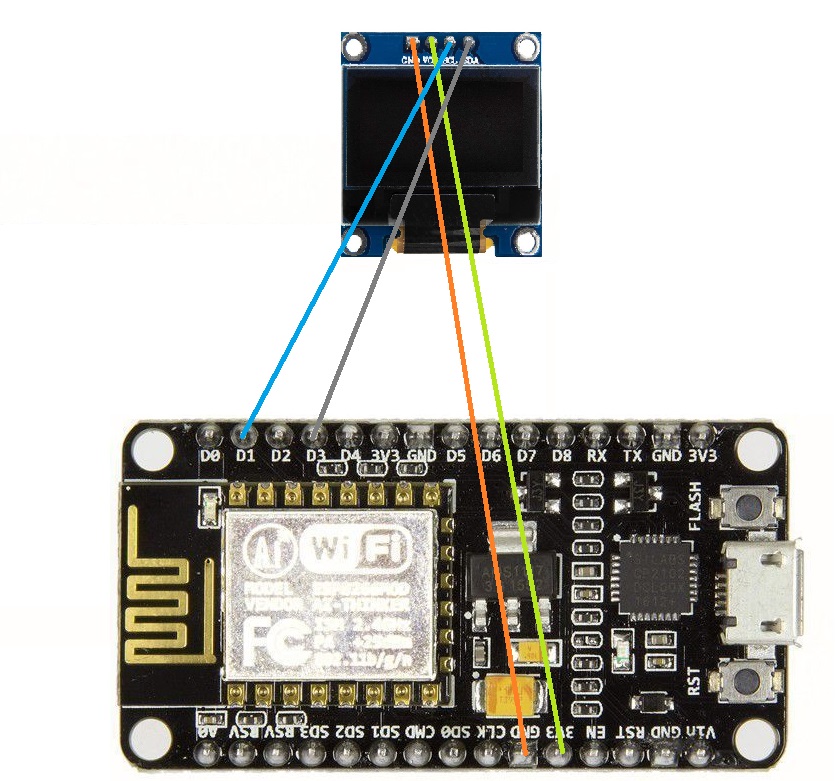
Specifikacije senzorja DHT22:

* Nizkocenovni senzor
* Območje delovanja 3 – 5V
* Podatek vlage (0 do 100%, kjer lahko pride do odstopanj 2 – 5%)
* Podatek temperature (-40 do 80°C, kjer lahko prihaja do odstopanj +-0.5°C)
* Digitalen signal (podatek)

## LED zaslon

V nadaljevanju vaj smo še na ESP8266 povezali mali LED zaslon, kateri je služil za prikaz temperature in vlage. LED zaslon smo priključili na naslednje *»pine«*: GND na GND, VCC na 3V3, SOL na D1 in SDA na D3.

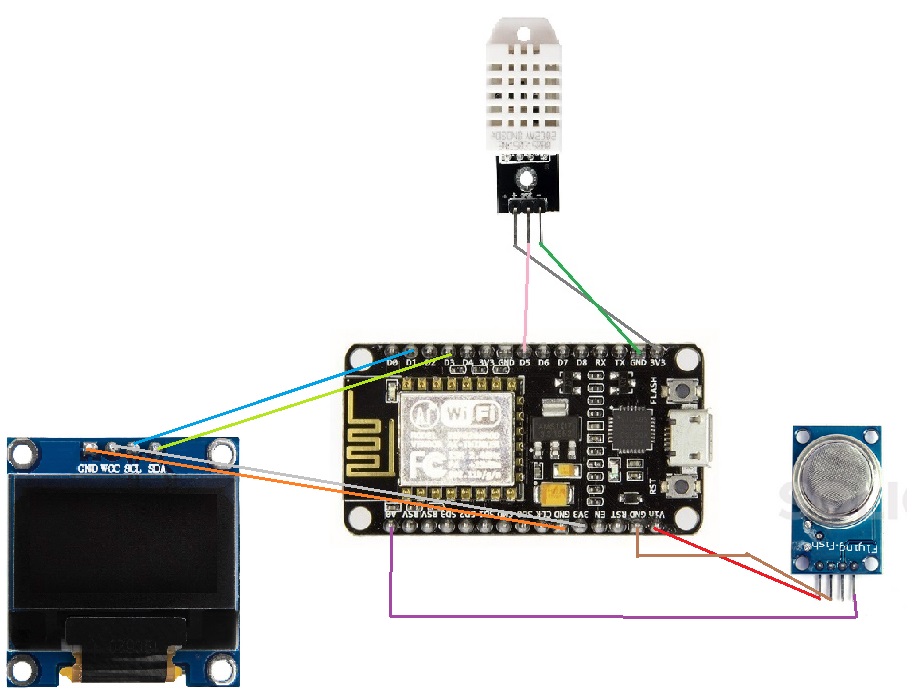
Povezavo smo naredili kot je prikazano na naslednji sliki:



Slika 15:prikaz povezave LED zaslona na ESP8266

Ko se nam je na senzorju prikazala trenutna vlaga in temperatura, smo še delovanje testirali. Testirali smo trenutno temperaturo v zaprtem prostoru, temperaturo ob odprtem oknu (kjer je temperatura padala) ter testirali naraščanje temperature in vlage s pihanjem v senzor.

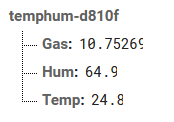
LED zaslon je veliko – kontrastni zaslon velikosti 0.96 inch. Njegova resolucija znaša 128 (v širino) in 64 (v višino) *»pixlov«*. Za komunikacijo uporablja protokol *»I2C«*. LED zaslon smo na ESP8266 povezali preko *»arduino – ta«*.



Slika 16:skica celotne povezave

## Podatkovna baza ESP8266 in senzorjev

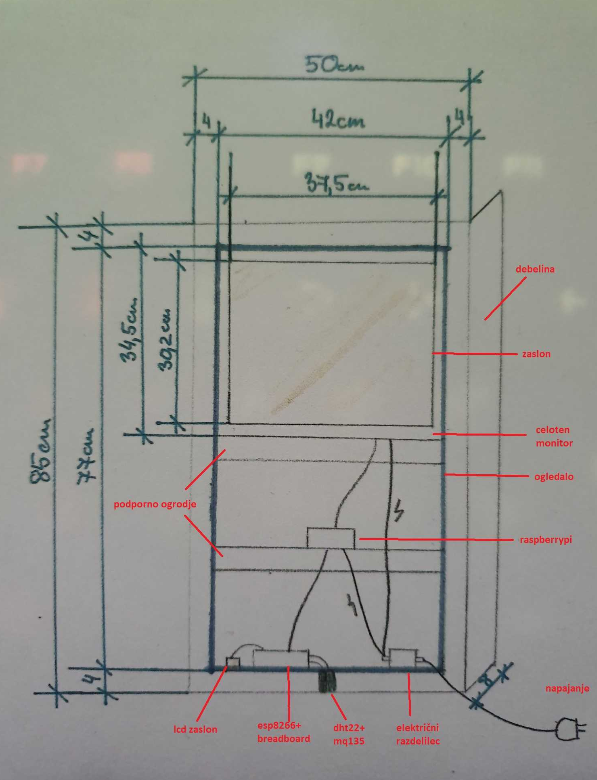
Ker bomo morda kdaj v prihodnosti še nadaljevali z delom projekta, smo se odločili implementirati podatkovno bazo. Za podatkovno bazo smo uporabili *Firebase*. Senzorja smo s pomočjo *Arduino* povezali s podatkovno bazo, na kateri so se nam prikazovale vrednosti v realnem času. V primeru da bi se kdaj odločili še narediti *Android* aplikacijo tako mikrokrmilnika ne bo potrebno razstavljati iz ogledala.



Slika 17:prikaz podatkovne baze Firebase

# Izdelava okvirja

Na začetku smo natančno izmerili vse komponente, naredili načrt postavitve in naredili načrt končnega izgleda pametnega zrcala, skupaj z vsemi meritvami.



Slika 18:prikaz skice izdelave okvirja

Okvir smo izdelali iz lesa, katerega smo utrdili z vijaki in lepilom za les. Med ogrodje okvirja smo še nato pritrdili dodatne lesene palice ki služijo za podporo strojne opreme.

V okvir smo zarezali zarezo v katero smo vstavili naše zrcalo.

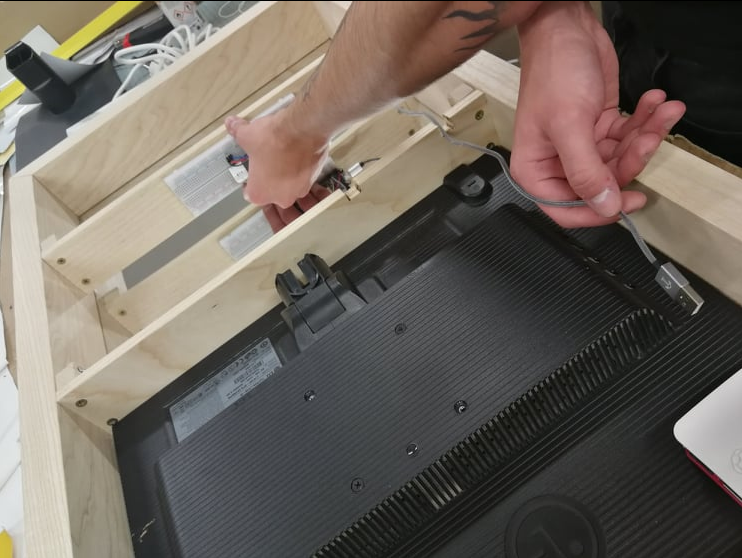
Na hrbtno stran smo nato vstavili monitor in priključili ter pritrdili vse komponente (Raspberry Pi, ESP8266 s senzorji in LED zaslonom, električni razdelilec, vse povezovalne kable,…). Komponente smo pritrdili s pomočjo gradbenega dvostranskega lepilnega traku in silikonom.



Slika 19:izdelava okvirja



Slika 20:izdelava zrcala



Slika 21:sestavljanje projekta



Slika 22:končen sestavljen projekt

# Izdelava projekta po tednih

## Teden 1:

Izbira tema, predstavitev ideje

## Teden 2:

Nakup microSD kartice, *»flashanje«* microSD kartice (Raspbian OS), test Raspbian OS, nakup pretvornika DAC (VHA->HDMI). Naročitev ESP8266, DHT22 in MQ-135 senzorja, LED zaslona.

## Teden 3:

Nastavitev osnovnih korakov Rasbian OS (jezik, gesla, časovni pas,…). Povezava RPi na Wi-Fi, nadgradnja celotnega sistema, rezervacija statičnega naslovna na modemu, namestitev potrebnih programov (*MagicMirror)*, nastavitev oddaljenega dostopa do RPi (preko SSH) – na ogledalo se lahko povežemo lokalno preko omrežja Wi-Fi, prikaz ure in datuma na zaslon.

## Teden 4:

Iskanje ogledala, primerjava cen, pregled cen, poštnin, pregled alternativ.

## Teden 5:

Naročanje zrcala, test zrcala. Povezava ESP8266 v omrežje, rezervacija IP naslova, test internetne povezave, priključitev MQ-135 senzorja na ESP8266.

## Teden 6:

Težave z DHT22 senzorjem, občasno je prebral podatke občasno pa ne – dogajanje naključno. Testiranje z drugimi povezovalnimi kabli, pregled kratkega stika, priključitev s in brez *»broadboarda«*, testiranje drugega USB kabla, drug USB vhod na računalniku, različnih *pinov* na ESP8266, napajanje 3.3V in 5V, testiranje s resistorjem.

## Teden 7:

Do sedaj je RPi prikazoval samo uro in datum, dodali smo še vreme (v stopinjah) in vremensko napoved ter prikazovanje obvestil iz *Android* mobilnega telefona. Naročanje dveh novih DHT22 senzorjev.

## Teden 8:

/

## Teden 9:

Test z drugim DHT22 senzorjem, težava je bila enaka. Najdena napaka v programski kodi, odpravljena, senzor DHT22 začel delovati.

## Teden 10:

Združitev programske kode za DHT22 in MQ-135. Izdelava natančnega načrta zrcala (meritve, velikost,…).

## Teden 11:

Na ESP8266 smo priključili mali LED zaslon, prikaz temperature in vlage v realnem času, podatki se osvežujejo vsakih 5 sekund.

## Teden 12:

Iskanje materiala za izdelavo okvirja.

## Teden 14:

MQ-135 senzor smo povezali s analitično platformo *ThingSpeak* in povezava senzorjev MQ-135 ter DHT22 s podatkovno bazo *Firebase*.

## Teden 15:

Izdelava okvirja, vgradnja komponent.

## Teden 16:

Izdelava poročila.

# Cena projekta

|  |  |  |
| --- | --- | --- |
|  | Moji ctroški [€] | Najnižja cena [€] |
| Električni razdelilec | Doma | 2.29 |
| ESP8266 | 4,66 | 2,51 |
| Breadboard | 1,17 | 1,17 |
| DHT22 | 2 \* 3,63 = 7,26 | 2,17 |
| MQ-135 | 1,35 | 1,35 |
| Breadboard kabli | 1,17 | 1,17 |
| Micro USB kabl | Doma | 1,17 |
| LED zaslon | 2,26 | 2,26 |
| RaspberryPi | Doma | 36,49 |
| microSD kartica | Doma | 3,25 |
| LCD monitor | Doma | 10 (rabljen) |
| RaspbberryPi napajalnik | Doma | 3,18€ |
| Zrcalo | 10 | 10 {50} |
| Les | 10 | 10 |
| SKUPAJ | 37,87 | 87,01 {127} |

Pri čemer najdemo ceno zapisano v »{ }« s optimalnim zrcalom, kjer bi vidnost bila še boljša.

# Primerjava

Za naš projekt smo okvirno odšteli 37,87€ - skupaj s komponentami ki smo jih imeli doma. Naš produkt bomo še primerjali s cenami različnih pametnih ogledal, katere smo našli na internetu ter primerjali njihove funkcije. Pri različnih primerih v »[ ]« najdemo funkcije ki jih naše pametno ogledalo nima.

## Funkcije pametnega ogledala projekta

* Prikaz temperature na zaslonu
* Prikaz vlage na zaslonu
* Možnost oddaljenega ogleda stopnje plinov
* Oddaljen ogled vlage in temperature preko podatkovne baze firebase
* Možnost oddaljenega dostopa do pametnega ogledala (RaspberryPi)
* Ura
* Datum
* Trenutna zunanja temperatura in vremenska napoved
* Prikaz obvestil iz mobilnega telefona

CENA: 37,87€ [z nakupom vse komponent ~ 127€]

## Pametno ogledalo primer 1

Pametno ogledalo pri primeru 1 ki ga bomo opisali spodaj, je tudi najcenejše pametno ogledalo, katerega smo našli na internetu.

* Slabost: manjše ogledalo
* Datum
* Čas
* Vreme
* [Promet – cestni, vlaki in podzemne]
* [Naslovi novic] – omogoča tudi naše ogledalo, vendar je zaradi malega prostora zaslona bilo odstranjeno
* [Koledar]
* [možnost glasovnega upravljanja (Angleščina)]
* [Zvočniki]

CENA: 210,06€ + 28,48€ (poštnina) = 238,54€

## Pametno ogledalo primer 2

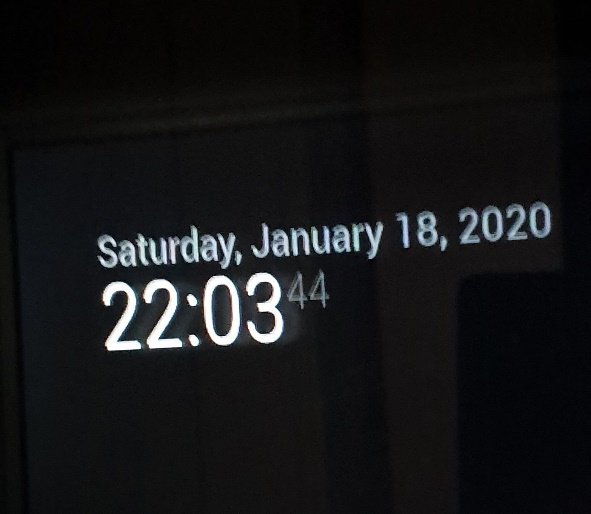
* Čas
* [Alarm]
* Vreme
* [Možnost spanja (v nočnem času se svetlost zmanjša zaradi varčevanja z energijo]
* Temperatura
* Vlaga
* Vremenska napoved
* [LED dioda kot indikator]

CENA: 230,01€ + 9,39€ (poštnina) = 239,40€

# Slike končnega izdelka



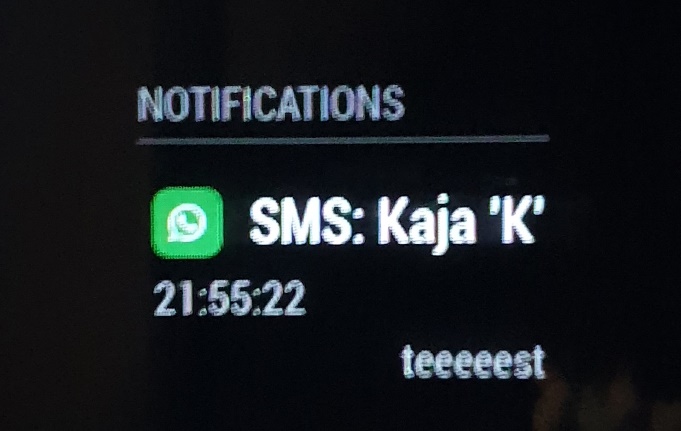
Slika 23:prikaz celotnega končnega ogledala v temi



Slika 24:prikaz datuma in ure



Slika 25:prikaz vremena in vremenske napovedi



Slika 26:prikaz obvestil iz mobilnega telefona



Slika 27:prikaz temperature in vlage



Slika 28:prikaz celotnega ogledala ob nočni svetlobi

# Koda

--knjižice--

#include <ESP8266WiFi.h>

#include <Wire.h>

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

#include <Adafruit\_Sensor.h>

#include <DHT.h>

#include "FirebaseArduino.h"

#define SCREEN\_WIDTH 128 // OLED display width, in pixels –definicija širinine LED zaslona--

#define SCREEN\_HEIGHT 64 // OLED display height, in pixels –definicija višine LED zaslona

#define FIREBASE\_HOST "temphum-d810f.firebaseio.com" –definicija povezave za firebase--

String apiKey = "VPPZHU198CX256MY"; --apiKey za ThingSpeak--

const char\* ssid = "Boobies ( . ) ( . )"; --ssid--

const char\* password = "3Racunal6Nicar0"; --geslo za Wi-Fi--

const char\* server = "api.thingspeak.com"; --server--

WiFiClient client; --client ki se lahko poveže na specifičen naslov--

Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, -1); --priprava zaslona--

#define DHTPIN 14 --digitalni DHT pin kjer je povezan senzor--

#define DHTTYPE DHT22 // DHT 22 (AM2302) –definicija DHT22 senzorja--

DHT dht(DHTPIN, DHTTYPE);

void setup() { --začetek funkcije--

Serial.begin(115200); --serial »baudrate«--

delay(10); --zamik--

Serial.println('\n'); --izpis--

Serial.println("Connecting to "); --izpis--

Serial.println(ssid); --izpis--

WiFi.begin(ssid, password); --povezava na Wi-Fi--

while (WiFi.status() != WL\_CONNECTED) { --izpis povezovanja na Wi-Fi--

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected"); --izpis ko se uspešno poveže na Wi-Fi--

dht.begin(); --začetek DHT22--

if(!display.begin(SSD1306\_SWITCHCAPVCC, 0x3C)) { --če se zaslon ne izvede--

Serial.println(F("SSD1306 allocation failed"));

for(;;);

}

delay(2000);

display.clearDisplay();

display.setTextColor(WHITE); --bel zaslon--

Firebase.begin(FIREBASE\_HOST); --začetek firebase strežnika--

}

void loop() { --zanka--

float a = analogRead(A0); --spremenljivka za analogni senzor MQ-135--

if (isnan(a)) {

Serial.println("Failed to read from MQ-5 sensor!");--v primeru neuspešnega branja--

return;

}

if (client.connect(server, 80)) { --povezava na »broker« thingspeak--

String postStr = apiKey;

postStr += "&field1=";

postStr += String(a/1023\*100);

postStr += "r\n";

client.print("POST /update HTTP/1.1\n");

client.print("Host: api.thingspeak.com\n");

client.print("Connection: close\n");

client.print("X-THINGSPEAKAPIKEY: " + apiKey + "\n");

client.print("Content-Type: application/x-www-form-urlencoded\n");

client.print("Content-Length: ");

client.print(postStr.length());

client.print("\n\n");

client.print(postStr); --pošiljanje na thingspeak--

Serial.print("Gas Level: ");

Serial.println(a/1023\*100);

Serial.println("Data Send to Thingspeak"); --izpis--

}

delay(500);

client.stop();

Serial.println("Waiting...");

delay(5000);

float t = dht.readTemperature(); --spremeljivki za vlago in temperaturo--

float h = dht.readHumidity();

if (isnan(h) || isnan(t)) {

Serial.println("Failed to read from DHT sensor!"); --v primeru napačnega branja--

}

display.clearDisplay(); --osvežitev zaslona--

display.setTextSize(1); --prikaz temperature na zaslonu--

display.setCursor(0,0);

display.print("Temperature: ");

display.setTextSize(2);

display.setCursor(0,10);

display.print(t);

display.print(" ");

display.setTextSize(1);

display.cp437(true);

display.write(167);

display.setTextSize(2);

display.print("C");

display.setTextSize(1); --prikaz vlage na zaslonu--

display.setCursor(0, 35);

display.print("Humidity: ");

display.setTextSize(2);

display.setCursor(0, 45);

display.print(h);

display.print(" %");

display.display();

Firebase.setFloat ("Temp", t); --pošiljanje podatkov na firebase--

Firebase.setFloat ("Hum", h);

Firebase.setFloat ("Gas", a/1023\*100);

}

# Viri

<https://en.wikipedia.org/wiki/Gas_detector> (14.1.2020)

<https://www.cytron.io/p-mq135-air-quality-sensor-module> (14.1.2020)

<https://learn.adafruit.com/dht> (14.1.2020)

<https://www.amazon.de/dp/B01L9GC470/ref=sr_1_3?keywords=arduino+lcd+i2c+display&qid=1579262215&sr=8-3> (17.1.2020)

<https://en.wikipedia.org/wiki/Raspberry_Pi> (14.1.2020)

<https://www.raspberrypi.org/downloads/raspbian/> (20.10.2019)

<https://rufus.ie/> (20.10.2019)

<https://www.magicmirrorcentral.com/complete-raspberry-pi-magic-mirror-tutorial/> (28.10.2019)

<https://github.com/jclarke0000/MMM-DarkSkyForecast> (24.11.2019)

<https://github.com/basknol/MMM-PushBulletNotifications> (25.11.2019)

<https://www.arissi.eu/esp8266/> (15.1.2020)

<https://www.arduino.cc/en/main/software> (9.11.2019)

<https://thingspeak.com/> (15.1.2020)

<https://www.ebay.co.uk/itm/Alexa-Smart-Mirror-Medium-Size-Oak-frame-7-smart-display/183111419164?hash=item2aa24a851c:g:EzEAAOSwcttcHRzN> [19.1.2020]

<https://www.ebay.co.uk/itm/Smart-Bathroom-Mirror-with-WIFI-Weather-Demist-Illuminate-LED-Light-Touch-Switch/153270208455?hash=item23af9dc7c7:m:me1SPfep3PzzYMR01tYFC8Q> [19.1.2020]