Web aplikacija za vizualizaciju vlastitih meteoroloških podataka

Subotić, Maja

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Maja Subotić

WEB APLIKACIJA ZA VIZUALIZACIJU VLASTITIH METEOROLOŠKIH PODATAKA

ZAVRŠNI RAD

Varaždin, 2023.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Maja Subotić

JMBAG: 0116122241

Studij: Primjena informacijske tehnologije u poslovanju

WEB APLIKACIJA ZA VIZUALIZACIJU VLASTITIH METEOROLOŠKIH PODATAKA

ZAVRŠNI RAD

Mentor:

doc. dr. sc. Matija Novak

Varaždin, veljača 2023.

Maja Subotić

Izjava o izvornosti

Izjavljujem da je moj završni rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristila drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autorica potvrdila prihvaćanjem odredbi u sustavu FOI-radovi

Sažetak

Cilj ovog rada je izrada meteorološke postaje i aplikacije koja prikazuje prikupljene podatke o temperaturi i tlaku zraka. Potrebno je izraditi shemu za hardverski dio meteorološke postaje te fizički sastaviti dijelove u funkcionalnu cjelinu po izrađenoj shemi. Za kompletiranu meteorološku postaju izrađuje se program koji kontrolira senzore, koristeći programski jezik Python. Aplikacija za prikupljanje podataka, spremanje u bazu te prikaz na web-u izrađuje se pomoću PHP-a, JavaScript-a, HTML-a uz korištenje Google Charts API-ja te jQuery okvira. Izrađuje se baza podataka kako bi se podaci sa meteorološke postaje spremali i bili dostupni za korištenje i u budućnosti. Za pisanje koda koristi se program Notepad++. U radu se namjerava prikazati jednostavnost izrade kućne meteorološke postaje, kako hardverskog tako i aplikacijskog dijela, obzirom na dostupnost fizičkih komponenti ali i dostupnost informacija, biblioteka koda i okvira za vizualizaciju podataka.

Ključne riječi: php, javascript, arduino, (web) aplikacija, meteorološki podaci

Sadržaj

1. Uvod……...1

2. Razvoj web aplikacija..... 2

2.1. Okviri za razvoj web aplikacija 3

2.2. jQuery 4

2.3. Usporedba JavaScript okvira za vizualizaciju podataka..... 7

2.4. Google Visualisation API (Google Charts) 8

3. Senzorski sustav – meteorološka postaja..... 11

3.1. Opis senzora 11

3.1.1. BME280 senzor modul 12

3.1.2. ESP8266 D1 Mini WiFi Dev Board 13

3.1.3. Dodatna PCB pločica (eng. *shield board*) 15

3.2. Opis sklopa 16

3.3. Opis koda..... 17

4. Web aplikacija za vizualizaciju meteoroloških podataka..... 21

4.1. Baza podataka..... 21

4.2. Opis dijela aplikacije za zaprimanje podataka 23

4.3. Opis korisničkog dijela aplikacije 26

4.3.1. Prijava korisnika u sustav 26

4.3.2. Registracija novog korisnika 28

4.3.3. Upravljačka ploča 30

4.3.4. Promjena korisničkih postavki..... 31

4.4. Opis dijela aplikacije za prikaz podataka 32

4.4.1. Prikaz podataka u tablici..... 32

4.4.2. Prikaz podataka na grafikonu 34

5. Kritički osvrt na razvoj aplikacije 40

6. Zaključak ……42

Popis literature..... 44

Popis slika …… 46

Popis tablica …… 47

Prilozi ……... 48

1. Uvod

Ovaj završni rad prikazati će jedan od jednostavnijih načina za izradu vlastite meteorološke postaje, te aplikacije koja samostalno prikuplja podatke o temperaturi i tlaku zraka sa pripadajućeg senzora, pohranjuje ih u bazu podataka, a zatim na zahtjev korisnika prikazuje na linijskom grafikonu i u tabličnom prikazu.

Web aplikacija je računalni program dostupan korisnicima putem web preglednika, a pohranjen je na udaljenom poslužitelju, za razliku od desktop aplikacije koju pokreće sam operacijski sustav računala. Zahvaljujući upravo tom svojstvu, aplikaciju se ne mora prilagođavati određenom operacijskom sustavu i karakteristikama računala, niti ju je potrebno posebno instalirati na računalo ili ažurirati nadogradnje. Prve web aplikacije razvijaju se 90-ih godina prošlog stoljeća. Kako je rastao broj korisnika na internetu tako je rasla i potražnja za web aplikacijama. Na sve veću kompleksnost web aplikacija utjecao je nagli razvoj osobnih računala te komponenti i sklopova.

Princip rada web aplikacija je povezanost klijenta (eng. *client*) sa poslužiteljem (eng. *server*), na način da se korisniku prikazuje sučelje definirano na samom poslužitelju, gdje se ujedno pohranjuju i podaci, čime se omogućava pristup sa bilo kojeg uređaja ili lokacije. Web aplikacija radi na način da korisnik šalje zahtjev web poslužitelju (putem web preglednika), web poslužitelj taj zahtjev prosljeđuje aplikaciji na serveru koja izvodi taj upit, generira rezultat te ga šalje web poslužitelju, koji ga prosljeđuje i prikazuje korisniku u njegovom web pregledniku. Web aplikaciju moguće je pokrenuti neovisno o tome koji operativni sustav koristimo, bio to MS Windows, Linux, UNIX, macOS ili neki drugi. Za pokretanje aplikacije potrebno je samo imati instaliran web preglednik koji podržava izvođenje odnosno pokretanje predmetne aplikacije. Obzirom da aplikaciju nije potrebno instalirati lokalno na računalo, ona ne zauzima diskovni prostor. Kako svi korisnici aplikacije istovremeno koriste istu verziju, otklanja se mogućnost nekompatibilnosti. Moglo bi se reći i da je održavanje ovih aplikacija jednostavnije jer se izvodi na serveru, te nisu potrebne nikakve intervencije na računalima korisnika. Kako bi se aplikacija mogla nesmetano koristiti potrebno je imati brzu i stabilnu internetsku vezu, a server mora biti dostupan čitavo vrijeme izvođenja aplikacije. Što se sigurnosti tiče, server na kojem se aplikacija izvodi izložen je mogućim napadima, što može dovesti do nedostupnosti aplikacije. („Web application“, bez dat.)

2. Razvoj web aplikacija

Aplikacije možemo podijeliti na nekoliko različitih vrsta obzirom na njihovu namjenu, sadržaj, funkcionalnosti i karakteristike. Najopćenitija podjela bi bila ona na statičke i dinamičke web aplikacije.

Statičke web aplikacije prikazuju neki jednostavan sadržaj. To može biti tekst, slike, video ili audio sadržaj. Pisane su uglavnom u HTML jeziku uz CSS i JavaScript, a sadržaj ovih aplikacija moguće je promijeniti isključivo promjenom samog programskog koda aplikacije. On se sastoji od unaprijed određenog broja datoteka koje su pohranjene na web poslužitelju, a korisniku se prikazuju na način da prilikom zahtjeva korisnika za određenom stranicom, poslužitelj vrati HTML datoteku navedenu u URL-u uz popratne JavaScript i/ili CSS datoteke. Kako se ove datoteke ne mijenjaju prilikom razmjene web poslužitelj-korisnik, svim korisnicima će izgled i funkcionalnost aplikacije biti isti. Iako su statičke, uz dovoljno vještine, ove aplikacije mogu biti privlačne i interaktivne, na način da imaju hiperveze, gumbe, obrasce, animacije pokretane JavaScriptom ili CSS-om i slično. Prednosti statičkih web aplikacija leži u njihovoj jednostavnosti, kako izrade tako i održavanja. Osim toga, brže su na strani korisnika obzirom da se dohvaćaju tražene datoteke sa poslužitelja i dostavljaju se korisniku.

Za razliku od statičkih, sadržaj dinamičkih web aplikacija je fleksibilan, što znači da može prikazati različit sadržaj različitim korisnicima, ovisno o njihovim postavkama, lokaciji, vremenu i dr. Osim prilagodbe sadržaja pojedinom korisniku, programiranje na strani korisnika daje nam nebrojeno mnogo opcija funkcionalnosti aplikacije. Nadalje, prednost dinamičkih aplikacija svakako je jednostavnost ažuriranja, obzirom da se izmjene mogu napraviti brzo i jednostavno. Kao nedostatke dinamičkih web aplikacija možemo navesti zahtjev za puno više uloženog vremena, znanja i truda za izradu te potencijalni pad izvedbe, odnosno brzine učitavanja web aplikacije zbog pozadinskih procesa. Za izradu dinamičkih web aplikacija koriste se programski jezici na strani poslužitelja (eng. *server side*) od kojih su najpopularniji PHP, Python i Ruby, uz HTML, CSS i JavaScript na klijentskoj strani (eng. *client side*). Osim korištenja programskih jezika, dinamičke aplikacije mogu se razvijati i korištenjem web razvojnih okvira (eng. *web framework*) koji pojednostavljuju razvoj aplikacija te omogućavaju i pojednostavljuju izradu mnogih funkcionalnosti koje pridonose kvaliteti, sadržaju i interaktivnosti web aplikacije.

2.1. Okviri za razvoj web aplikacija

Web okvir (eng. *web framework*) ili okvir web aplikacije (eng. web *application framework*) je softverski okvir koji je dizajniran za podršku u razvoju web aplikacija uključujući web usluge, web resurse i web API-je (eng. *application programming interface)*. Web okviri pružaju standardni način izgradnje i implementacije web aplikacija na World Wide Web-u. Na primjer, mnogi web okviri pružaju biblioteke (eng. *library*) za pristup bazi podataka, predloške okvira i upravljanje sesijama, a također promiču ponovnu upotrebu koda. Iako često ciljaju na razvoj dinamičkih web stranica, primjenjivi su i na statičke web stranice. („Web framework“, bez dat.)

U današnjem svijetu svakodnevno se povećava količina podataka koju treba analizirati, odnosno razumjeti. Upravo tome doprinosi vizualizacija podataka, a alati i okviri za vizualizaciju podataka omogućavaju korisniku interakciju tim podacima. Okviri za vizualizaciju omogućavaju jednostavnu objavu rezultata, prate sve veće zahtjeve poslovanja, a opet trebaju biti jednostavni za korištenje. Okvire možemo podijeliti na klijentske kao što su Vue.js, AngularJS, EmberJS, jQuery – JavaScript te poslužiteljske kojima pripadaju Django - Python, Zend - PHP, Spring – Java i mnogi drugi. („Web framework“, bez dat.)

U Tablici 1. prikazani su statistički podaci o korištenju pojedinih JavaScript biblioteka, gdje možemo primijetiti da je tržišni udio jQuery-ja gotovo 95%, dok nam Tablica 2. prikazuje najbrže rastuće JavaScript biblioteke, gdje se podaci prikazuju u dnevnom porastu broja stranica na milijun, iz kojih možemo zaključiti kako bi u skoroj budućnosti neke nove biblioteke mogle preuzeti tržište, ako njihovo korištenje nastavi rasti ovakvim tempom.

Tablica 1. - Najpopularnije JavaScript biblioteke (Izvor: w3techs.com)

		Korištenje	**Promjena od 1.siječnja 2023.**	**Tržišni udio**	**Promjena od 1.siječnja 2023.**
1.	jQuery	77.6%		94.3%	-0.2%
2.	Bootstrap	21.3%	-0.2%	25.9%	-0.2%
3.	Modernizr	9.2%		11.2%	
4.	Underscore	8.7%	-0.1%	10.6%	-0.1%
5.	Popper	4.4%		5.3%	-0.1%

Postotaka mjesta

Tablica 2. - Najbrže rastuće JavaScript biblioteke (Izvor: w3techs.com)

		Stranice
1.	React	38.0
2.	Angular	37.9
3.	Lodash	24.6

Dnevni porast broja stranica na milijun

Web okviri kreirani su kako bi olakšali izgradnju web aplikacija koje se temelje na jednom programskom jeziku, od alata opće primjene koji povećavaju mogućnosti određenog jezika, do paketa koje je moguće programirati na izvornom jeziku, a izgrađeni su oko neke specifične aplikacije. Moraju funkcionirati sukladno pravilima protokola i preglednika. Poslužitelj web stranice poslužuje, a preglednik ih onda može mijenjati pomoću JavaScripta.

Promjene web stranica na strani poslužitelja zahtijevaju osvježavanje stranice, ali dopuštaju korištenje više računalne snage i bilo kojeg jezika, dok promjene na klijentskoj strani omogućavaju ažuriranja malih dijelova stranice, ali ograničene su JavaScript-om i snagom računala jer se izvode u pregledniku korisnika. U praksi se najčešće koristi kombinacija ove dvije mogućnosti. Aplikacije koje ažuriraju samo dijelove stranice i kontinuirano aktivno koriste JavaScript zovu se jednostranične aplikacije, te se kod njih za organizaciju koda uglavnom koristi JavaScript web okvir na klijentskoj strani.

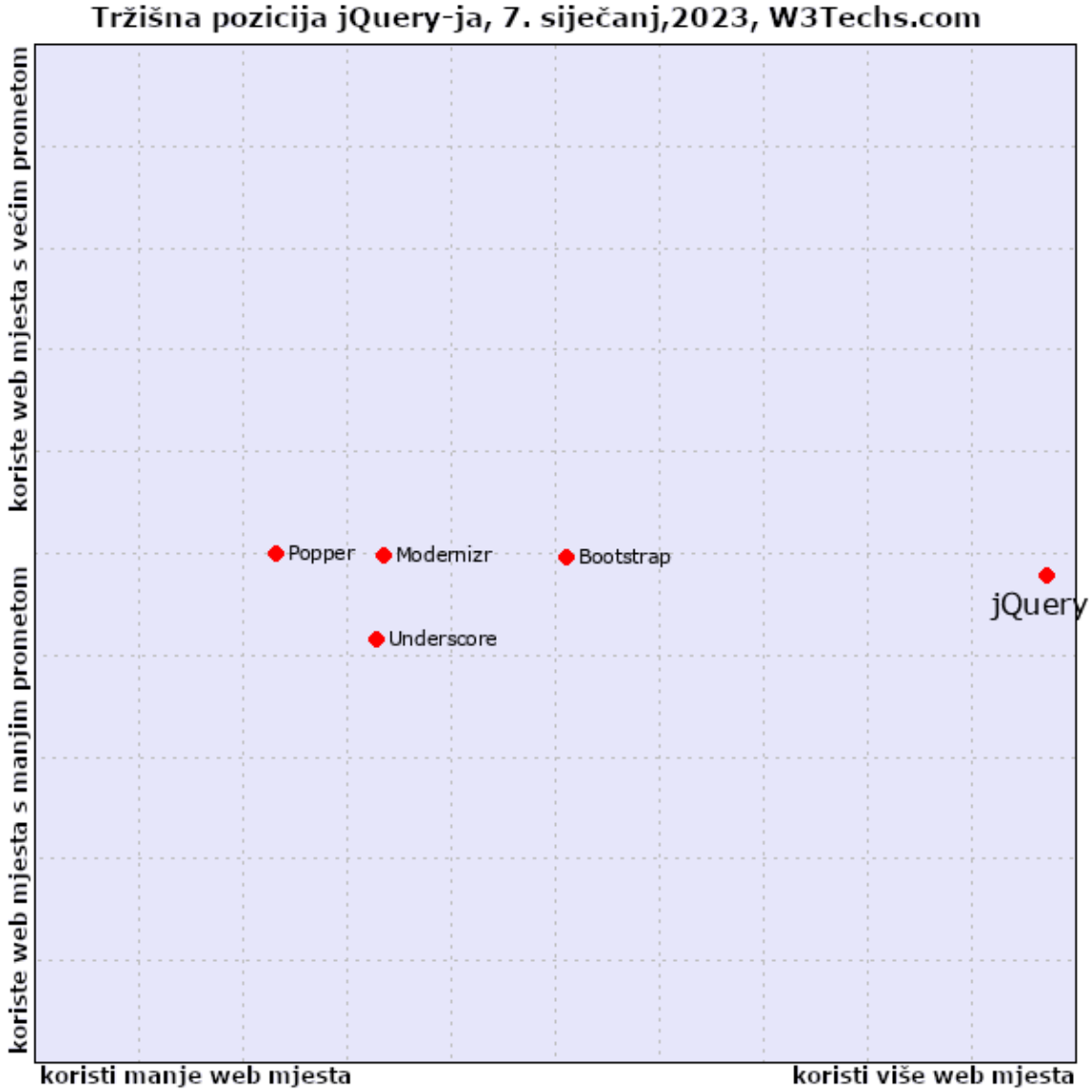
2.2. jQuery

Jedan od najranijih frontend okvira naziva se jQuery, a predstavljen je 2006. godine. Bez obzira na to što je prošlo puno vremena od prvog predstavljanja, čak i u današnjem svijetu kada se tehnologija razvija toliko brzo da je teško uvijek biti „u korak“ sa novim tehnologijama, jQuery je ostao relevantan. To je tako upravo zbog lakoće korištenja i jednostavnosti, ali i smanjene potrebe za pisanjem opširnih JavaScript kodova. Također zahvaljujući dugogodišnjem postojanju, razvila se poprilično mnogobrojna zajednica za jQuery rješenja. (Chaffer,2013.)

jQuery se koristi za manipulaciju CSS-om (eng. *cascading style sheets)* i DOM-om (eng. d*ocument object model*) te za optimizaciju funkcionalnosti i interaktivnosti web stranice. Iako donedavno nije bilo moguće izgraditi mobilne aplikacije s jQueryjem, razvojem jQuery Mobilea i to je omogućeno. Razvojem ovog okvira omogućilo se programerima da izgrade

izvorne mobilne aplikacije sa svojim HTML5 sustavom korisničkog sučelja. Prilagodljiv je preglednicima i podržava svaki preglednik koji korisnik namjerava koristiti.

U sljedećem dijagramu prikazane su tržišne pozicije obzirom na popularnost i promet najviše korištenih JavaScript biblioteka. Tehnologije koje su pozicionirane u donjem desnom kutu koristi puno web mjesta, ali ta web mjesta imaju prosječan promet, dok tehnologije pozicionirane u gornjem lijevom kutu koristi manje web mjesta, ali ta web mjesta uglavnom imaju veliki promet. Gornji desni kut dijagrama smatra se najboljim položajem jer su tu pozicionirane tehnologije koje koristi puno web mjesta sa velikim prometom. jQuery ovdje se pozicionirao kao tehnologija koju koristi najviše web mjesta koja imaju uglavnom srednju količinu prometa.



Slika 1. - Statistika korištenja JavaScript bibilioteka (Prema: W3Techs.com, 2023.)

Zahvaljujući samoj strukturi jQuery-ja, programeri imaju mogućnost stvaranja dodatnog koda kao proširenja njegove funkcije (eng. *plug-in*). Postoji jako velik broj dodataka koji su dostupni na webu, a pokrivaju cijeli niz funkcija, među kojima su i dinamičke liste, rukovanje kolačićima, događaji, modalni prozori, XML i XSLT alati i mnogi drugi. Za rad sa ovim dodacima, programeri mogu koristiti neke od već postojećih struktura ili napisati vlastiti kod. Neka od osnovnih načela programskog razvoja koristeći jQuery su razdvajanje JavaScript-a od HTML-a, jasnoća i sažetost u korištenju značajki poput skraćenih naziva funkcija i tzv. „lančanih" (eng. *chainable*) funkcija, jednostavno dodavanje i ponovno korištenje novih elemenata, metoda i događaja, te uklanjanje nekompatibilnosti različitih preglednika pružajući konzistentno sučelje kako bi korisnik mogao koristiti više različitih preglednika.

Jedna od bitnijih značajki jQuery-ja je upravljanje događajima kao što su prelazak mišem preko elementa stranice, odabir polja formulara, klik na određeni element stranice i slični događaji. jQuery rješava problem programiranja događaja, što je u JavaScriptu složen proces koji često izaziva nepredviđene probleme. Upravo to je jedan od najčešćih načina kontrole ponašanja događaja na stranici. Kada se obave sve potrebne operacije prilikom učitavanja stranice, programirani događaji preuzimaju ponašanje stranice. Funkcionalnost sustava se višestruko može proširiti raznim gotovim komponentama i dodacima (eng. *plug-in*), koje imaju mnogostruke mogućnosti od crtanja grafikona, napredne mogućnosti sortiranja, provjere ispravnosti podataka koji su uneseni u obrazac itd.

jQuery uglavnom koriste napredni programeri, te velike kompanije. Unatoč tome, često je korišten i od strane početnika u JavaScript-u, ali i ostalih amatera web dizajna, upravo zbog jednostavnosti učenja osnova, dok je za naprednija rješenja prilično jednostavno pronaći potreban dodatak, odnosno plug-in. Kao i kod svakog koda, izuzetno je važno poštovati sintaksu, jer i ovdje jedna točka, zarez ili drugi znak, mogu onesposobiti rad cijele aplikacije. jQuery olakšava dinamičan sadržaj, DOM je fleksibilan za dodavanje ili uklanjanje elemenata, a i slanje HTTP (eng. *hypertext transfer protocol*) zahtjeva je pojednostavljeno. Uz prethodno navedene prednosti, jQuery ima i neke nedostatke koje moramo spomenuti, kao što su relativno spora radna sposobnost, API-ji objektnog modela dokumenta su zastarjeli, a uz to dostupne su i mnoge naprednije alternative od jQuery-ja. („jQuery Introduction", bez dat.)

2.3. Usporedba JavaScript okvira za vizualizaciju podataka

JavaScript podržava biblioteke otvorenog koda (eng. *open-source libraries*), ali i komercijalne biblioteke grafika i grafikona. Zahvaljujući tome JavaScript nudi mnoga rješenja za vizualizaciju podataka, dodavanje animacija korisničkom sučelju, stvaranje dijagrama i grafikona te stvaranje 2-D i 3-D objekata i slika. Za odabir pravog okvira potrebno je obratiti pažnju na nekoliko stvari kao što su vrsta podataka koje je potrebno pretvoriti, vrsta željenog grafikona i slično. Također, treba pripaziti na kompleksnost samog okvira odnosno biblioteke, kako se ne bi doveli u situaciju da ne znamo koristiti odabrani okvir ili biblioteku. To ovisi o znanju samog programera, obzirom da neki okviri zahtijevaju široko znanje, dok su drugi relativno jednostavni za korištenje. Neke od najpopularnijih JavaScript biblioteka za grafikone su među ostalima i Chart.js te Google Charts. Oba okvira su otvorenog koda, stoga su i besplatni za korištenje.

Chart.js je jednostavan okvir za kreiranje grafikona. Kreiran je 2013.godine, ali od tada se konstantno mijenja, dopunjuje i usavršava. Licenciran je pod licencom MIT-ja (*Massachusetts Institute of Technology*), a održava ga aktivna zajednica programera. Manje promjene uvode se otprilike svaka dva mjeseca, dok se velike i ključne promjene događaju otprilike svakih nekoliko godina. Ova biblioteka pruža određeni broj najčešće korištenih vrsta grafikona, opcija prilagodbe i dodataka (eng. *plugins*). Osim ugrađenih vrsta grafikona, korisnicima daje mogućnost korištenja dodatnih vrsta grafikona koje održava sama zajednica. Također, ovaj okvir omogućava spajanje nekoliko različitih vrsta grafikona u jedan mješoviti grafikon. Koristi HTML5 platno (eng. *canvas*) za vrlo dobre performanse i prikaz u svim web preglednicima. Za ovaj okvir postoji detaljna dokumentacija uz lako razumljive primjere korištenja. Besplatan je i otvorenog je koda, te je prilično jednostavan i brz za savladati. Kao neke od nedostataka ovog okvira za vizualizaciju možemo navesti ograničene značajke (podržava samo osam vrsta grafikona), ne nudi mnogo mogućnosti za prilagodbu, kako se temelji na platnu, zna stvarati probleme sa ne vektorskim formatima. (Chart.js, bez dat.)

U Tablici 4 nalazi se usporedba pojedinih značajki i funkcionalnosti Chart.js i Google Charts okvira za vizualizaciju podataka, kao što su krivulja učenja, prilagodljivost, aktivan razvoj, interaktivnost, tipovi podataka, kompatibilnost i sl. Google Charts nudi daleko najveći broj grafikona među svim bibliotekama otvorenog koda, ali isto tako ne nudi dovoljno mogućnosti za prilagodbu i interaktivnost, pa je idealan za upotrebu kod grafikona bez složenih

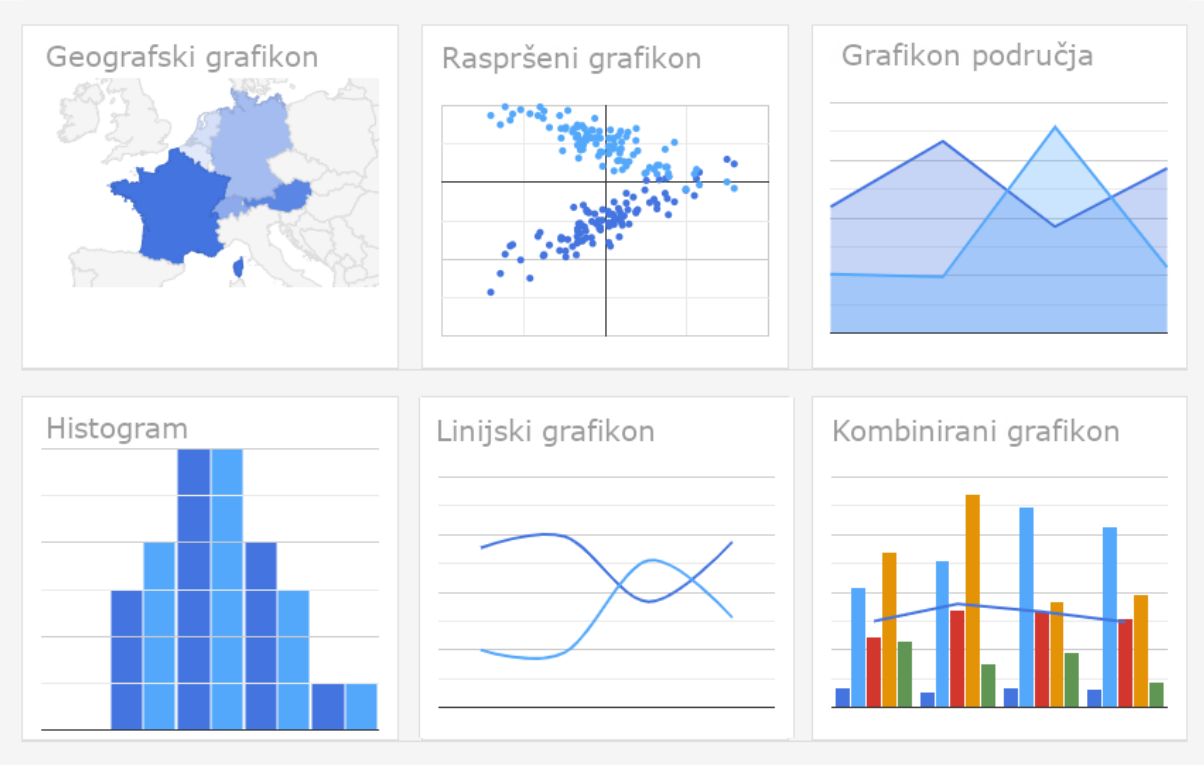
interakcija. Chart.js dobro je koristiti kod potrebe za jednostavnim i besplatnim skupom grafikona koji imaju minimalne konfiguracije.

Tablica 3. - Usporedba značajki Chart.js i Google Charts (Prema: fusioncharts.com)

	Chart.js	**Google Charts**
Krivulja učenja	Mala	Mala
Broj potrebnih linija koda	10-15	10-15
Prilagodljivost	Srednja	Srednja
Razvoj	Aktivan	Sporadičan
Tipovi podataka	JSON, JavaScript array	Google DataTable
Interaktivnost	Niska	Niska
Kompatibilnost	Mozilla Firefox, Google Chrome, Safari, Internet Explorer	Mozilla Firefox, Google Chrome, Safari, Internet Explorer, Opera, Android
Događaji (eng. *Events*)	Ne	Da
Predefinirani grafikoni	8+	60+
Izvoz na klijentskoj strain	Ne	PNG
Odabir raspona datuma	Ne	Da
Vremenski navigator	Ne	Da
Format ulaznih podataka	JavaScript API	CSV, JSON
Rendering Engine	Canvas	SVG

2.4. Google Visualisation API (Google Charts)

Za izradu ove aplikacije odabran je Google Charts. On omogućava izradu grafikona i aplikacija za izvještavanje preko strukturiranih podataka i pomaže ih integrirati izravno u web stranicu. Omogućava pristup strukturiranim podacima koji su preuzeti iz podržanih izvora podataka ili kreirani lokalno u pregledniku u tabličnom formatu. Moguća je i implementacija vlastitog izvora podataka kao izvora podataka API-ja za vizualizaciju čime možemo omogućiti, na primjer, pristup našim podacima bilo kojoj aplikaciji. Format je najpogodniji za korištenje u aplikacijama za analizu, izvještavanje i grafikone, a njime se mogu vizualizirati podaci ili aplikacijama dodati neke nove funkcije. (Google Chart API, bez dat.)



Slika 2. - Google Charts tipovi grafikona (Prema: Google Developers, 2023.)

Najčešći način korištenja Google Charts je JavaScript koji se ugrađuje u web stranicu. Učitaju se Google Charts biblioteke, navedu se podaci za prikaz, odabiru se opcije za prilagodbu grafikona te se stvara objekt grafikona s odabranim ID-em. Nakon toga, na web stranici je potrebno kreirati sa tim ID-jem za prikaz grafikona koji se ugrađuje u željenu web stranicu. (GoogleCharts, bez dat.).

Sve vrste grafikona popunjavaju se podacima pomoću klase „DataTable“, kako bi bio olakšan prijelaz sa jedne vrste grafikona na drugu dok tražimo željeni izgled. Pomoću „DataTable“ moguće je sortiranje, modificiranje i filtriranje podataka, koji se mogu popunjavati iz baze podataka, izravno sa web stranice ili bilo kojeg drugog davatelja podataka koji podržava protokol izvora podataka (uključujući jezik upita sličan SQL-u). Moguće je poslati SQL upit nekom izvoru podataka, koji nam kao odgovor vraća tablicu popunjenu traženim podacima. Slanje zahtjeva relativno je jednostavno u nekoliko koraka. Potrebno je instancirati objekt upita sa URL-om izvora podataka, na način da se URL-om naznače traženi podaci sintaksom razumljivom tom izvoru podataka. Moguće je po potrebi navesti opcije zahtjeva (npr. metoda slanja) kao drugi parametar objekta upita. Po želji korisnika, moguće je dodati niz jezika upita kako bi se sortirali ili filtrirali rezultati. Kada su svi potrebni parametri zadovoljeni, šalje se

upit. Ako izvor kojem šaljemo upit ne razumije jezik upita, ignorirati će SQL niz upita, ali bez obzira na to, vratit će DataTable. Nakon toga, šalje se upit, navodeći rukovatelja koji će biti pozvan kada se zaprimi odgovor.

Google Charts omogućava korištenje preko 30 različitih vrsta grafikona, kao što su stupčasti, kružni, mjehurićasti, razbacani, kombinirani i tablični grafikoni, ali i linije trenda, gantogram, histogram i mnogi drugi. Osim navedenih vrsta grafikona, moguće je koristiti i linijski grafikon koji je korišten u aplikaciji koja je predmet ovog rada. Grafikoni su izuzetno interaktivni te omogućuju povezivanje događaja pomoću kojih se mogu stvoriti složeni sustavi povezani sa web stranicom. Grafikoni se prikazuju korištenjem HTML5/SVG tehnologije kako bi bili kompatibilni sa svim preglednicima i mobilnim platformama (Android, iPhone…) te ne bi bili potrebni razni dodaci ili software-i.

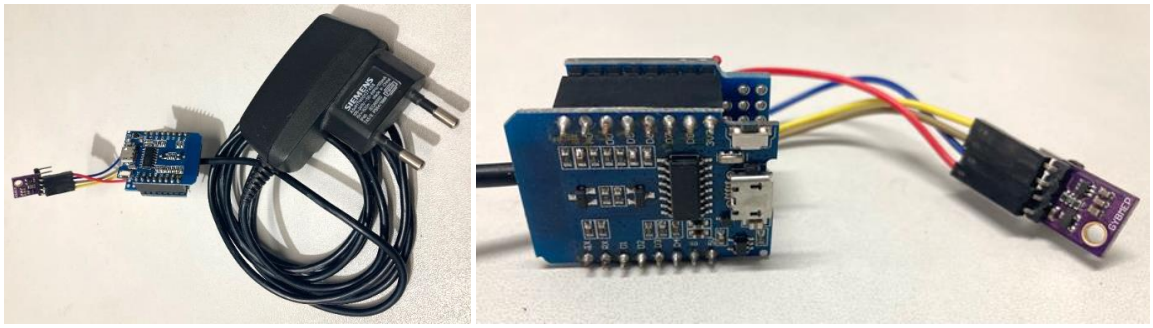
3. Senzorski sustav – meteorološka postaja

Proteklih 5 godina radim u firmi koja se bavi izradom senzora za praćenje vremenskih uvjeta u polarnim predjelima (Arktik, Antarktika…). Tamo sam stekla zanimanje za izradu raznih vrsta senzora, sustava i općenito hardvera. Dok su ti sustavi puno kompliciraniji i sa puno više senzora, načina rada i dr., meteorološka postaja koja se izrađuje u ovom radu jednostavnija je za izradu i upravljanje. Sve je više uređaja koji su povezivi na internet te se njima može manipulirati na daljinu, pratiti njihovo stanje i slično. Upravo to me ponukalo na izradu male kućne meteorološke postaje koja mjeri temperaturu i tlak zraka.

Meteorološka postaja je uređaj koji prikuplja podatke o okolišu i vremenu koristeći različite vrste senzora koje prikupljaju različite vrste podataka. Mogli bismo reći da postoje dvije vrste meteoroloških aplikacija, one koje koriste podatke sa poslužitelja meteoroloških postaja pomoću Interneta stvari (eng. *Internet of Things,* kraće IoT), a druga su vrsta meteorološke postaje koje imaju vlastite senzore. Naša meteorološka postaja je ove druge vrste, odnosno koristi svoj senzor za prikupljanje podataka. U nastavku ću pobliže opisati svaku od pločica sustava, te njihove specifikacije i korištenje.

3.1. Opis senzora

Senzorski sustav sastoji se od ESP8266 D1 Mini WiFi razvojne pločice koja se zasniva na arduino tehnologiji, dodatne pločice (eng. shield board) za adapter napajanja te BME280 modul pločice sa senzorom za tlak i temperaturu. Sve tri pločice su međusobno povezane žicama ili pinovima (eng. pin header).

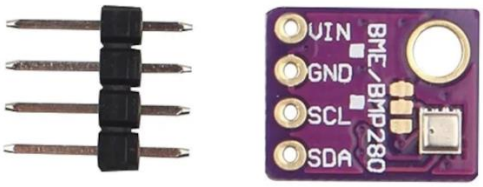


Slika 3. - Senzorski sustav meteorološke postaje (Izvor: autorski rad)

3.1.1. BME280 senzor modul

Najvažnija komponenta ovog sustava svakako je BME280 modul pločica sa senzorom pomoću koje očitavamo vrijednosti temperature i tlaka zraka. BME280 je visoko precizni senzorski modul koji mjeri atmosferski tlak i temperaturu zraka. Modul sadrži senzor barometarskog tlaka BME280 tvrtke Bosch koji je sljedeća generacija senzora BMP085/BMP180/BMP183. Senzori su unaprijed kalibrirani, pa samim time odmah nakon uključivanja počinju mjeriti temperaturu i tlak. Za kalibraciju ili rad sa ovim senzorom nisu potrebne dodatne komponente. Ovaj senzor je izvrstan za sve vrste detekcije vremena, a može se čak koristiti i u I2C i SPI sučelju. Ovaj vrlo precizan senzor za mjerenje barometarskog tlaka s ±1 hPa apsolutne točnosti i temperature s točnošću od ±1,0°C, najbolje je jeftino rješenje za ovaj projekt. („BME280“, bez dat.)

Veličina modula je 11.5x15mm, razmak između pinova (eng. *pin pitch*) je 2.54mm. Modul sadrži senzor BMP280 sa pripadajućim otpornicima i kondenzatorima. Proizvođač je predvidio mogućnost izbora između dvije I2C adrese. Početna zadana (eng. *default*) I2C adresa je 0x76, ali se može promijeniti. Senzor je zalemljen na PCB pločicu i dolazi s 3,3V regulatorom i pretvaračem logičke razine tako da ga se može bez brige koristiti s 3V ili 5V logičkim mikrokontrolerom. Na PCB pločici nalaze četiri pina - SDA i SCL za komunikaciju te VIN i GND za napajanje. I2C je serijski komunikacijski protokol, tako da se podaci prenose bit po bit duž jedne žice (SDA linija).



Slika 4. - BME280 senzor (Izvor: robu.in, 2020.)

Pinovi za napajanje:

Vin - pin za napajanje

GND – zajednički ground za napajanje i logiku

I2C logički pinovi:

SDA – „Serial Data“ linija za slanje i primanje podataka

SCL – „Serial Clock“ linija koja prenosi signal sata

Specifikacija senzora:

- Napon: 3.3V
- Raspon mjerenja tlaka: 30kPa - 110kPa; Pogreška u mjerenju tlaka: +-12Pa
- Raspon mjerenja temperature: -40C - 80C; Pogreška u mjerenju temperature: +-1C
- Potrošnja prilikom mjerenja: 1mA; Potrošnja u stanju mirovanja: <5uA

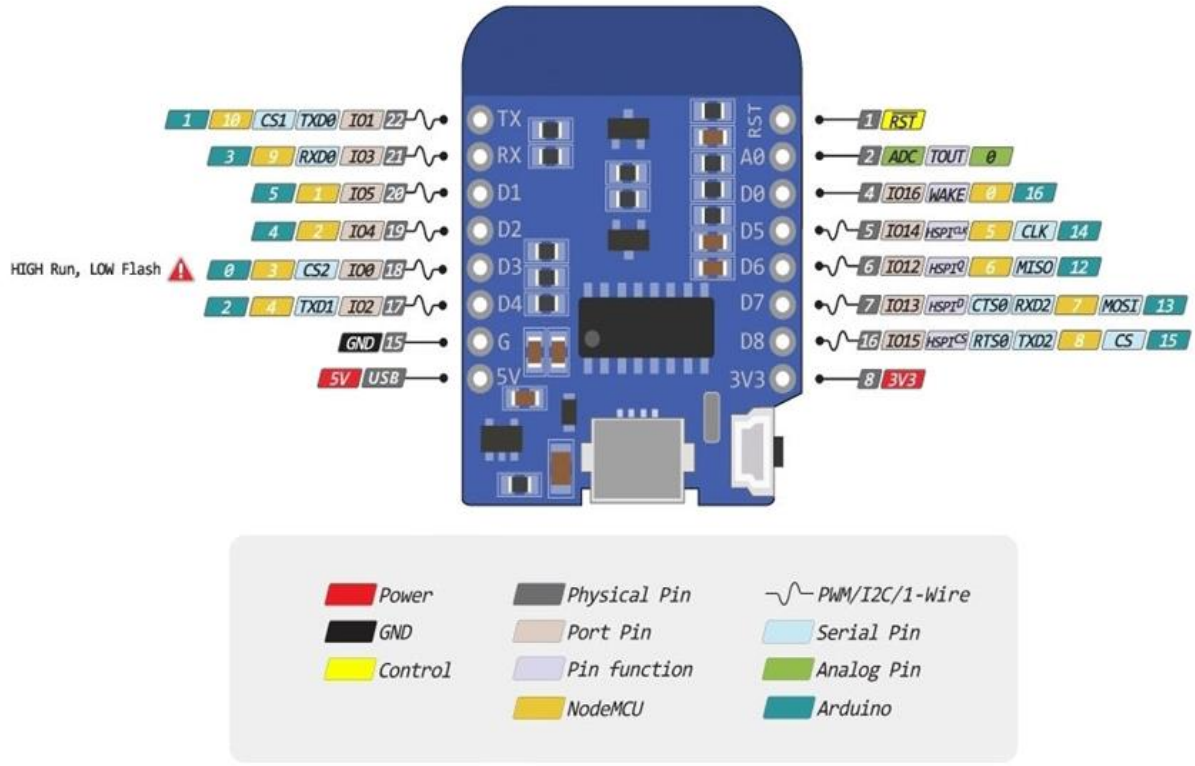
3.1.2. ESP8266 D1 Mini WiFi Dev Board

ESP8266 je razvojna pločica koja sadrži bežični mikrokontroler (WiFi). Modul je moguće programirati izravno iz Arduino IDE sučelja zahvaljujući ugrađenom microUSB konektoru koji se može povezati izravno na računalo pripadajućim kabelom. Za programiranje nije potreban drugi dodatan hardver, te se vrlo jednostavno programira, kao i svaki mikrokontrolerski modul koji se bazira na Arduino sustavu. („Protis“, bez dat.)

Slika 5. - ESP8266 D1 Mini WiFi Dev Bord (Izvor: addicore.com, 2023.)

Arduino tehnologija je izraz za platforme otvorenog koda (eng. *open source*) koje služe za kreiranje elektroničkih prototipova. Bazira se sklopovlju te jednostavnom i fleksibilnom programskom paketu. Platforme temeljene na arduino tehnologiji sastoje se od skupa jednostavnih softverskih i elektroničkih komponenata koje je moguće povezati u neku složeniju cjelinu. Najbitniji dijelovi ovih platformi su mikrokontroleri. To su mala računala koja se nalaze na jednom integriranom sklopu kojima upravljamo pomoću programa napisanog na računalu, a koji se potom izvodi na samom mikrokontroleru. Osim mikrokontrolera, platforme najčešće sadrže i integrirani sklop za komunikaciju sa računalom, te ostalih elektroničkih dijelova koji osiguravaju mogućnost rada, kao što su kvarcni oscilatori frekvencije takta, regulatori napona itd. Kao open source platforma, dozvoljava dijeljenje i preuređivanje kako bi se kreirale nove platforme koji su međusobno kompatibilne, kao što je ova ESP8266 platforma. (www.e-elektro.com.hr: Što je Arduino? bez dat.)

Ova pločica sadrži 11 digitalnih ulazno/izlaznih pinova, 1 analogni ulazni pin (3,3 V Max), 16MB (128M bit) flash, vanjski antenski konektor, ugrađenu keramičku antenu i sadrži novi CP2104 USB to UART IC. Svi I/O pinovi imaju prekid (eng. interrupt), PWM, I2C i „one-wire“ sposobnost, osim pina D0. Obzirom da je ovo uređaj od 3,3 V, ako ga želimo spojiti na 5V digitalne senzore ili uređaje, potreban je pretvarač logičke razine. Senzor koji se koristi za ovu meteorološku postaju je kompatibilan sa logikom ove pločice, te stoga pretvarač nije potreban.



Slika 6. - Raspored pinova ESP8266 D1 Mini pločice (Izvor: addicore.com, 2023.)

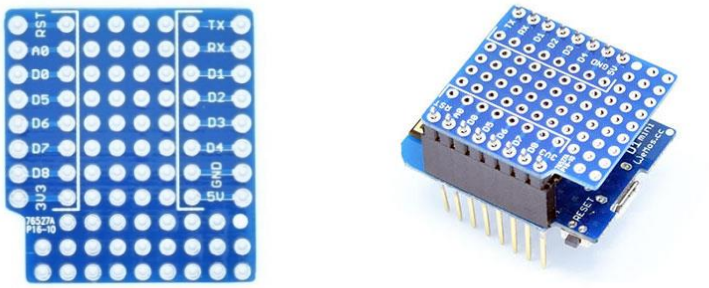
Pločica ima WiFi-SoC ESP8266 i napravljena je za brzu izradu prototipa Interneta stvari (IoT). Sa samo nekoliko redaka koda, pločicu je moguće povezati na lokalnu mrežu te ju je moguće kontrolirati od strane drugih uređaja na mreži poput računala i pametnih telefona. Opremljen je najnovijom verzijom firmware-a i može se odmah postaviti i programirati. Na ploču je ugrađen USB Micro konektor i „USB to serial“ (UART) pretvarač, što olakšava napajanje i programiranje pločice jednostavnim povezivanjem na računalo pomoću USB Micro kabela. Kompatibilna je sa Arduino IDE i NodeMCU softverima, te sa matičnom pločom koja se temelji na ESP8266. („D1 Mini WiFi Development Board“, bez dat..)

Specifikacije:

- Radni napon: 3.3V; 1 analogni ulaz (3,2V max ulaz)
- Brzina takta: 80MHz/160MHz; Flash: 16MB, 16M bajtova (128M bita) Flash
- 11 digitalnih ulazno/izlaznih pinova, interrupt / pwm / I2C / one-wire
- Priključak za vanjsku antenu, Ugrađena keramička antena; USB-TO-UART IC
- Dužina: 34,2 mm; Širina: 25,6 mm

3.1.3. Dodatna PCB pločica (eng. *shield board*)

Dodatna odnosno „shield“ PCB (eng. *printed circuit board*) pločica korištena je kako bi se regulirao napon napajanja ESP8266 pločice. Ideja je bila da ova meteorološka postaja konstantno vrši mjerenja bez prestanka, te da se može koristiti na različitim lokacijama. Kako bi mjerenja bila konstantna, postaja mora biti pod neprestanim napajanjem, a da bi se mogla koristiti na više lokacija, mora biti neovisna od računala, odnosno računalnog napajanja. Kao najjednostavnije rješenje, iskorišten je adapter starog mobilnog uređaja koji se može priključiti na bilo koju utičnicu koja isporučuje 220V. Korišten je adapter marke Siemens, specifikacija 100-240V / 100mA; 5,4V/620mA. Kako taj adapter nije odgovarao maksimalnom ulaznom naponu ESP8266 pločice, bilo je potrebno napraviti prilagodni sklop odnosno regulator napona. Uz izračun potrebne korekcije, napravljen je prilagodni sklop od pripadajućih kondenzatora (0.1uF i 0.33uF) i regulatora napona LM7805. LM7805 je regulator napona koji isporučuje +5 volti. („LM7805“, bez dat.) Kao i većina drugih regulatora na tržištu, to je integrirani krug sa tri pina; ulazni pin za prihvaćanje dolaznog istosmjernog napona, pin za uzemljenje (eng. *ground*) i izlazni pin koji napaja +5 volti. Kada uređaj nije priključen u utičnicu preko adaptera, moguće ga je koristiti priključenog na računalo pripadajućim USB kabelom.

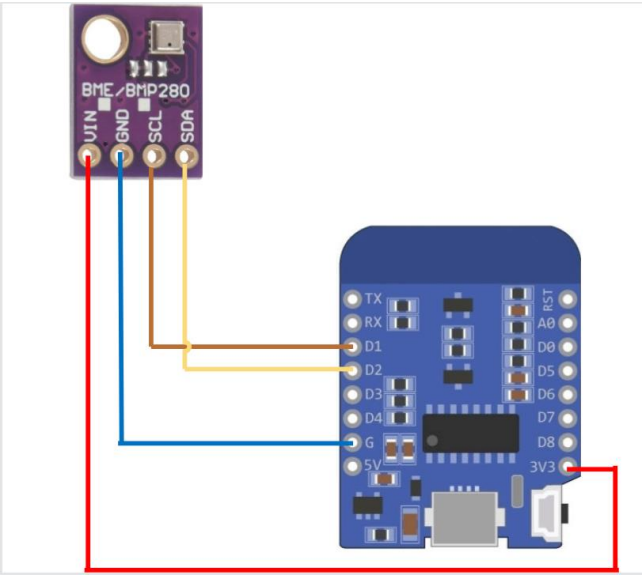


Slika 7. - Dodatna (shield) pločica (Izvor: ebay.de)

3.2. Opis sklopa

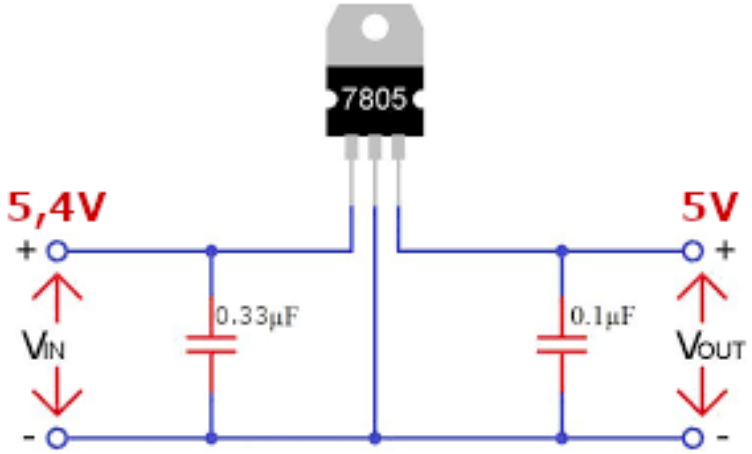
Meteorološka postaja sastoji se od nekoliko kompatibilnih PCB pločica od kojih svaka ima svoju funkciju, bilo to mjerenje, prikupljanje podataka ili napajanje. Kako bi se uopće počelo sa izradom, potrebno je napraviti dijagram sklapanja, pribaviti senzore i potrebnu elektroniku te ju učiniti funkcionalnom pisanjem koda za čitanje podataka sa senzora. Podaci koje meteorološka postaja prikupi, šalju se i pohranjuju u bazu podataka, te će biti vizualizirani putem web aplikacije. Postoji još puno mogućnosti za korištenje ove meteorološke postaje povezivanjem sa drugim uređajima u stanu ili kući, kao što su grijanje, klima i slično, ali osnovna joj je funkcija reći kakvo je vrijeme tamo gdje je postavljena. Prateći temperaturu i tlak zraka, možemo znati na kojoj smo nadmorskoj visini, pada li kiša, hoće li u skorije vrijeme početi padati kiša (promjena tlaka zraka pred kišu), koja je temperatura i dr.

Pločice su međusobno povezane žicama i/ili konektorima (eng. *pin headers*), ovisno o funkciji pločice. BME280 modul povezan je žicama za ESP8266 modul kako bi bio odvojen od regulatora koji se griju te tako mogu utjecati na preciznost mjerenja samog senzora temperature. Dodatna pločica i ESP8266 modul povezani su konektorima zbog lakše manipulacije pločicama i komponentama na njima. BME280 modul povezan je sa sva svoja četiri pina na pripadajuće pinove ESP8266 modula. Pin GND povezan je na istoimeni pin GND, Vin pin povezan je na 3.3V pin, SCL pin povezan je na pin D1, a SDA pin povezan je na pin D2. Konkretne pinove odredili smo temeljem rasporeda pinova ESP8266 modul pločice (Slika 6.) Dijagram spajanja senzora meteorološke postaje je prikazan na slici 8.



Slika 8. - Shema spajanja senzora (Izvor: autorski rad)

Na dodatnu pločicu instaliran je prilagodni sklop napravljen od kondenzatora i regulatora napona. Sklop radi na način da zaprima do +12V, a u našem slučaju 5.4V, koji preko kondenzatora od 0.33uF odlaze na regulator napona LM7805, te iz njega, preko kondenzatora od 0.1uF izlazi čistih 5V koje koristimo za napajanje ESP8266 modula, iz kojeg se onda sa 3,3V napaja BME280 modul sa senzorima. Shema prilagodnog sklopa prikazana je na slici 9.



Slika 9. - Shema prilagodnog sklopa (Izvor: autorski rad)

3.3. Opis koda

Za pisanje skripti, korišten je program za uređivanje teksta Notepad++. Kako biblioteka za čitanje sa senzora BME280 prema zadanim postavkama nije dio standardne MicroPython biblioteke, bilo je potrebno prenijeti ju na ESP8266 pločicu. Biblioteka se može naći nekim od brojnih foruma i stranica, ja sam koristila onu sa stranice github.com[1] za koju smatram da je najrelevantnija.

Određujemo i2c adresu senzora, a obzirom da nismo mijenjali hardware komponente, ona je ostala kao tvornički zadana adresa (0x76). („MicroPython Programming with ESP32 and ESP8266 eBook“, 2022.)

```
import time
from ustruct import unpack, unpack_from
from array import array

BME280_I2CADDR = 0x76
```

[1]https://github.com/RuiSantosdotme/ESP-MicroPython/blob/master/code/WiFi/HTTP_Client_IFTTT_BME280/BME280.py

```
BME280_OSAMPLE_1 = 1
BME280_OSAMPLE_2 = 2
BME280_OSAMPLE_4 = 3
BME280_OSAMPLE_8 = 4
BME280_OSAMPLE_16 = 5

BME280_REGISTER_CONTROL_HUM = 0xF2
BME280_REGISTER_CONTROL = 0xF4
```

Određujemo moguće načine rada, te radimo validaciju načina rada u kojoj postavljamo eventualne poruke za grešku prilikom rada kako bismo znali o kojoj grešci se radi.

```
class BME280:
def __init__(self,
mode=BME280_OSAMPLE_1,
               address=BME280_I2CADDR,
               i2c=None,
               **kwargs):
        if mode not in [BME280_OSAMPLE_1,
BME280_OSAMPLE_2,BME280_OSAMPLE_4,
                      BME280_OSAMPLE_8, BME280_OSAMPLE_16]:
   raise ValueError(
             'Unexpected mode value {0}. Set mode to one of '
             'BME280_ULTRALOWPOWER, BME280_STANDARD, BME280_HIGHRES, or '
             'BME280_ULTRAHIGHRES'.format(mode))
      self._mode = mode
      self.address = address
      if i2c is None:
          raise ValueError('An I2C object is required.')
         self.i2c = i2c
```

Funkcijom „read_raw_data“ čitamo neobrađene podatke sa senzora, pri čemu je argument „result“ niz gdje će rezultat biti pohranjen, u redoslijedu: temperatura, tlak zraka. Ova funkcija ne vraća podatke.

```
def read_raw_data(self, result):
   self._l1_barray[0] = self._mode
   self.i2c.writeto_mem(self.address, BME280_REGISTER_CONTROL_HUM,
self._l1_barray)
   self._l1_barray[0] = self._mode << 5 | self._mode << 2 | 1
   self.i2c.writeto_mem(self.address, BME280_REGISTER_CONTROL,
self._l1_barray)


   sleep_time = 1250 + 2300 * (1 << self._mode)
   sleep_time = sleep_time + 2300 * (1 << self._mode) + 575
   sleep_time = sleep_time + 2300 * (1 << self._mode) + 575
   time.sleep_us(sleep_time)  # Wait the required time


   self.i2c.readfrom_mem_into(self.address, 0xF7, self._l8_barray) readout
= self._l8_barray
   raw_press = ((readout[0] << 16) | (readout[1] << 8) | readout[2]) >> 4
   raw_temp = ((readout[3] << 16) | (readout[4] << 8) | readout[5]) >> 4


   result[0] = raw_temp
   result[1] = raw_press
```

Za razliku od prethodne funkcije, funkcija „red_compensated_data“ čita podatke sa senzora i vraća obrađene podatke. Argument u ovoj funkciji je također niz u kojem će biti pohranjen rezultat, a vratiti će niz sa temperaturom i tlakom.

```
def read_compensated_data(self, result=None):
   self.read_raw_data(self._l3_resultarray)
   raw_temp, raw_press = self._l3_resultarray
# temperature
   var1 = ((raw_temp >> 3) - (self.dig_T1 << 1)) * (self.dig_T2 >> 11)
   var2 = (((((raw_temp >> 4) - self.dig_T1) *
            ((raw_temp >> 4) - self.dig_T1)) >> 12) * self.dig_T3) >> 14
   self.t_fine = var1 + var2
   temp = (self.t_fine * 5 + 128) >> 8

# pressure
   var1 = self.t_fine - 128000
   var2 = var1 * var1 * self.dig_P6
   var2 = var2 + ((var1 * self.dig_P5) << 17)
```

```
var2 = var2 + (self.dig_P4 << 35)
var1 = (((var1 * var1 * self.dig_P3)>> 8)+((var1 * self.dig_P2) << 12))
var1 = (((1 << 47) + var1) * self.dig_P1) >> 33
if var1 == 0:
    pressure = 0
else:
    p = 1048576 - raw_press
    p = (((p << 31) - var2) * 3125) // var1
    var1 = (self.dig_P9 * (p >> 13) * (p >> 13)) >> 25
   var2 = (self.dig_P8 * p) >> 19
   pressure = ((p + var1 + var2) >> 8) + (self.dig_P7 << 4)

   if result:
          result[0] = temp
          result[1] = pressure
         return result

return array("i", (temp, pressure))
```

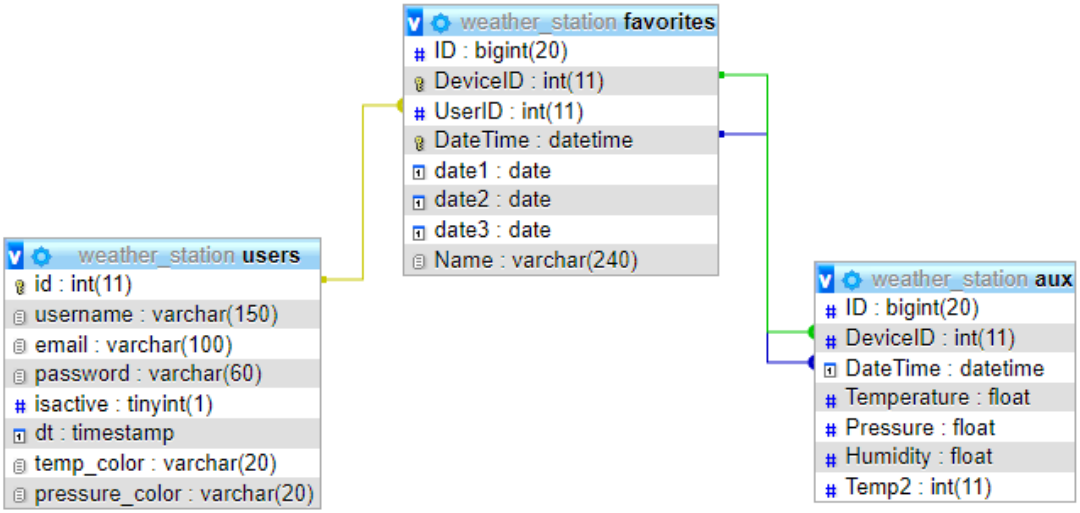
4. Web aplikacija za vizualizaciju meteoroloških podataka

Ova aplikacija korisniku pruža nekoliko funkcionalnosti. Prvenstveno to je funkcija prikaza trenutne temperature i tlaka zraka na grafikonu, uz prikaz vremena posljednjeg očitanja i proteka vremena od posljednjeg ispravnog očitanja sa senzora. Omogućeno je pretraživanje i prikaz podataka u određenom periodu putem filtera koji dohvaća stare podatke iz baze podataka i prikazuje ih na grafikonu. Filter je napravljen tako da prikazuje kalendar za lakši odabir datuma koje želimo pretražiti odnosno prikazati. Aplikacija također omogućava korisniku usporedbu podataka o temperaturi i tlaku zraka za tri različita datuma, na način da se podaci o određenom prethodnom datumu prikupljaju iz baze podataka gdje se spremaju nakon očitanja. Kako bi korištenje aplikacije bilo što ugodnije korisniku, omogućena je promjena postavki prikaza podataka na grafikonu, na način da korisnik može odabrati boju kojom se prikazuje određena vrsta podataka. Osim prethodno spomenutih značajki, korisnik može spremiti određene datume pod odabranim imenom kao „omiljene“, kako bi im u svakom trenutku mogao pristupiti na najjednostavniji način. Aplikacija je, zbog jednostavnijeg snalaženja u kodu i opisivanja istog, podijeljena na nekoliko dijelova odnosno cjelina. Pa se tako sastoji od dijela koji se bavi zaprimanjem podataka i spremanjem istih u bazu podataka, korisničkog dijela aplikacije koji sadrži kod za registraciju, prijavu, odjavu, promjenu postavki korisnika i sl., dijela za grafički i tablični prikaz podataka; te baze podataka u koju se spremaju podaci sa meteorološke postaje i korisnički podaci.

4.1. Baza podataka

Baza podataka nalazi se na CentOS Apache serveru. Baza se sastoji od tri jednostavne tablice u koju se unose podaci prikupljeni sa senzora, podaci o korisniku te spremljeni favoriti pojedinog korisnika. Tablice su međusobno povezane vanjskim ključevima što doprinosi funkcionalnosti aplikacije. Tablica „aux“ sastoji se od stupaca u koji se upisuju podaci tipa *float* o temperaturi, tlaku i vlažnosti zraka prikupljeni sa senzora, zajedno sa vremenom očitanja podatka sa senzora tipa *datetime*. U ovoj tablici se također nalaze ID stupca i ID uređaja odnosno senzora u tipu podatka *integer*. U tablicu „users“ upisujju se podaci o korisnicima. Ovi podaci prikupljaju se prilikom registracije korisnika, ali i prilikom promjene postavki prikaza na grafikonima. Sukladno tome, ova tablica osim ID-ja, korisničkog imena, lozinke i e-mail adrese, sadrži i stupce u koji se upisuju željene boje prikaza krivulja na grafikonu. Tablica

„favorites“ popunjava se podacima koje korisnik unosi spremanjem određenih datuma u favorite. Ona se sastoji od stupaca u koji se upisuju podaci o datumima koje je korisnik odabrao, ime samog favorita, a pomoću vanjskih ključeva povezane su sa tablicama „users“ i „aux“. U ERA dijagramu na slici 21 prikazane su tablice od kojih se sastoji baza podataka i njihove relacije.



Slika 10. - ERA dijagram baze podataka (Izvor: autorski rad)

Korištenjem PHP-a moguće je povezati se i upravljati bazom podataka. Najprije je potrebno spojiti se na server i bazu. Kako bi nam kasnije korištenje baze bilo lakše, kreirana je funkcija „getDB“. Pomoću „mysql_connect“ funkcije unošenjem potrebnih parametara spajamo se na bazu. Ukoliko spajanje nije uspješno, sustav će poslati upozorenje da je došlo do problema sa spajanjem na bazu podataka. („PHP MySql“, bez dat.)

```
function getDB($db) {
global $db_config;
   $connection_data=$db_config[$db];
   $link=@mysql_connect$connection_data['server'],$connection_data['user
'],$connection_data['passwd'],true)
   or Error("problem connecting to DB: " . $connection_data['dbase']);
   mysql_select_db($connection_data['dbase']);
   return $link; }
```

Podacima manipuliramo putem upita (*eng. query*), koje koristimo za dohvaćanje podataka iz baze, konkretno temperaturu, tlak zraka i vrijeme prikupljanja tih podataka, te filtriranje podataka za određeni period koji korisnik upiše u tražilicu. Osim upita korisnika, putem query-ja unosimo nove podatke u bazu podataka, mijenjamo ili brišemo postojeće podatke o korisnicima, te ih koristimo za druge funkcionalnosti koje se provode u pozadini poput upita o trenutno prijavljenom korisniku, validaciju korisničkog imena i lozinke i sl.

```
function runQuery($sql, $link) {
return @mysqli_query($link,$sql) or Error("Database Error(".mysqli_errno
($link)."): " . mysqli_error($link) . "<br>SQL: " . $sql);
        }
$encoded_pass = $_REQUEST['pass'];
$sql2 = "INSERT INTO users SET username='" . $_REQUEST['username'] ."',
password='" . $encoded_pass ."', email='" . $_REQUEST['email'] ."'";
$user = runQuery($sql2, $link);
```

Za zatvaranje veze na bazu podataka koristimo funkciju „mysql_close“. („PHP MySql“, bez dat.)

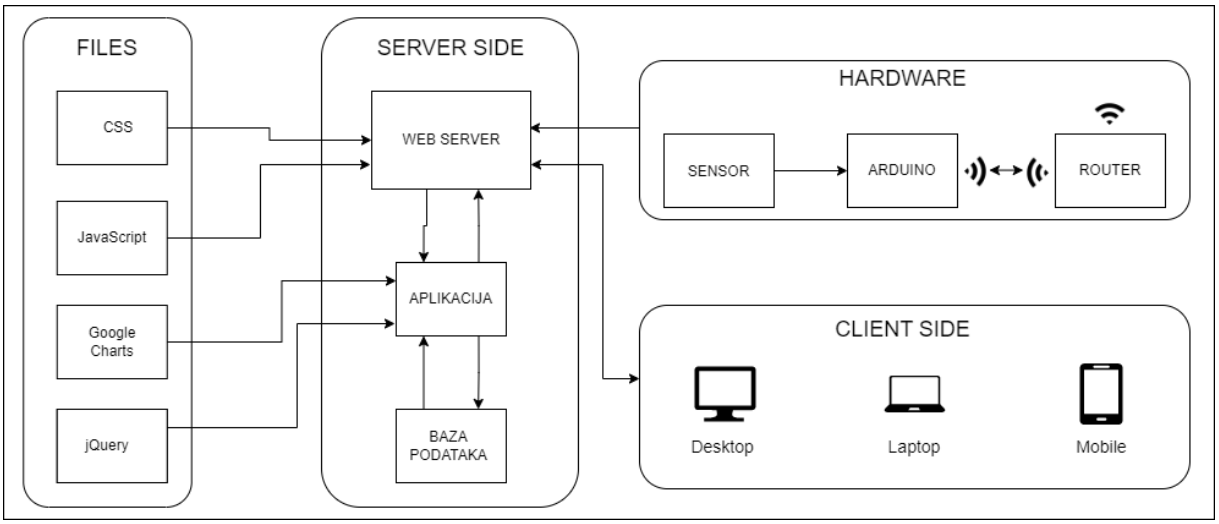
```
function closeDB($dblink) {
   mysql_close($dblink);
}
```

4.2. Opis dijela aplikacije za zaprimanje podataka

Dio aplikacije koji zaprima podatke i sprema ih u bazu podataka sastoji se od nekoliko PHP (eng. *Hypertext Preprocessor*) skripti zbog lakšeg korištenja i manipulacije samim kodom. Web API (*eng. Application programming interface*) je sučelje za programiranje aplikacija za web kao što mu i samo ime govori. Sastoji se od jedne ili više javno dostupnih krajnjih točaka (eng. *end points*) definiranog sustava poruka zahtjev-odgovor (*eng. request-response*). Sa klijentskih uređaja kao što su računala, mobilni telefoni, prijenosna računala pristupa se na poslužitelj putem HTTP (*eng. HyperText Transfer Protocol*) protokola na način da klijent šalje zahtjev u obliku HTTP zahtjeva te dobiva odgovor uglavnom u formatu XML (*eng. Extensible Markup Language*) ili JSON (*eng. JavaScript Object Notation).* Uglavnom se koriste za slanje upita poslužitelju tražeći specifične podatke sa tog poslužitelja. („Web API“, bez dat.)

JavaScript je omogućio kreiranje dinamičkih web stranica odnosno aplikacija obzirom na interakciju sa krajnjim korisnikom uz smanjenje potrebe komunikacije sa poslužiteljem.

Upravo takav odnos zaslužan je za trenutan odaziv prema korisniku, nema čekanja ponovnog učitavanja stranice, što povećava interaktivnost same stranice odnosno aplikacije. Uz AJAX (eng. *Asynchronous JavaScript and XML*) olakšava korištenje aplikacije komunikacijom sa odvojenim serverskim programom bez potrebe da se ponovno učitava sav sadržaj aplikacije. JSON je standard koji je predviđen za prijenos odnosno razmjenu podataka kao što su jednostavne varijable poput teksta (eng. *string*), brojeva (eng. *number*), logičkih podataka (eng. *boolean*), objektne varijable i vrijednosti *null*. jQuery nam omogućava jednostavno korištenje gotovih komponenata i proširenja, te manipulaciju istima. Google Chart API alatom stvaramo grafikon iz podataka te ga ugrađujemo u samu aplikaciju. Podaci i parametri se ugrađuju u HTTP zahtjev, a Google „vraća“ sliku grafikona u PNG formatu. Trenutno podržava samo izradu grafikona slanjem zahtjeva URL-om, ali moguće je implementirati i vlastiti kod za sastavljanje URL-a. („Charts- Developers“, bez dat.)



Slika 11. - Arhitektura sustava za vizualizaciju meteoroloških podatka (Izvor: autorski rad)

Potrebno je spojiti se na bazu podataka koju smo definirali, nakon toga provjeravamo imamo li broj parametara koji je veći od nula, odnosno postoji li išta u parametru „data“ ($_GET), ako postoji, izvršavaju se funkcije „send2BruncinServer“ i „writeLog“. Dohvaćamo parametar „data“ iz URL-a HTTP/HTTPS protokolom putem GET metode. Kada se zaprime podaci u obliku npr. „192.168.1.1./ws/r.php?data=12.4,30.1“ uzimaju se definirani parametri i zapisuju se u bazu podataka.

```
include_once ("config.php");
$link = getDB("ws1");

if(count($_GET) > 0) {
   send2BruncinServer();
   writeLog($link);
   echo(".OK.");
} else {
   echo(".ERROR.");
function writeLog($link) {
   $data_p = explode(",", $_GET["data"]);
   $date = date('Y-m-d H:i:s', time());

   $sql = "INSERT INTO aux SET DeviceID=" . $_GET["id"] . ", ";
   $sql .= "DateTime='" . $date . "', ";
   $sql .= "Temperature='" . $data_p[0] . "', ";
   $sql .= "Pressure='" . $data_p[1] . "' ";
   $r = runQuery($sql, $link);
   closeDB($link);
}
function send2BruncinServer() {
   $ch = curl_init();
   curl_setopt($ch, CURLOPT_URL,
"http://**.***.**.***/weather_station/r.php?id=1&data=" . $_GET["data"]);
   curl_setopt($ch, CURLOPT_HEADER, 0);
   curl_exec($ch);
   curl_close($ch);
}
```

Vrijeme zapisa podataka je lokalno vrijeme servera. Staza gdje se šalju podaci sa senzora i sama bežična konekcija, konfigurirani su u posebnoj skripti zbog lakšeg korištenja.

```
ESSID = "WifiNetworkName"
ESSID_pass = "WifiNetworkPassword"
APIURL = "http://192.168.1.1/weather_station/r.php?id=1&data="
APIURL2 = "http:// 192.168.1.2/weather_station/r.php?id=1&data="
```

4.3. Opis korisničkog dijela aplikacije

U ovom poglavlju biti će opisan dio aplikacije koji je vidljiv korisniku i kojim korisnik može manipulirati te ga uređivati prema svojim željama. Aplikacija je dostupna samo registriranim korisnicima, pa samim time početna stranica aplikacije sastoji se od obrasca za prijavu, odnosno registraciju korisnika. Nakon prijave, korisnik ima pristup aplikaciji i podacima prikupljenim sa meteorološke postaje. Osim prikaza trenutne temperature i tlaka zraka, korisnik može pristupiti podacima iz prošlosti putem postavljenog filtera, može spremiti podatke za određene datume kako bi im mogao pristupiti u budućnosti, te može mijenjati postavke same aplikacije u vidu prikaza boja na grafikonima.

4.3.1. Prijava korisnika u sustav

Prilikom pokretanja aplikacije, svaki korisnik će prvenstveno biti preusmjeren na stranicu za prijavu korisnika. Kako bi korisnik mogao koristiti aplikaciju mora izvršiti prijavu. Stranica za prijavu se sastoji od obrasca za prijavu, u koji je potrebno unijeti korisničko ime i lozinku koje je korisnik odabrao prilikom registracije, te dva gumba, jedan za prijavu, koji u ovom slučaju ima funkciju „submit“ , a drugi za registraciju, koji u ovom slučaju preusmjerava korisnika na stranicu za registraciju.

Slika 12. - Prijava u sustav

Za provedbu prijave korisnika kreirana je funkcija „login“ koja na početku provjerava postoje li uopće uneseno korisničko ime i lozinka u bazi podataka, ako postoje i ako ta kombinacija odgovara onome što je u bazi, postavljaju se kolačići „COOKIE_USERNAME“ i

„COOKIE_USERID“ u trajanju od jedne godine, korisniku se prikazuje poruka „Korisnik uspješno logiran“, te ga se preusmjerava na stranicu „dashboard.php“. Ukoliko upisana kombinacija korisničkog imena i lozinke ne postoji u bazi podataka, korisniku se prikazuje poruka „Neuspješno logiranje!“, te ga se preusmjerava na stranicu „index.php“ gdje se ponovno može pokušati prijaviti u aplikaciju. Ako se dogodi situacija da je korisnik unio korisničko ime, ali nije lozinku ili obratno, prikazat će mu se poruka „Nepotpuni podaci“, te će biti preusmjeren na stranicu „index.php“ gdje ponovno može pokušati prijavu.

```
function login($link) {
   if (isset($_REQUEST['username']) && isset($_REQUEST['pass'])) {
      $encoded_pass = $_REQUEST['pass'];
      $sql2 = "SELECT * FROM users WHERE username='" .
$_REQUEST['username'] ."' AND password='" . $encoded_pass ."' LIMIT 1";
      $user = getQueryResult($sql2, $link, false);
      if (count($user)>0 and $user["username"] == $_REQUEST['username']) {
      setcookie(COOKIE_USERNAME,$user['username'], time()+60*60*24,
COOKIE_PATH, COOKIE_DOMAIN);
      setcookie(COOKIE_USERID,$user['id'], time()+60*60*24,COOKIE_PATH,
COOKIE_DOMAIN);
            $_SESSION["message"] = "Korisnik uspješno logiran!";
                  header('Location: dashboard.php');
      } else {
      setcookie(COOKIE_USERNAME, "", time()-60000, COOKIE_PATH,
COOKIE_DOMAIN);
      setcookie(COOKIE_USERID, "", time()-60000, COOKIE_PATH,
COOKIE_DOMAIN);
            $_SESSION["message"] = "Neuspješno logiranje!";
            header('Location: index.php');
      }
   } else {
      setcookie(COOKIE_USERNAME, "-", time()-60000, COOKIE_PATH,
COOKIE_DOMAIN);
      setcookie(COOKIE_USERID, "-", time()-60000, COOKIE_PATH,
COOKIE_DOMAIN);
      $_SESSION["message"] = "Nepotpuni podaci!";
      header('Location: index.php');
   }
}
```

4.3.2. Registracija novog korisnika

Ukoliko korisnik još nije registriran, na početnoj stranici aplikacije postoji gumb „Registracija“ koji ga preusmjerava na stranicu za registraciju. Stranica za registraciju korisnika sastoji se od jednostavnog obrasca u kojem se od korisnika traži unos osnovnih podataka, kao što su korisničko ime, e-mail adresa, lozinka te ponavljanje lozinke. Nakon što su ispunjena sva polja, obrazac se šalje pritiskom na gumb „Registracija“ koji u ovom slučaju ima funkciju „submit“.

Slika 13. - Registracija u sustav

Sve ove funkcionalnosti predviđene su kreiranom funkcijom „registration($link)“. Specifično za ovaj dio koda je upravo zahtjev za ponovnim upisivanjem željene lozinke, kako bi se korisnik uvjerio da je unio ispravnu lozinku, te da neće imati problema prilikom budućih prijava u aplikaciju. Provjera ponovljene lozinke vrši se pomoću „if“ uvjeta u kojem se provjerava odgovaraju li znakovi prvo upisane lozinke znakovima drugo upisane lozinke, te ako ne odgovaraju, korisniku se prikazuje poruka „Ponovljena lozinka nije ista!“ te ga se vraća na stranicu sa registracijskim obrascem. Ukoliko su obje unesene lozinke iste, te su popunjena

sva potrebna polja u obrascu, podaci iz registracijskog obrasca spremaju se u bazu podataka u pripadajuće retke. Korisniku se prikazuje poruka „Registracija uspjela, možete se logirati“, potom ga aplikacija preusmjerava na stranicu index.php Ukoliko korisnik nije ispunio sva potrebna polja u registracijskom obrascu, prikazuje mu se poruka „Nepotpuni podaci, probajte ponovo“, te ga se preusmjerava na stranicu index.php

```
function getHash($password)
{ return password_hash($password, PASSWORD_BCRYPT, ['cost' => 10]);}
function registration($link) {
      if   (isset($_REQUEST['username'])   &&   isset($_REQUEST['pass'])   &&
isset($_REQUEST['pass2'])) {
            if($_REQUEST['pass'] != $_REQUEST['pass2']) {
                  $_SESSION["message"] = "Ponovljena lozinka nije ista!";
                  header('Location: register.php');
                  }
      $encoded_pass = md5($_REQUEST['pass'].$_REQUEST['username']);
      $sql2 = "INSERT INTO users SET username='" . $_REQUEST['username'] ."',
password='"   .   $encoded_pass   ."',   email='"   .   $_REQUEST['email']   ."',
temp_colro='#ff0000', pressure_color='#0008ff'";
      $user = runQuery($sql2, $link);
setcookie(COOKIE_USERNAME, "-", time()-60000, COOKIE_PATH, COOKIE_DOMAIN);
setcookie(COOKIE_USERID, "-", time()-60000, COOKIE_PATH, COOKIE_DOMAIN);
      $_SESSION["message"] = "Registracija uspjela, možete se logirati";
      header('Location: index.php');
      } else {
setcookie(COOKIE_USERNAME, "-", time()-60000,COOKIE_PATH, COOKIE_DOMAIN);
setcookie(COOKIE_USERID, "-", time()-60000,COOKIE_PATH, COOKIE_DOMAIN);
      $_SESSION["message"] = "Nepotpuni podaci, probajte ponovo!";
            header('Location: index.php');
      }
}
```

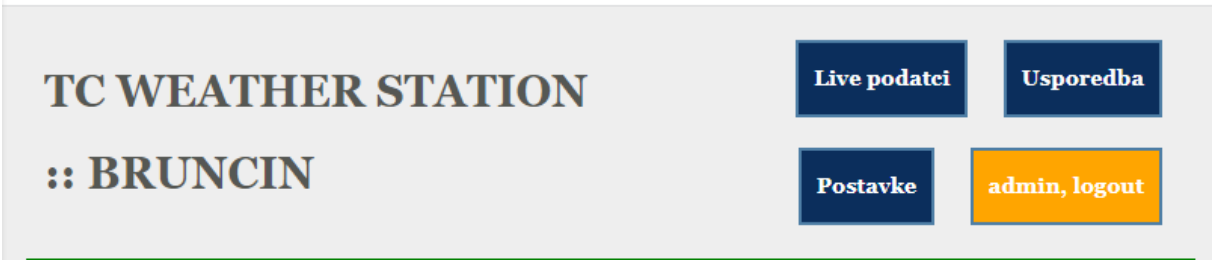
Podaci iz obrazaca šalju se POST metodom, kao najsigurnijom metodom obzirom da sakriva parametre koje šaljemo od potencijalnih napadača. Parametri se, kod POST metode nalaze u samom tijelu poruke, dok GET metoda sadrži parametre u URL-u, što ju čini ranjivom te se ne savjetuje korištenje GET metode prilikom slanja osjetljivih podataka. POST metoda, osim sigurnijeg prijenosa podataka, omogućava i slanje velikog broja parametara u samom zahtjevu.

4.3.3. Upravljačka ploča

Na svakoj stranici aplikacije nalazi se upravljačka ploča koja je smještena u zaglavlju stranice. Upravljačka ploča sastoji se od 4 gumba, od kojih je jedan za odjavu iz aplikacije, odnosno „logout“ gumb, koji se ističe drugačijom bojom od ostalih, a koristi se za odjavu korisnika iz aplikacije te ga preusmjerava na početnu stranicu, što je definirano funkcijom „logout()“. Ova funkcija ručno briše kolačiće iz korisničkog sustava tako da postavlja iste kolačiće pod imenima „COOKIE_USERNAME“ i „COOKIE_USERID“, ali bez vrijednosti te sa datumom isteka u prošlosti, odnosno datum isteka kolačića je trenutno vrijeme minus 60000 sekundi. Ovdje se može koristiti bilo koji negativan broj, ali zbog varijacija računalnih vremena, preporučuje se koristiti veći broj od -1. Nakon toga prikazuje poruku korisniku da je uspješno odjavljen te ga preusmjerava na stranicu index.php.

```
function logout() {
setcookie(COOKIE_USERNAME, "-", time()-60000, COOKIE_PATH, COOKIE_DOMAIN);
setcookie(COOKIE_USERID, "-", time()-60000, COOKIE_PATH, COOKIE_DOMAIN);
      $_SESSION["message"] = "Korisnik uspješno odjavljen";
      header('Location: index.php');
   }
```

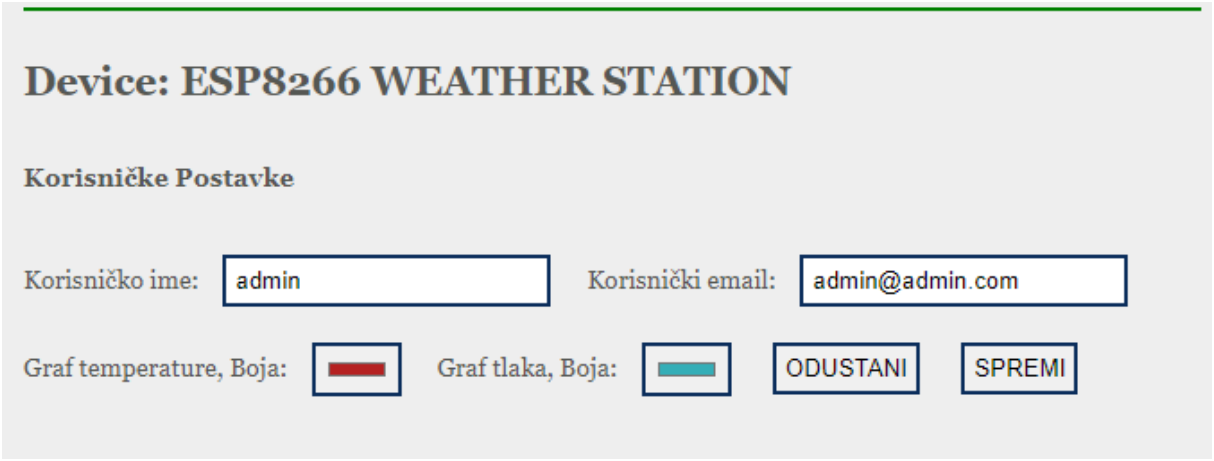
Gumb „Usporedba“, preusmjerava korisnika na stranicu „compare.php“ na kojoj je korisniku omogućena usporedba temperature i tlaka zraka do tri različita datuma. Podaci se povlače iz baze podataka, te se prikazuju na jednom grafikonu za temperaturu i na jednom grafikonu za tlak zraka, što omogućuje korisniku da vidi razliku u podacima za tražene datume. Funkcionalnosti ove stranice detaljnije će biti razrađene sljedećim poglavljima.



Slika 14. - Upravljačka ploča

4.3.4. Promjena korisničkih postavki

Treći gumb na upravljačkoj ploči je gumb „Postavke“ koji korisnika preusmjerava na stranicu „settings.php“ koja omogućava promjenu postavki korisnika putem obrasca. Stranica za promjenu postavki omogućava korisniku da aplikaciju prilagodi sebi, čime i sam korisnik postaje uređivač aplikacije. Korisniku je omogućena naknadna promjena korisničkog imena i e-mail adrese, a podaci se unose u polja za unos. U poljima za unos već je upisano trenutno korisničko ime i e-mail koji se dohvaćaju iz baze podataka u koju su pohranjeni prilikom registracije korisnika. Osim korisničkog imena i mail adrese, korisnik može promijeniti boju krivulja kojima se prikazuju podaci na grafikonu. Ovdje je korišten html input element „<input type=color >“ koji korisniku omogućuje određivanje boje koristeći vizualno sučelje za odabir boje. Ukoliko korisnik promijeni neku od tih komponenata, skripta će izvršiti upit „UPDATE“ prema bazi podataka i ažurirati željena polja. Posljednji gumb „Live podatci“ vraća nas na početnu stranicu „dashboard.php“ koju ću pobliže opisali u nastavku rada.



Slika 15. - Korisničke postavke

4.4. Opis dijela aplikacije za prikaz podataka

Stranica „dashboard.php“ je glavna stranica aplikacije te ona prikazuje najnovije podatke prikupljene sa meteorološke postaje. Ova stranica u ovom obliku vidljiva je samo prijavljenim korisnicima, a ako neprijavljeni korisnik pokuša izravno ući na tu stranicu (upisujući u URL punu stazu), skripta će ga preusmjeriti na stranicu sa obrascem za prijavu. To smo postigli uvjetom „if“ koji je priložen u nastavku teksta.

```
if(!isset($_COOKIE[COOKIE_USERNAME])||!isset($_COOKIE[COOKIE_USERID])){
   header("Location: index.php");}
```

Na stranici je vidljiv prikaz temperature i tlaka zraka unutar tablice i kao grafikon sa dvije krivulje. Prema zadanim postavkama ukoliko je sve u redu sa senzorom, priljevom podataka i konekcijom na bazu podataka, prilikom učitavanja stranice, u tablici se prikazuje trenutno stanje temperature i tlaka zraka u prostoru gdje je postavljena meteorološka postaja. Grafikon, prema zadanim početnim postavkama, krivuljama prikazuje stanje temperature i tlaka zraka u proteklih 24 sata. Korisnik ima mogućnost prikaza podataka o temperaturi i tlaku zraka za odabrani period, na način da u polja za unos datuma upiše željene datume ili da ih odabere na kalendaru koji će se prikazati prilikom pritiska mišem na polje za unos datuma. Nakon pritiska gumba „View“, na grafikonu će se prikazati krivulje koje predstavljaju podatke o temperaturi i tlaku zraka u odabranom periodu, dok će se u tablici i dalje prikazivati posljednji spremljeni dobavljivi podaci iz baze podataka, odnosno sa senzora. Krivulje su različitih boja, a korisnik može u postavkama iste promijeniti po vlastitom odabiru, što je detaljnije objašnjeno u poglavlju 4.2.4. Promjena korisničkih postavki.

4.4.1. Prikaz podataka u tablici

Tablični prikaz podataka sa meteorološke postaje zamišljen je i izveden na način da prikazuje zadnje izmjerene podatke dohvatljive iz baze podataka te informacije o vremenu očitanja, uz prikaz proteka vremena od zadnjeg očitanja sa senzora i dohvaćanja iz baze podataka.

```
<?php
list($DateTimeDiff,$DateTimeColor)=DateDiff($ypr_last_record["DateTime"]);
?>
<table class="table">
      <tr class="row header"><td colspan="2">Last record</td></tr>
      <tr class="row"><td>Datetime</td><td><?php echo
```

```
            $ypr_last_record["DateTime"]; ?> <?php echo "<font color='" .
            $DateTimeColor ."'>" . $DateTimeDiff ."</font>"; ?></td></tr>
      <tr class="row"><td>Temperature</td><td><?php echo
            $ypr_last_record["Temperature"]; ?> °C</td></tr>
      <tr class="row"><td>Pressure</td><td><?php echo
            $ypr_last_record["Pressure"]; ?> HPa</td></tr>
</table>
```

Kako bi omogućili prikaz proteka vremena od zadnjeg očitanja sa senzora, treba dohvatiti taj podatak, izračunati razliku od zadnjeg očitanja do trenutka učitavanja stranice. Upravo pomoću tog podatka možemo saznati radi li meteorološka postaja ispravno. Ako je razlika u proteku vremena velika, možemo pretpostaviti da nešto nije u redu sa senzorom ili nema internetske veze putem koje meteorološka postaja komunicira sa bazom podataka. Kod prikaza proteka vremena implementirane su različite boje kako bi podatak korisniku bio još više uočljiv, one se također primjenjuju prilikom filtriranja određenog perioda. Crvena ako je prošlo više od 48 sati, žuta ako je prošlo više od 12 sati, ali manje od 48 sati te zelena ako je prošlo manje od 12 sati. Također, definiran je način tekstualnog prikaza proteka vremena, ovisno radi li se o minutama, satima ili danima.

```
function DateDiff($data_date) {
   $now_date = date('H:i T d/m/Y');
   $now_year = date('Y');
   $now_month = date('m');
   $now_day = date('d');
   $now_hour = date('H');
   $now_min = date('i');
   $now_sec = date('s');
   $time_parts = strptime($data_date, '%Y-%m-%d %H:%M:%S');
   $local_time =
date('U',mktime($now_hour,$now_min,$now_sec,$now_month,$now_day,$now_year);
   $data_time =
date('U',gmmktime($time_parts['tm_hour'],$time_parts['tm_min'],$time_parts[
'tm_sec'],($time_parts['tm_mon']+1),$time_parts['tm_mday'],($time_parts['tm
_year']+1900)));
   $time_diff = $local_time-$data_time;
   if($time_diff < 0) { $bg_color="#FF3300";  $datetimestr = "Time Parsing
Error"; return array($datetimestr, $bg_color); }
   $minute = sprintf("%02d", $time_diff/60);
   $hour = sprintf("%02d", $time_diff/(60*60));
   $day = sprintf("%02d", $time_diff/(60*60*24));
```

```
   if ($minute<120)  { if($minute <= 1) { $datetimestr = "$minute minute
ago"; } else { $datetimestr = "$minute minutes ago"; } }
   elseif ($hour<25) { $minute = sprintf("%02d", ($minute - ($hour*60)));
if($hour>8) $bg_color="#FF3300"; else $bg_color="#FFFF00"; $datetimestr =
"$hour:$minute hours ago"; }
   else {  if($day == 1) { $datetimestr = "$day day ago"; } else {
$datetimestr = "$day days ago"; } }
   if($hour <= 12) {$bg_color="#00FF00";}
   else if($hour > 12 && $hour <= 48) { $bg_color="#FFFF00"; }
   else { $bg_color="#FF3300"; }
   return array($datetimestr, $bg_color);
   }
```

Last record	
Datetime	2022-08-07 15:44:08 00 minute ago
Temperature	26.63 °C
Pressure	1003.47 HPa

Last record	
Datetime	2022-08-05 23:59:05 01 day ago
Temperature	27.47 °C
Pressure	999.66 HPa

Last record	
Datetime	2022-08-04 23:59:04 02 days ago
Temperature	27.69 °C
Pressure	1002.02 HPa

Slika 16. - Razlikovanje boja kod proteka vremena

4.4.2. Prikaz podataka na grafikonu

Grafikon prikazuje svako očitanje sa senzora kao jednu točku, a sve te točke povezane su linijom te tako tvore grafikon. Prelaskom mišem preko određene točke, otvara se mali prozorčić koji prikazuje detaljnije podatke o tom očitanju (vrijeme i podatke o temperaturi i tlaku zraka), a moguće je i približiti (eng. *zoom in*) prikaz kako bi se lakše vidjele i najmanje promjene u podacima.

Grafikon temperature i tlaka zraka izrađen je pomoću Google Visualisation framework-a. Temeljen na JavaScriptu omogućava širok spektar grafikona koji se mogu ugraditi na web stranice. Trenutno postoje dva načina za učitavanje JavaScript biblioteke za Google Charts.

Gstatic loader [2] je novijeg datuma, te se očekuje potpuni prelazak na ovaj način učitavanja, a postoji i jsapi [3] koji je korišten prilikom izrade ove aplikacije. Za uspješnu implementaciju Google Chart grafikona, potrebno je izvesti nekoliko koraka. („google-visualisation-tutorial“, bez dat.)

Prvo uključujemo loader sa oznakom skripte:

```
<script type="text/javascript"src="https://www.google.com/jsapi"></script>
```

Kada smo uključili program za učitavanje, koristimo ga za učitavanje određenih paketa biblioteka pozivanjem funkcije za učitavanje:

```
<script type="text/javascript">
google.load('visualization', '1.0', {'packages':['corechart']});
var $YprData = <?php echo json_encode($ypr_data); ?>;
var $temp_color = '<?php echo $user["temp_color"]; ?>';
var $pressure_color = '<?php echo $user["pressure_color"]; ?>'; </script>
```

Prilikom učitavanja paketa potrebno je pričekati da potpuno završe s učitavanjem kako bi ih se moglo koristiti. Čekanje se postavlja putem povratnog poziva uz pomoć funkcije setOnLoadCallback:

```
google.setOnLoadCallback(function() {
try {
      drawTemperaturePlot($YprData);
} catch(e) {
}
});
```

Unutar te funkcije, implementirali smo funkciju kao svojevrsnu zaštitu od rušenja stranice zbog nemogućnosti dohvaćanja podataka iz baze ili dohvaćanja neiskoristivih podataka. Ukoliko se nemamo dobre podatke iz baze, a pokušavamo ih implementirati u grafikon, ovom funkcijom osiguravamo da se neće srušiti stranica i sve će funkcionirati kako očekujemo, samo će ova JS skripta stati i neće se izvoditi. Na primjer, pokazat će nam se prazan grafikon ili će se prikazati stari podaci, ako se ne uspiju dohvatiti novi, a stari podaci postoje u bazi.

[2] https://www.gstatic.com/charts/loader.js
[3] https://www.google.com/jsapi

Naziv vizualizacije klase je „google.visualization.LineChart“, dodjeljujemo mu jedinstveni ID "airTempHumPlot" pomoću kojeg ćemo ga pozvati unutar html elementa kako bi ga prikazali na web-u.

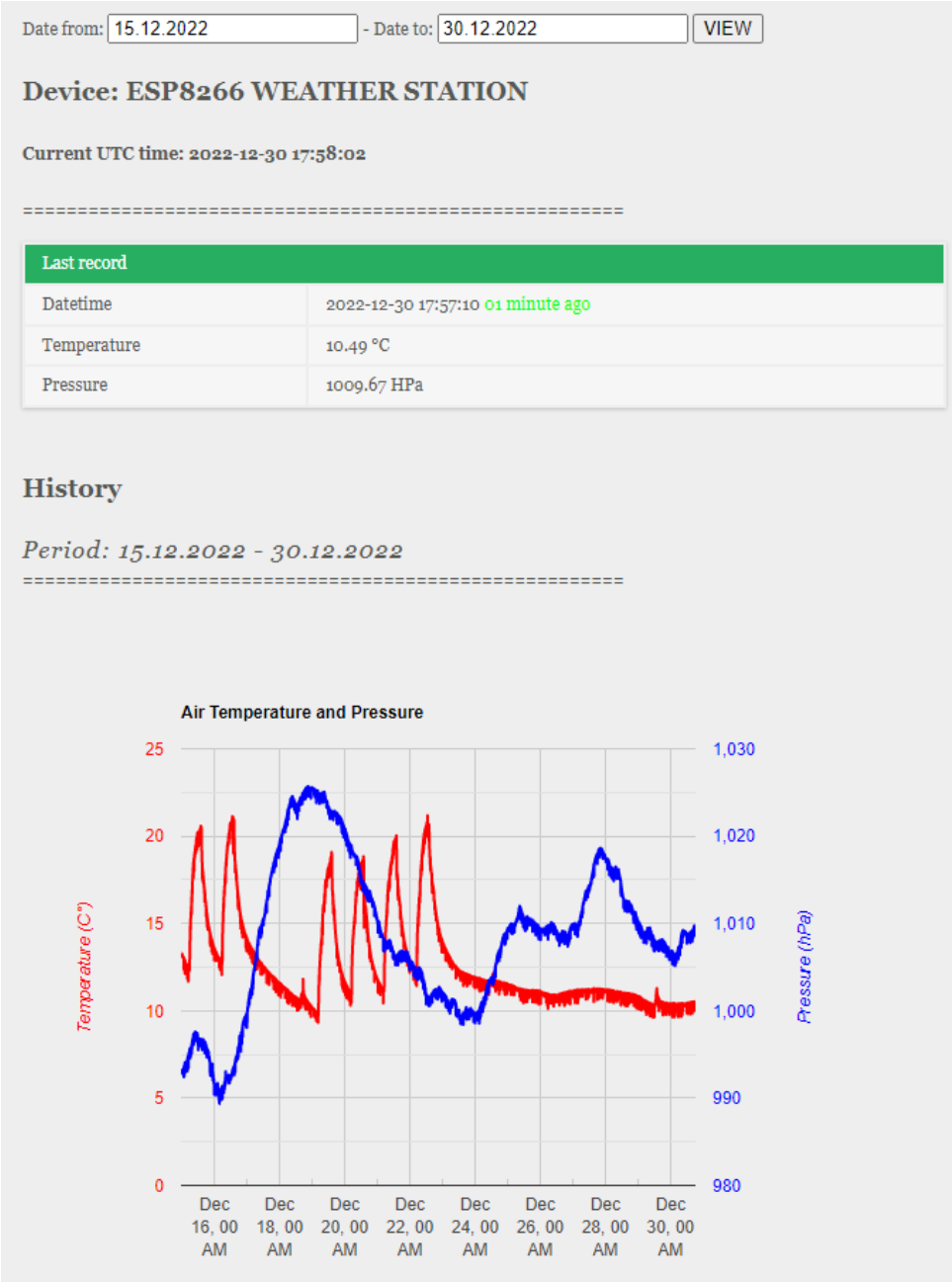
```
var chart_1=
newgoogle.visualization.LineChart(document.getElementById('airTempHumPlot';
```

Unutar te funkcije grafikonu određujemo dimenzije, naslov, mogućnosti pomicanja, uvećavanja i smanjivanja, tekst, boju itd.

```
chart_1.draw(data_1, {curveType: "function",
explorer: {
maxZoomOut:2,
keepInBounds: true},
width: '100%', height: 500,
title: 'Air Temperature and Pressure',
hAxis:{slantedText:'false',slantedTextAngle:90,format:'MMM d,HH a'},
series:{0:{color:'red',targetAxisIndex:0},1:{color:'blue',targetAxisIndex:1
}},
vAxes:[{title:'Temperature(C°)',textStyle:{color:'red'},
titleTextStyle:{color: 'red'}},
{title:'Pressure(hPa)',textStyle:{color:'blue'},titleTextStyle:{color:'blue
'}}],
legend: 'none',
backgroundColor: '#EEEEEE',
fontSize: 12,
'fontName': '"Arial"'});
```

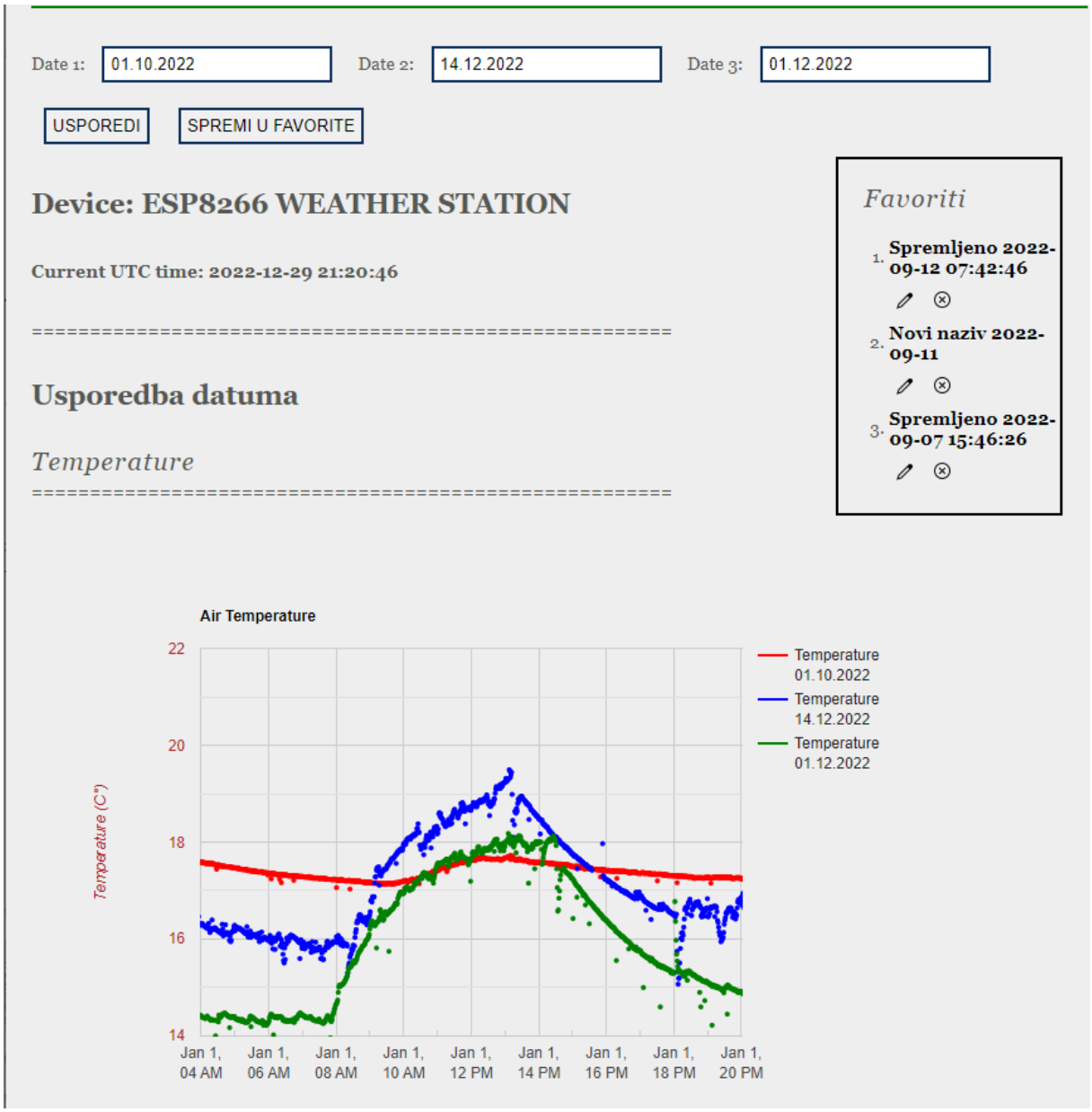
U html-u prikazujemo grafikon pomoću elementa, identificirajući ga sa jedinstvenim ID-jem koji smo dodjelili tom grafikonu u prethodnom koraku. Grafikon prikazuje podatke iz perioda kojeg je korisnik odabrao u kalendaru, odnosno prilikom prvog učitavanja stranice uzima se trenutni datum kao završni, a jedan dan prije kao početni parametar za prikaz grafikona.

```
<h2>History</h2>
<h3>Period: <?php echo $_GET["startDate"] . " - " . $_GET["endDate"];
?></h3>
<p>=====================================================</p>
<div class="cplot" id="airTempHumPlot">PRESSURE / TEMPERATURE</div>
<p>=====================================================</p>
```



Slika 17. - Prikaz web aplikacije

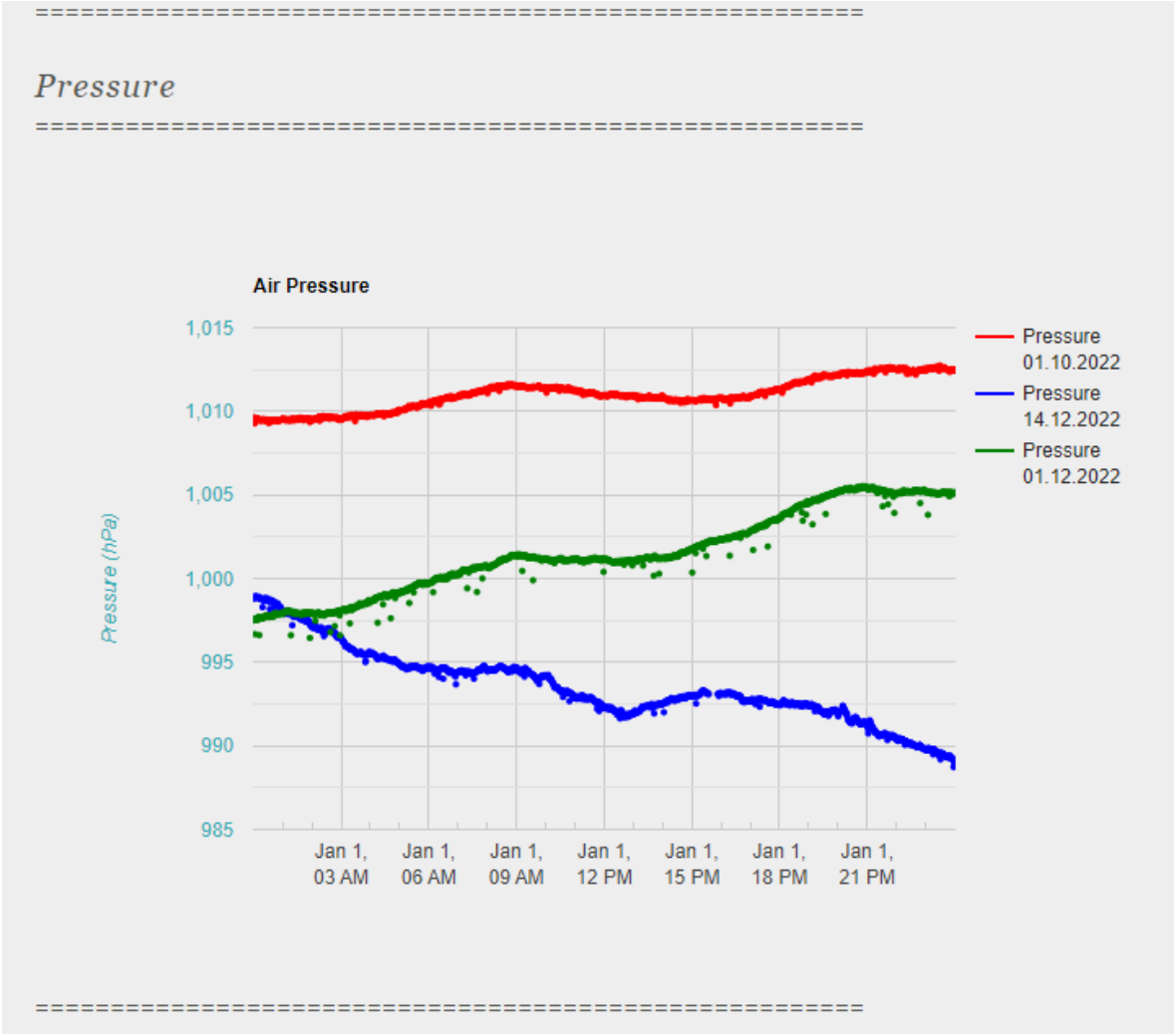
U aplikaciji je korisnicima dostupna funkcionalnost usporedbe odabranih datuma, odnosno podataka o temperaturi i tlaku zraka na odabrane dane. Kako bi se pokrenula usporedba, potrebno je da korisnik unese do 3 datuma u za to predviđena polja obrasca ili odabere datume na kalendaru koji se otvara pritiskom mišem na polje za unos datuma, a zatim pritisne gumb „Usporedi“. Aplikacija prikazuje podatke o temperaturi na jednom grafikonu, a podatke o tlaku zraka na drugom grafikonu. Na svakom grafikonu prikazuje se onoliko krivulja koliko je korisnik unio datuma za usporedbu u za to predviđena polja, pri čemu je svaka krivulja različite boje.



Slika 18. - Usporedba datuma – temperatura zraka

Prilikom korištenja funkcionalnosti usporedbe određenih datuma, korisnicima je omogućena i funkcionalnost spremanja tih pretraga u „omiljene“, kako bi što jednostavnije pristupili tim podacima, bez potrebe ponovnog upisivanja datuma ili odabira datuma na kalendaru. Nakon što su unijeli odabrane datume u za to predviđena polja, pritiskom na gumb „Spremi u favorite“, odabrana pretraga se sprema u bazu podataka sa id-jem tog korisnika i id-jem te pretrage, te je vidljiva unutar okvira „Favoriti:“ u desnom gornjem kutu ekrana. Zadano ime spremljenog favorita je datum i vrijeme spremanja, ali korisnici mogu promijeniti naziv omiljene spremljene pretrage, pa čak i promijeniti datume koji se uspoređuju te izbrisati nepotrebne ili nepoželjne pretrage iz „favorita“. Do tog dijela aplikacije dolazi se pritiskom na

ikonicu olovke unutar okvira „Favoriti:“, prilikom čega se korisnika preusmjerava na stranicu sa id-jem spremljenog favorita, te je tamo moguće obaviti sve navedene promjene.



Slika 19. - Usporedba datuma - tlak zraka

5. Kritički osvrt na razvoj aplikacije

Sama aplikacija nije veoma zahtjevna za razvoj. Temelji se na relativno jednostavnim funkcionalnostima php-a, JavaScript-a i html-a, uz pomoć web okvira i biblioteka programskog koda. Sve eventualne pogreške ili nedoumice mogu se riješiti uz pretragu foruma i web mjesta koja se bave tom tematikom. Korištenjem web okvira, ubrzao se proces izrade jer su mnogi elementi izrađeni, te sam time izbjegla samostalnu izradu koja bi potrajala puno dulje, a i bilo bi puno teže za izradu.

Kao neki najveći nedostatak aplikacije, ali i problem na koji sam naišla prilikom izrade, vezan je za dio aplikacije koji omogućava usporedbu temperature i tlaka zraka za do tri različita datuma, odnosno dana. Tri krivulje različite boje prikazuju se na jednom grafikonu gdje svaka točka na krivulji predstavlja jedan pristigli paket podataka sa temperaturom/tlakom zraka. Vrijeme pristizanja paketa pozicionirano je na horizontalnoj, a podatak o temperaturi/tlaku zraka na vertikalnoj osi, dok sjecište tih dviju linija predstavlja jednu točku. Kako paketi podataka pristižu u različito vrijeme, odnosno ne stižu uvijek u istom satu, minuti i sekundi, te točke ne nalaze se na istim pozicijama za različite datume. To se najbolje može vidjeti kada se klikne mišem na jednu točku na krivulji grafikona, gdje će se u skočnom prozoru (eng. *pop-up window*) prikazati brojčana vrijednost te točke, odnosno podatak o temperaturi koji je pristigao u paketu u neko određeno vrijeme, dok će se brojčana vrijednost za drugi ili druga dva datuma u to vrijeme prikazati kao „NULL“ jer u toj sekundi nema podatka, a koji je možda stigao sekundu prije ili kasnije te samim time ne leži na istoj osi vremena. Iako je vizualno vidljiva razlika u krivuljama i može se usporediti, za neki ozbiljniji znanstveni rad ili matematički odnosno algoritamski dokaz razlike u podacima, ovo ne bi bilo dovoljno precizno. Za rješavanje ovog problema, može se pristupiti sa softverske, ali i hardverske strane.

Što se softvera tiče, trebalo bi uložiti puno više vremena za razvoj algoritma koji bi računao vrijednost točke koja nam nedostaje u vertikalnoj liniji. Algoritam pak može biti razvijen na jednostavniji ili kompliciraniji način. Najjednostavniji način bio bi računanje prosječne vrijednosti (eng. *average*) između dvije najbliže točke, odnosno najbliže točke s lijeve strane i najbliže točke s desne strane, te taj prosjek uzeti kao točku koja nedostaje. Kompliciraniji način je računanje pozicije točke preko nagiba i omjera udaljenosti od dviju najbližih točki s lijeve i desne strane, ako pretpostavimo da se radi o linearnom grafikonu, premda je u realnosti riječ o nekoj vrsti krivulje, kao što se može vidjeti iz samih podataka.

Algoritam bi vjerojatno trebalo aplicirati kroz JavaScript kako bi bio moguć izračun u stvarnom vremenu (eng. *realtime*). Konkretno kada korisnik prijeđe pokazivačem miša preko tražene točke, algoritam nalazi najbliži lijevi i najbliži desni podatak, te provodi izračun za pronalazak točke ili točaka koje nedostaju, te prikazuje rezultat izračuna kao točku na grafikonu, uz brojčani prikaz u skočnom prozoru. Ovdje nam pak ostaje problem što to onda ne predstavlja stvarni podatak nego rezultat izračuna algoritma, te također ne bi bilo prihvatljivo za neki ozbiljan znanstveni dokaz.

Kod pristupa rješavanju ovog problema sa hardverske strane, postoji mogućnost korištenja modula sata stvarnog vremena (eng. *real time clock - RTC*), koji bi putem alarma koji radi na principu prekida (eng. *interrupt*), dao signal GPIO-u koji budi platformu odnosno mikrokontroler, odradi mjerenje te zatim pošalje podatke. Na ovaj način podaci bi bili poslani u istoj minuti te bi sve točke bile postavljenje na istoj osi, što bi rezultiralo preciznim podacima za usporedbu razlike u temperaturi odnosno tlaku zraka u određenom trenutku.

6. Zaključak

Izrađena je kućna meteorološka postaja za mjerenje temperature i tlaka zraka te je za nju napravljena web aplikacija koja prikuplja, sprema i prikazuje podatke. Kako bi se proizvelo jednostavno i jeftino rješenje, hardverski dijelovi su nabavljeni u lokalnim trgovinama elektroničkom robom, te putem internetskih trgovina (AliExpress.com). Korišten je senzor tlaka i temperature zraka BME280 napojen modulom ESP8266. Izrađena je shema spajanja senzora i modula. Sastavljen je prilagodni sklop koji regulira napajanje modula ESP8266 kako ne bi bio ovisan o spajanju na računalo, nego se može priključiti u bilo koju utičnicu koja isporučuje 220V. Modul ESP8266 spojen je na bežičnu mrežu (WiFi) putem koje šalje podatke sa senzora u bazu podataka.

Pri izradi aplikacije korišteni su HTML, PHP, JavaScript, CSS, a senzori su programirani u programskom jeziku Python. Sav programski kod pisan je u programu Notepad++. Za vizualizaciju podataka putem grafikona korišten je Google Charts API, koji je uz jQuery i biblioteke koda za senzore olakšao izradu aplikacije. U radu su objašnjeni svi koraci za implementaciju Google Charts grafikona za prikaz prikupljenih podataka.

Aplikacija prikazuje podatke u realnom vremenu (±1 minuta), uz mogućnost prikazivanja podataka iz povijesti, usporedbu podataka za do tri različita datuma, te spremanje određenih datuma u favorite za kasniju bržu upotrebu. To omogućuje baza podataka u koju se spremaju podaci prikupljeni sa senzora sa lokalnim vremenom poslužitelja kao vremenom očitanja. Podaci se pozivaju iz baze uz pomoću upita kojim se pretražuje određeni period između datuma ili za datume koje je korisnik unio u predviđena polja u aplikaciji. Aplikacija prikazuje podatke o temperaturi i tlaku zraka u tablici i na grafikonima. Podaci koji se prikazuju u tablici su posljednje dohvaćeni podaci iz baze podataka, te se uz podatke sa postaje, prikazuje i protek vremena od trenutka slanja upita do vremena posljednje dohvatljivog podatka iz baze. Osim što se prikazuje tekstualno, protek vremena prikazuje se i u različitim bojama – što je duži period proteka vremena tako se mijenjaju boje od zelene preko žute do crvene, kako bi odmah ukazao korisniku na eventualni problem sa senzorom. Podaci na grafikonu su prikazani također u dvije boje kako bi korisniku bila vidljivija razlika između dvaju različitih vrsta podataka. Kao dodatna funkcionalnost za korisnika, omogućen je izbor boja u kojima se prikazuju podaci na grafikonima.

Unatoč relativno jednostavnoj implementaciji Google Charts grafikona u web aplikaciju, naišla sam na problem sinkronizacije podataka za različite datume. Obzirom da vrijeme pristizanja paketa podataka uglavnom nije isto, te varira u nekoliko sekundi prije i poslije, što stavlja podatke na različite vremenske osi u grafikonu, nije moguće izvesti precizan matematički ili algoritamski dokaz iz ovih podataka.

Meteorološka postaja i pripadajuća aplikaciju ostavljaju mogućnost daljnjeg razvoja i ozbiljnijeg pristupa ukoliko za to bude volje i potrebe, u smislu izrade pripadajućeg kućišta, obogaćivanja aplikacije novim funkcionalnostima i razvoj mobilne verzije aplikacije, a nije isključena ni komercijalna primjena ako se za to ukaže prilika.

Aplikacija[4] je dostupna javnosti, a za korištenje aplikacije potrebno se registrirati kao korisnik, uz popunjavanje jednostavnog obrasca za registraciju.

[4] http://bruncin.ydns.eu/weather_station/

Popis literature

1. Chaffer, J. (2013). *Learning jQuery—Fourth Edition*. Packt Publishing Ltd.
2. *Chart Gallery | Charts*. (bez dat.). Google Developers. Preuzeto 07. siječanj 2023., od https://developers.google.com/chart/interactive/docs/gallery
3. *Chart.js | Chart.js*. (bez dat.). Preuzeto 05. veljača 2023., od https://www.chartjs.org/docs/latest/
4. *Charts*. (bez dat.). Google Developers. Preuzeto 19. srpanj 2022., od https://developers.google.com/chart
5. *D1 Mini WiFi Development Board*. (bez dat.). www.addicore.com. Preuzeto 07. kolovoz 2022., od https://www.addicore.com/product-p/ad318.htm
6. Ember js vs Angular js | 6 Most Valuable Differences You Should Know. (2018, listopad 15). *EDUCBA*. https://www.educba.com/ember-js-vs-angular-js/
7. *Google Charts vs Chart.js*. (bez dat.). Fusioncharts.Com. Preuzeto 05. veljača 2023., od https://www.fusioncharts.com
8. *GY-BME280-5V-Temperature-and-Humidity-Sensor-1.jpg (800×800)*. (bez dat.). Preuzeto 29. siječanj 2023., od https://robu.in/wp-content/uploads/2020/10/GY-BME280-5V-Temperature-and-Humidity-Sensor-1.jpg
9. *JQuery Introduction*. (bez dat.). Preuzeto 10. kolovoz 2022., od https://www.w3schools.com/jquery/jquery_intro.asp
10. *PHP: MySQL Database*. (bez dat.). Preuzeto 07. kolovoz 2022., od https://www.w3schools.com/php/php_mysql_intro.asp
11. *PHP: MySQL (Original)—Manual*. (bez dat.). Preuzeto 07. kolovoz 2022., od https://www.php.net/manual/en/book.mysql.php

12. *Protis—WeMos D1 Mini, WiFi modul sa ESP8266, s konektorom za antenu*. (bez dat.). Preuzeto 09. kolovoz 2022., od https://www.protis.hr/products/details/wemos-d1-mini-wifi-modul-sa-esp8266-s-konektorom-za-antenu/171122
13. *Prototyp Schild für Wemos D1 Mini IoT Blynk ESP8266 Arduino Schaltung Proto*. (bez dat.). eBay. Preuzeto 29. siječanj 2023., od https://www.ebay.de/itm/331919055342
14. Santos, R. (2022). *MicroPython Programming with ESP32 and ESP8266 eBook* [Python]. https://github.com/RuiSantosdotme/ESP-MicroPython/blob/d2c83890c18f0a9d88302643faed704c235daf41/code/WiFi/HTTP_Client_IFTTT_BME280/BME280.py (Original work published 2018)
15. *Usage Statistics and Market Share of jQuery for Websites, January 2023*. (bez dat.). Preuzeto 07. siječanj 2023., od https://w3techs.com/technologies/details/js-jquery
16. *Visualization: Line Chart (Image) | Charts*. (bez dat.-a). Google Developers. Preuzeto 03. kolovoz 2022., od https://developers.google.com/chart/interactive/docs/gallery/imagelinechart
17. *Visualization: Line Chart (Old Version) | Charts*. (bez dat.-b). Google Developers. Preuzeto 03. kolovoz 2022., od https://developers.google.com/chart/interactive/docs/gallery/linechart_old
18. *W3Techs—Extensive and reliable web technology surveys*. (bez dat.). Preuzeto 07. veljača 2023., od https://w3techs.com/
19. www.e-elektro.com.hr: Što je Arduino? (bez dat.). *www.e-elektro.com.hr*. Preuzeto 07. siječanj 2023., od http://e-elektro.blogspot.com/2014/06/sto-je-arduino.html
20. Zhu, Y. (2012). Introducing Google Chart Tools and Google Maps API in Data Visualization Courses. *IEEE Computer Graphics and Applications*, *32*(6), 6–9. https://doi.org/10.1109/MCG.2012.114

Popis slika

Slika 1. - Statistika korištenja JavaScript bibilioteka (Prema: W3Techs.com, 2023.) 5
Slika 2. - Google Charts tipovi grafikona (Prema: Google Developers, 2023.) 9
Slika 3. - Senzorski sustav meteorološke postaje (Izvor: autorski rad) 11
Slika 4. - BME280 senzor (Izvor: robu.in, 2020.) 12
Slika 5. - ESP8266 D1 Mini WiFi Dev Bord (Izvor: addicore.com, 2023.) 13
Slika 6. - Raspored pinova ESP8266 D1 Mini pločice (Izvor: addicore.com, 2023.) 14
Slika 7. - Dodatna (shield) pločica (Izvor: ebay.de) 15
Slika 8. - Shema spajanja senzora (Izvor: autorski rad) 16
Slika 9. - Shema prilagodnog sklopa (Izvor: autorski rad) 17
Slika 10. - ERA dijagram baze podataka (Izvor: autorski rad) 22
Slika 11. - Arhitektura sustava za vizualizaciju meteoroloških podatka (Izvor: autorski rad) .. 24
Slika 12. - Prijava u sustav 26
Slika 13. - Registracija u sustav 28
Slika 14. - Upravljačka ploča 31
Slika 15. - Korisničke postavke 31
Slika 16. - Razlikovanje boja kod proteka vremena 34
Slika 17. - Prikaz web aplikacije 37
Slika 18. - Usporedba datuma – temperatura zraka 38
Slika 19. - Usporedba datuma - tlak zraka 39

Popis tablica

Tablica 1. - Najpopularnije JavaScript biblioteke (Izvor: w3techs.com)..... 3
Tablica 2. - Najbrže rastuće JavaScript biblioteke (Izvor: w3techs.com)..... 4
Tablica 4. - Usporedba značajki Chart.js i Google Charts (Prema: fusioncharts.com)..... 8

Prilozi

Ovom radu priložen je programski kod aplikacije u zip arhivi.