Univerzitet u Beogradu

Fakultet organizacionih nauka

Katedra za elektonsko poslovanje

Primena IoT u poljoprivredi (Projekat)

Seminarski rad iz Elektronskog poslovanja

|  |  |
| --- | --- |
| Profesor: | Student: |
| Tamara Naumović | Aleksandar Nedeljković 2023/0320  Tijana Vujović 2023/0169  Radica Golubović 2023/0078 |

Beograd 2024.

Sadržaj

[PYTHON APLIKACIJA 3](#_Toc156123018)

[1. Opis projekta: 3](#_Toc156123019)

[2. Upotreba API ključa: 3](#_Toc156123020)

[3. Povezivanje sa NodeMCU: 3](#_Toc156123021)

[4. Funkcije programa: 4](#_Toc156123022)

[get\_weather\_data(lat, lon, api\_key) 4](#_Toc156123023)

[get\_weather\_forecast(lat, lon, api\_key) 4](#_Toc156123024)

[get\_soil\_data(poly\_id, api\_key) 5](#_Toc156123025)

[get\_uv\_index(poly\_id, api\_key) 5](#_Toc156123026)

[list\_parcels(api\_key): 6](#_Toc156123027)

[display\_parcel\_options(selected\_polygon): 7](#_Toc156123028)

[5. Ostale pomoćne funkcije: 8](#_Toc156123029)

[clear\_terminal() 8](#_Toc156123030)

[kelvin\_to\_celsius(kelvin) 8](#_Toc156123031)

[convert\_unix\_time(unix\_time) 8](#_Toc156123032)

[6. Upotreba programa: 8](#_Toc156123033)

[7. Dodatne informacije: 8](#_Toc156123034)

[NODEMCU 9](#_Toc156123035)

[1. Opis projekta: 9](#_Toc156123036)

[2. Povezivanje sa Wi-Fi mrežom: 9](#_Toc156123037)

[3. Povezivanje sa senzorima i relejima: 9](#_Toc156123038)

[5. Postavljanje uređaja: 10](#_Toc156123039)

[6. Funkcije programa: 10](#_Toc156123040)

[handleGetSensor(): 10](#_Toc156123041)

[handleGetRelayState(): 10](#_Toc156123042)

[handlePostRelay(): 11](#_Toc156123043)

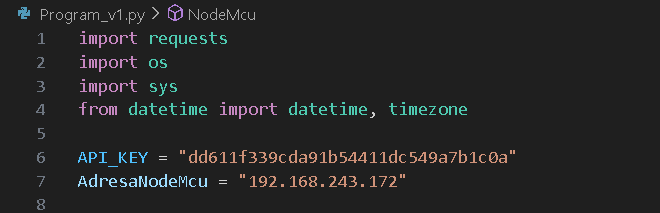
## PYTHON APLIKACIJA

## 1. Opis projekta:

Python program koji omogućava praćenje vremenskih i poljoprivrednih podataka za različite parcele. Program koristi API-jeve za dobijanje vremenskih podataka, prognoze, indeksa UV zračenja i podataka o zemljištu.

## 2. Upotreba API ključa:

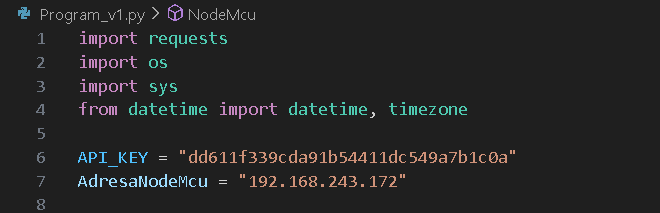
Da biste koristili program, neophodno je da imate API ključ sa AgroMonitoring API servisa. API ključ treba uneti u promenljivu API\_KEY u samom programu.



## 3. Povezivanje sa NodeMCU:

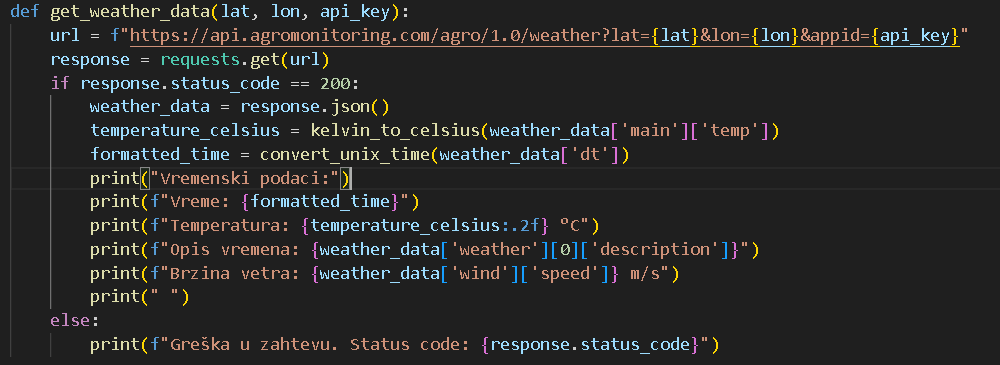
Program ima funkcionalnost za povezivanje sa NodeMCU uređajem putem lokalne mreže. NodeMCU prikazuje podatke o temperaturi, vlažnosti i stanju releja. Korisnik može upravljati relejima putem interfejsa u konzoli.

AdresaNodeMcu = "192.168.243.172"



## 4. Funkcije programa:

### get\_weather\_data(lat, lon, api\_key)

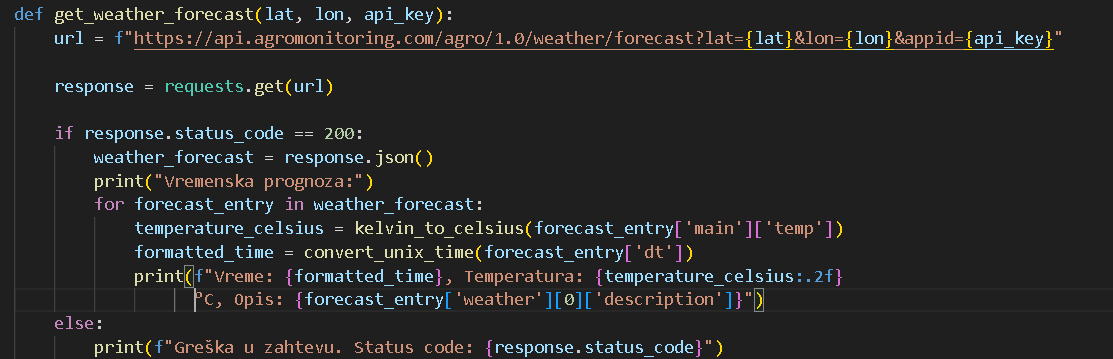


Ova funkcija šalje GET zahtev na AgroMonitoring API kako bi dobila trenutne vremenske podatke za određenu geografsku širinu (lat) i dužinu (lon).

Ako je odgovor uspešan (status code 200), podaci se formatiraju i prikazuju, uključujući vreme, temperaturu, opis vremena i brzinu vetra.

Ako je odgovor neuspešan, ispisuje se greška sa status kodom.

### get\_weather\_forecast(lat, lon, api\_key)



Ova funkcija šalje GET zahtev na AgroMonitoring API kako bi dobila vremensku prognozu za određenu geografsku širinu (lat) i dužinu (lon).

Ako je odgovor uspešan (status code 200), prikazuje vremensku prognozu za svaki unos u odgovoru, uključujući vreme, temperaturu i opis vremena.

Ako je odgovor neuspešan, ispisuje se greška sa status kodom.

### get\_soil\_data(poly\_id, api\_key)

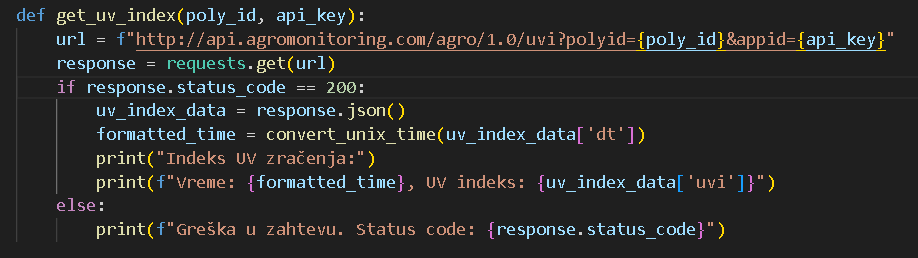


Ova funkcija šalje GET zahtev na AgroMonitoring API kako bi dobila podatke o zemljištu za određenu parcelu sa identifikatorom (poly\_id).

Ako je odgovor uspešan (status code 200), prikazuju se podaci o zemljištu, uključujući vreme, temperaturu na 10 cm dubine, vlažnost zemljišta i temperaturu na površini zemljišta.

Ako je odgovor neuspešan, ispisuje se greška sa status kodom.

### get\_uv\_index(poly\_id, api\_key)



Ova funkcija šalje GET zahtev na AgroMonitoring API kako bi dobila indeks UV zračenja za određenu parcelu sa identifikatorom (poly\_id).

Ako je odgovor uspešan (status code 200), prikazuju se podaci o indeksu UV zračenja, uključujući vreme i sam indeks.

Ako je odgovor neuspešan, ispisuje se greška sa status kodom.

### list\_parcels(api\_key):



Šalje GET zahtev na AgroMonitoring API kako bi dobila informacije o parcelama.

Prikazuje korisniku listu dostupnih parcela sa rednim brojem i imenom.

Omogućava korisniku da izabere parcelu ili da se vrati na glavni meni.

Parametri:

api\_key: Ključ koji se koristi za pristup AgroMonitoring API.

Povratne vrednosti:

Ako uspešno dobije podatke, vraća odabrani objekat koji predstavlja parcelu.

Ako ne uspe u dobijanju podataka ili korisnik odustane, vraća None.

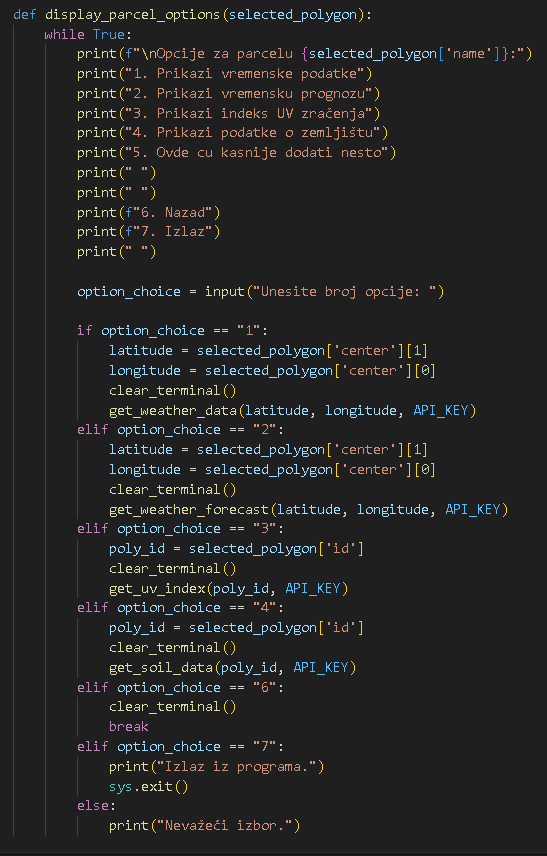
Detalji:

Prikazuje listu parcela i dodaje opciju za odabir NodeMcu uređaja.

Korisnik bira parcelu unosom rednog broja.

Omogućava korisniku da se vrati na glavni meni ili izađe iz programa.

### display\_parcel\_options(selected\_polygon):



Prikazuje opcije koje korisnik može izabrati za odabranu parcelu.

Omogućava korisniku da bira između prikaza vremenskih podataka, vremenske prognoze, indeksa UV zračenja, podataka o zemljištu ili drugih dodatnih opcija.

Omogućava korisniku da se vrati na prethodni meni ili izađe iz programa.

Parametri:

selected\_polygon: Objekat koji predstavlja odabranu parcelu.

Detalji:

Prikazuje korisniku meni sa opcijama.

Na osnovu korisničkog unosa, izvršava odgovarajuću funkcionalnost (prikaz vremenskih podataka, vremenske prognoze, itd.) ili se vraća na prethodni meni.

Omogućava korisniku izlazak iz programa.

## 5. Ostale pomoćne funkcije:

### clear\_terminal()



Funkcija za čišćenje terminala radi bolje preglednosti.

### kelvin\_to\_celsius(kelvin)



Funkcija za konverziju temperature iz Kelvina u Celzijus.

convert\_unix\_time(unix\_time) 

Funkcija za konverziju vremena iz Unix formata u čitljiv format.

## 6. Upotreba programa:

Program se pokreće kao glavni modul i pruža interfejs za pregled podataka o različitim parcelama, vremenskim podacima, prognozama, indeksima UV zračenja i podacima o zemljištu.

## 7. Dodatne informacije:

Program je razvijen kao deo sistema za praćenje poljoprivrednih uslova i donošenje informisanih odluka. Korišćenjem različitih funkcija, korisnik može pratiti relevantne podatke za specifične parcele i upravljati relejima putem NodeMCU uređaja.

Napomena:

Pre pokretanja programa, obezbedite da su sve neophodne biblioteke instalirane, a API ključ validan.

Za interakciju sa NodeMCU uređajem, obezbedite tačnu IP adresu uređaja u promenljivoj AdresaNodeMcu.

Program pruža mogućnost praćenja podataka u realnom vremenu, kao i kontrole nad NodeMCU uređajem.

# NODEMCU

## 1. Opis projekta:

Arduino program namenjen NodeMCU uređaju koji omogućava prikupljanje podataka o temperaturi, vlažnosti vazduha, stanju releja i pruža HTTP API za daljinsku kontrolu releja.

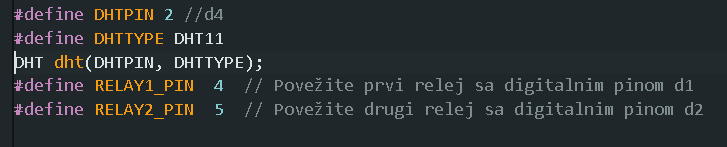
## 2. Povezivanje sa Wi-Fi mrežom:

Potrebno je uneti SSID i lozinku Wi-Fi mreže u odgovarajuće promenljive kako bi se NodeMCU povezao na mrežu.



## 3. Povezivanje sa senzorima i relejima:

Koristi se DHT11 senzor za merenje temperature i vlažnosti vazduha koji se povezuje na D2.

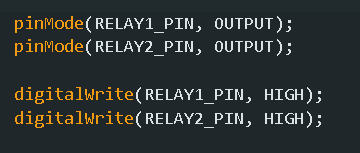
Prva dva digitalna pina (D1 i D2) koriste se za kontrolu dva releja. 4. Konfiguracija HTTP servera:

Postavljanje HTTP servera na port 80.

Definisanje endpointova za prikazivanje senzorskih podataka (/sensor), postavljanje stanja releja (/relay) i prikazivanje stanja releja (/relayState).

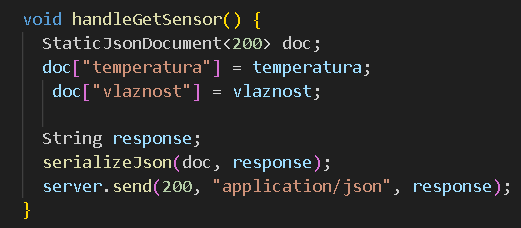


## 5. Postavljanje uređaja:

Inicijalizacija pinova i postavljanje početnog stanja za releje. }

## 6. Funkcije programa:

### handleGetSensor():

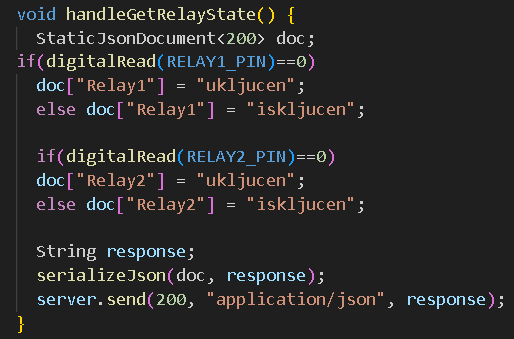


Ova funkcija odgovara na HTTP GET zahtev i šalje JSON odgovor koji sadrži trenutne podatke senzora (temperature i vlažnosti).

Podaci dolaze iz globalnih promenljivih temperatura i vlaznost.

Koristi StaticJsonDocument za kreiranje JSON dokumenta i serializeJson za konvertovanje u string.

### handleGetRelayState():



Odgovara na HTTP GET zahtev i šalje JSON odgovor sa trenutnim stanjem releja (Relay1 i Relay2).

Koristi digitalRead da proveri trenutno stanje svakog releja.

Ovisno o stanju, postavlja odgovarajući tekst u JSON dokument.

Koristi StaticJsonDocument i serializeJson za konverziju u string.

### handlePostRelay():



Odgovara na HTTP POST zahtev, parsira JSON podatke i postavlja stanje releja.

Koristi deserializeJson za parsiranje JSON podataka dobijenih iz tela zahteva.

Proverava vrednosti relay1 i relay2 iz JSON-a i postavlja globalne promenljive relayState1 i relayState2.

Na osnovu stanja, postavlja odgovarajuće nivoe na pinovima releja (RELAY1\_PIN i RELAY2\_PIN).

Šalje odgovor sa tekstualnim podacima o stanju releja.