PD.TVZ.HR
POLYTECHNIC & DESIGN Vol. 8, No. 3, 2020.
DOI: 10.19279/TVZ.PD.2020-8-3-02

IMPLEMENTACIJA HRANILICE ZA KUĆNE LJUBIMCE UPRAVLJANE POMOĆU MREŽNE APLIKACIJE I UGRAĐENOG SUSTAVA

PET FEEDER AUTOMATIZATION BASED ON EMBEDDED SYSTEM CONTROLLED BY WEB APPLICATION

Mario Lukec[1], Ivica Dodig[2], Jelena Kapelac Gulić[2], Brigitta Cafuta[2], Ognjen Mitrović[2]

[1]Tehničko veleučilište u Zagrebu, Vrbik 8, 10000 Zagreb, Hrvatska, Student
[2]Tehničko veleučilište u Zagrebu, Vrbik 8, 10000 Zagreb, Hrvatska

SAŽETAK

U radu je dan pregled pogodnih mikrokontrolera za implementaciju hranilice za kućne ljubimce. Također navedena su njihova osnovna obilježja te su odabrani najoptimalniji mikrokontroleri za implementaciju. Kao najpogodniji mikrokontroleri opisana je Arduino platforma i ESP32.
Poseban naglasak usmjeren je na povezivanje mikrokontrolera s mrežnom aplikacijom. Za povezivanje putem mrežne aplikacije prikazani su razvojni okviri za Javu te je dan pregled potrebnog okruženja za njenu izradu. Kao alati za jednostavniju implementaciju primijenjeni su Spring Boot i React, koji opisuju sve potrebne elemente i konvenciju iz prakse kreiranja mrežne aplikacije. Prikazano je potrebno sklopovlje za implementaciju hranilice za kućne ljubimce te je i izrađena hranilica za kućne ljubimce upravljiva putem mrežne aplikacije.

Ključne riječi: *arduino, ESP32, mikrokontroler, hranilica, WiFi modul, Spring Boot, React*

ABSTRACT

The paper provides an overview of suitable microcontrollers for the implementation of a pet feeder. Their basic features are also listed and the most optimal microcontrollers for implementation are selected. The Arduino platform and ESP32 have been described as the most suitable microcontrollers. Special emphasis is placed on connecting the microcontroller to the web application. For connecting via a web application, development frameworks for Java are presented and an overview of the necessary environment for its development is given.
Spring Boot and React were used as tools for easier implementation, which describe all the necessary elements and conventions from the practice of creating a web application. The hardware for the implementation of the pet feeder is shown, and the made pet feeder is manageable via a network application.

Keywords: *arduino, ESP32, microcontroller, pet feeder, WiFi module, Spring Boot, React*

1. UVOD
1. INTRODUCTION

U današnje vrijeme, mikrokontroleri se sve više koriste u svakodnevnom životu, koji se povezuju na internet. Što se može primijetiti velikim rastom IoT industrije. Nekada je bilo nezamislivo da će se uređaji poput hladnjaka, perilica rublja moći povezivati sa web servisima, što je danas slučaj da ih sve više koriste IoT, odnosno povezani su na internet. Kako bi drugi uređaji ili korisnici mogli sa njima udaljeno komunicirati od bilo kuda i u bilo koje vrijeme, bilo da koriste neki preglednik ili mobilno sučelje. Danas se na bilo kojoj internet tražilici lako može pronaći veliki broj zanimljivih „kućnih“ projekata gdje se mikrokontroleri poput Arduina, Raspberry Pi, ESP32 koriste za upravljanje nekim sustavom. Takve sustave zgodno je i praktično povezati na internet, te ih nadzirati i upravljati preko nekog mrežnog (web) sučelja.
Ovaj rad opisat će cijeli postupak izrade platforme, koji se sastoji se od Arduino mikrokontrolera, koji za povezivanje na internet koristi ESP32, još nekoliko elektroničkih sklopova, te web aplikacije za nadzor i upravljanje

Koja koristi backend baziran na Spring Boot programskom okviru za programski jezik Java, te frontend baziran na React programskom okviru za Javascript. Funkcionalnost ovog sustava je kontrola i analiza prehrane kućnih ljubimaca putem mrežne (web) aplikacije. U posudama se konstantno važe masa vode i hrane, te se ti podaci šalju na poslužitelj gdje se spremaju i omogućuju korisniku da vidi razne statistike putem web aplikacije.

U drugom poglavlju opisana je Arduino platforma, od programskog jezika do mikrokontrolera i njegovih verzija, te ESP32 mikrokontroler. Treće poglavlje podijeljeno je na dvije glavne cjeline – Spring Boot i React, te opisuju sve potrebne elemente i konvenciju iz prakse kreiranja mrežne aplikacije. Četvrto poglavlje pokazuje programsko rješenje spajanja Arduino Uno i ESP32 međusobno i sa poslužiteljem, odnosno mrežnom aplikacijom, uz odgovarajuće primjere. Peto poglavlje donosi kompletan postupak implementacije glavne teme ovoga rada, hranilice za kućne ljubimce upravljane pomoću mrežne aplikacije i ugrađenog sustava, te su dostupne i slike konačnog rješenja sklopovlja i mrežne aplikacije.

2. ARDUINO I ESP32 MIKROKONTROLERI

2. ARDUINO AND ESP32 MICROCONTROLLERS

Arduino je open-source platforma koja je jednostavna za korištenje. Arduino programski jezik baziran je na programskom jeziku C.

Glavna komponenta je 8-bitni mikrokontroler koji se nalazi u srcu Arduino pločice. Arduino platforma je skup elektroničkih i softverskih komponenti koje se mogu jednostavno povezivati u složenije cjeline s ciljem izrade zabavnih i poučnih elektroničkih rješenja. Pošto je Arduino platforma open-source tipa, dozvoljeno je njezino dijeljenje i preuređivanje u svrhu kreiranja novih platforma koje su međusobno kompatibilne tako da su razvojem nastale još mnoge inačice razvojnih okruženja baziranih na ovoj platformi. Osim mikrokontrolera, na Arduino pločici nalaze se i druge komponente poput integriranog sklopa za komunikaciju s računalom, perifernih elektroničkih dijelova za osiguravanje mogućnosti rada mikrokontrolera – stabilizatori napona, kvarcni oscilator za generiranje frekvencije takta i slično. Postoji nekoliko vrsta/veličina Arduino pločica, a najpopularnije su: Uno Mega i Mini.

Svaka Arduino pločica ima analogne i digitalne pinove, od kojih se svaki može postaviti u jedan od dva načina rada: input i output (ulaz i izlaz). Digitalni pinovi mogu čitati i postavljati isključivo logička stanja jedan i nula, dok kao što i ime sugerira, kod analognih je moguće čitati i postavljati i vrijednosti koje nisu isključivo nula ili jedan. Tako će digitalRead() vratiti HIGH (1) ili LOW(0), dok analogRead() vraća broj između 0 i 1023. Analogna čitanja su očigledno puno preciznija, ali uz cijenu vremena, jer su digitalna čitanja puno brža.

Arduino IDE razvojna okolina mogla bi biti i puna bolja, pogotovo usporedivši ju sa nekim naprednim drugim razvojnim okolinama za ostale jezike, poput IntelliJ-a. Činjenica da se radi o open-source platformi omogućava drugim proizvođačima izradu svoje verzije Arduino pločice, što rezultira puno jeftinijim varijantama koje su dostupne za svega par američkih dolara.[1] ESP32 je mikrokontroler koji za razliku od Arduino pločice, ima integriran WiFi i Bluetooth modul što ga čini vrlo popularnim na tržištu.

Radi se o vrlo pouzdanom sustavu, koji bez poteškoća radi na temperaturama između -40°C i +125°C. Ima vrlo malo potrošnju energije te ga čini i vrlo praktičnim za sustave koji se napajaju preko baterije. Moguće je konfigurirati Arduino razvojno sučelje kako bi podržavao programiranje za ESP32. Ima micro usb sučelje za povezivanje sa računalom i/ili napajanje. Neke izvedbe dolaze i sa ugrađenim senzorima, poput senzora dodira i temperature.[2]

3. WEB APLIKACIJE UZ SPRING BOOT I REACT TEHNOLOGIJE

3. WEB APPLICATIONS WITH SPRING BOOT AND REACT TECHNOLOGIES

Glavna zadaća Spring Boot programskog okvira je lagana konfiguracija Spring web aplikacije, te korištenje gotovih rješenja.

Inicijalizira kontekst Springa, te koristi ugrađeni (eng. embedded) poslužitelj – podrazumijevani je Apache Tomcat. Moguće ga je koristiti u kombinaciji sa Java, Kotlin ili Groovy programskim jezikom.

Uz pomoć Spring boot Initializr-a vrlo je lako započeti novi projekt – bilo kroz plugin za razvojno sučelje ili preko preglednika – potrebno je samo unijeti imena aplikacije i paketa, te označiti željene dependency-e [1]. Za build alat, koji upravlja svim depenceny-ma najčešće se koriste Gradle (baziran na Groovy programskom jeziku) i Maven (baziran na XML-u).[3]

Repozitorij je sloj aplikacije koji direktno komunicira s bazom podataka. Označava se anotacijom **@Repository** iznad klase. Dodavanjem dependency-a "spring-boot-starter-data-jpa" u projekt, te nasljeđivanjem sučelja JpaRepository, a upravljanje podacima iz baze vrlo je jednostavno. JpaRepository već sadrži osnovne metode poput save, delete, findAll, a vrlo lako može i iz imena metode, te povratnog tipa i tipa parametara, zaključiti što se od te metode očekuje.

Servisni sloj odrađuje većinu posla, odnosno u servisnim metodama nalazi se većina programske logike. Označava se anotacijom @Service iznad klase. Ideja je da se s endpointa u controlleru pozove servisna metoda koja će vratiti ono što se očekuje kao rezultat pozivanja tog endpointa. U servisnoj klasi se najčešće „autowire-aju“ beanovi repozitorija, ostalih servisa, mappera i slično. Također, praksa je kreirati sučelje sa metodama, te ih onda implementirati u klasi. Na taj način možemo imati više različitih implementacijskih klasa za neko sučelje (interface).

Controller je klasa koja ima definirane endpointe, odnosno „putanje“ koje se poziva sa frontenda ili nekog drugog sutava. Označava se anotacijom @ Controller ili @RestController, te uz to najčešće ide i @RequestMapping, koji postavlja prvi dio putanje do nekog endpointa unutar controllera. Moguće je definirati više tipova endpointa, najkorišteniji su svakako get, post, delete i put. Kao što je spomenuto prije, najčešće je ideja unutar metode u controlleru imati što manje koda, odnosno pozvati servisnu metodu koja će obaviti većinu ili cijeli posao.

Liquibase je biblioteka (library) pomoću kojeg je vrlo lako kreirati shemu baze podataka, i popuniti tu bazu nekim podacima. Nastao je još 2006. godine kako bi olakšao praćenje izmjena na bazi. Baziran je na XML-u i vrlo ga je lako koristiti. Glavni element je changeSet, koji označava skup naredbi. Ukoliko je changeSet izvršen, zapisi o tome zapisat će se u tablice databasechangelog i databasechangeloglock, te ukoliko se jedan od tih izvršenih changeSetova izmjeni, prilikom ponovnog pokretanja aplikacije izbacit će se exception te se aplikacija zbog toga neće pokrenuti ukoliko se prethodno ne izvrši „dropanje“ baze podataka. Kako bi se izbjegla ta situacija, potrebno je promjene dodati u novi changeSet. Upravo je to cijela poanta ovog library-a.

React je, kako i njegov slogan kaže, JavaScript library za kreiranje korisničkog sučelja. Specifičnost ovog JavaScript library-a su komponente (eng. components) koje omogućuju da se samo one “renderaju”, odnosno ponovno učitaju kada se promjeni njihov sadržaj, za razliku od ostalih rješenja kod kojih se kod bilo kakve promjene na stranici cijeli DOM (Document Object Model), tj. cijela stranica ponovno učitava. Takvim rješenjem dobivamo puno brži, te efikasniji sustav.[4]

4. POVEZIVANJE ARDUINA I ESP32 S WEB APLIKACIJOM

4. CONNECT ARDUINO AND ESP32 TO A WEB APPLICATION

Pošto Arduino pločica nema integriran WiFi modul, potrebno je koristiti dodatne komponente. Ovdje će primjer biti povezivanje s ESP32 mikrokontrolerom, koji ima ugrađen WiFi modul. Koristeći WiFi.h library, povezivanje ESP-a na WiFi mrežu vrlo je jednostavno [5]:

```
#include <WiFi.h>

const char* ssid = "SOME_SSID";
const char* password =  "secret";

void setup() {

       WiFi.begin(ssid, password);
```

```
       while (WiFi.status() != WL_
CONNECTED) {
          delay(1000);
          Serial.println("Connecting
to WiFi..");
       }

       Serial.println("Connected to
the WiFi network");
       Serial.println(WiFi.
localIP()); /*dobiveni IP*/
}
```

Kada je ESP32 povezan na WiFi mrežu, potrebno je još ostvariti komunikaciju između Arduina i ESP-a. To se može postići tako da se RX ESP-a poveže s TX pinom na Arduinu, a TX ESP-a na RX Arduina. HTTP zahtjevi s ESP32

Da bi bilo moguće slati POST HTTP zahtjeve, potrebno je uključiti library HTTPClient.h. Ukoliko je potrebno slati podatke u JSON formatu, dobro je koristiti jedan od library-a koji to olakšavaju, u ovom primjeru ArduinoJson.h.

```
#include <HTTPClient.h>
#include <ArduinoJson.h>
.
.
.
if ((WiFi.status()==WL_CONNECTED))
{//Provjeri postoji li WiFi
konekcija
      HTTPClient http;
      StaticJsonBuffer<200>
jsonBuffer;
      JsonObject& root = jsonBuffer.
createObject();
      root["temperature"] = temp;
      root.printTo(buf,
sizeof(buf)); // kreiraj JSON

      http.
begin("http://192.168.5.15:8080/
temperature");
      http.addHeader("Content-Type",
"application/json");
      http.POST(buf);

      http.end(); //Oslobodi resurse
}
```

Vrlo slično kao i POST zahtjev može se ostvariti i GET zahtjev, kao što je vidljivo i u primjeru koji slijedi:

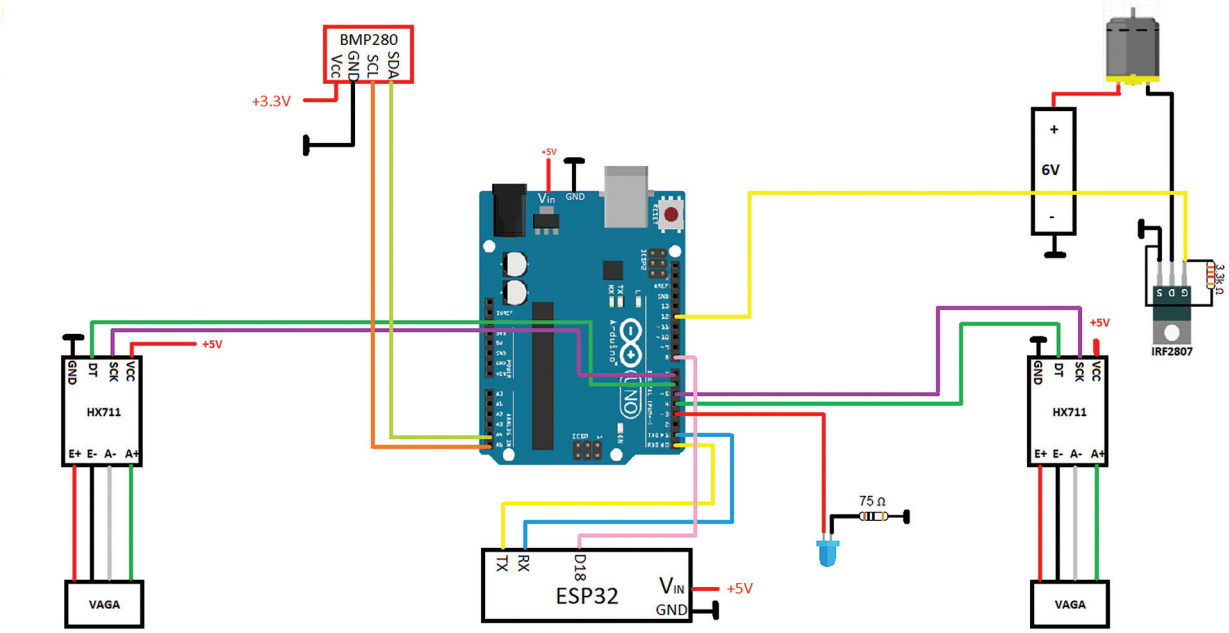
```
if ((WiFi.status()==WL_CONNECTED))
{//Provjeri postoji li WiFi
konekcija
    HTTPClient http;
    http.
begin("http://192.168.5.19:8080/
food/refill-status");

    /*U ovom slučaju zanima nas samo
status code odgovora, ali
    moguće je i dohvaćati podatke*/
    int res = http.GET();

    //ako je status code odgovora
200, obavi neku radnju
    if(res == 200){
       digitalWrite(18, HIGH);
    }
 }
```

Na istom principu moguće je raditi i DELETE, PUT i ostale zahtjeve.

5. IMPLEMENTACIJA
5. IMPLEMENTATION

Sklopovski dio platforme sastoji se od dvije vage (voda, hrana), dva HX711 analogno digitalnih pretvornika, Arduino Uno i ESP32 mikrokontrolera, senzora za temperaturu BMP 280, istosmjernog motora, MOSFET NPN tranzistora IRF2807, baterije (6V) za napajanje motora, tri led diode i četiri otpornika. Kako je sve to ukomponirano moguće je vidjeti na logičkoj shemi slika 1.

Svaka vaga spojena je na svoj HX711 analogno digitalni pretvornik. Vaga izgleda poput kvadra. Radi se o metalnom kućištu u kojem se nalaze promjenjivi otpornici koji mijenjaju otpor ovisno o naprezanju (sili) koja djeluje na kućište. Četiri otpornika povezana su u Wheatstoneov most, tako se jednom paru otpornika smanjuje otpor, a drugom povećava, a to omogućava da se iz te promjene otpora izračuna iznos sile.



Slika 1 *Logička shema cijelog sklopovlja*

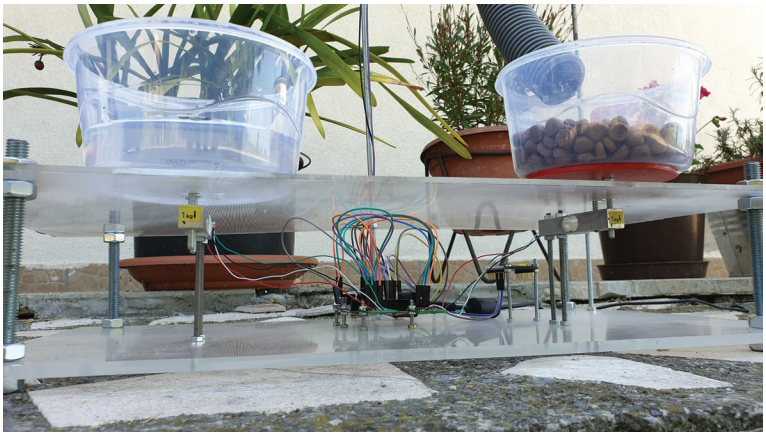
Figure 1 *Logical scheme of the entire hardware*

Pošto su promjene otpora jako male, onda su i promjene napona male i zbog toga se koristi analogno digitalni pretvornik HX711 s pojačalom koji pojačava signal 128 puta. Pretvornik pretvara mjerenje otpora u digitalne podatke i tako se šalju na Arduino mikrokontroler koji dalje obrađuje te podatke kako bi izračunao masu. [6]

Arduino Uno pločica u sebi ne sadrži WiFi modul, stoga se u ovom rješenju za to koristi ESP32 mikrokontroler koji ga sadrži. Nakon što Arduino izračuna i obradi potrebne podatke, ukoliko je potrebno, rezultate će poslati na poslužitelj. Tako će temperaturu slati svakih pola sata, dok će masu hrane i vode poslati kada se ona promjeni za više od 2 grama. Pošto ih sam Arduino ne može direktno slati preko WiFi-a, on će podatke preko serijske komunikacije poslati na ESP32, odakle će se on pobrinuti da podaci završe na poslužitelju. Kako ostvariti serijsku komunikaciju između ova dva mikrokontrolera objašnjeno je u prethodnoj cjelini.

Pošto je potrebna i komunikacija u drugom smjeru, odnosno s web aplikacije moguće je poslati komandu sklopovlju, postavlja se pitanje kako to ostvariti, pošto se ESP32 spaja na WiFi prilikom pokretanja, njegova IP adresa neće uvijek biti ista, te je bez tog IP-a praktički nemoguće „dosegnuti“ ESP32. Stoga, ovdje je primijenjeno rješenje da se svaku sekundu sa ESP-a šalje GET zahtjev na poslužitelj, koji mu vrati HTTP status 200 ukoliko je potrebno dodati još hrane u posudu, te HTTP status 204 ukoliko to nije potrebno. Kada ESP pročita status 200, na poslužitelju (backendu) će se odmah promijeniti sesijska Boolean varijabla koja kaže da je potrebno nadopuniti hranu, na „false“. Tu sesijsku varijablu korisnik može postaviti na „true“ pritiskom na gumb „refill food“ u web aplikaciji na ekranu gdje se nalazi statistika o hrani. Nakon što je ESP dobio status 200, on postavi digitalni pin 18 u stanje logičke jedinice. Zatim Arduino, kojem je povezen digitalni pin 8 sa ESP-ovim pinom 18, očitava stanje tog ulaza. Ukoliko pročita logičku jedinicu, on će pokrenuti sipanje hrane, te će serijskom komunikacijom poslati ESP-u poruku da je počeo sa sipanjem hrane u posudu i da ugasi svoj pin 18.

Sipanje hrane radi tako da se generira pravokutni impuls na istosmjerni motor, koji se tako vrti u intervalima, te se time postiže puno elegantnije i tiše sipanje hrane. Radi se o 40 intervala gdje se prvo 50 milisekundi motor vrti, odnosno dobiva napajanje, a idućih 100 milisekundi ne. Da se ne radi o intervalima, hrana ne bi ispadala umjerenim tempom te bi motor proizvodio puno veću buku.



Slika 2 *Platforma za vaganje vode i hrane, te dodavanje hrane*

Figure 2 *Platform for weighing water and food, and adding food*

Kako bi bilo moguće generirati impuls s Arduina prema motoru, koristi se NPN MOSFET IRF2807 tranzistor koji se ovdje ponaša kao sklopka. Glavna karakteristika MOSFET tranzistora je vrlo veliki ulazni otpor (~10^{15} Ω). Kada se na vrata (G) dovede napon do maksimalno 20V (vrijedi za korišteni tranzistor IRF2807, u ovom rješenju dovodi se napon od 5V s Arduina), struja će poteći između izvora (S) i odvoda (D), što logički možemo zamisliti kao da se sklopka spustila. [7] Struja tada prolazi kroz motor i on vrti svoju osovinu. Kao što je vidljivo i na shemi cijelog sklopovlja, između izvora i odvoda postavljen je otpornik od 3.3k Ω jer inače bi čak i vrlo mali napon, poput dodirnog (statički elektricitet) bio dovoljan da se „sklopka“ spusti i motor se krene vrtjeti.

Motor se nalazi vertikalno iznad otvora lijevka, te mu je na osovinu spojen feder, koji zbog svojih zavoja, prilikom vrtnje gura hranu prema dolje. Uz feder, prethodno je testirana spirala bakrena žica, međutim to rješenje se nije pokazalo dovoljno dobrim jer bi hrana vrlo lako zablokirala protok hrane kroz „grlo“, stoga je elastični feder idealno rješenje. Testiranjem je primijećeno da prilikom vrtnje, feder zbog svoje elastičnosti može „iskočiti“ gore u lijevak, stoga je na dno federa pričvršćen savijeni komad žice koji zbog svoje dimenzije onemogućava federu da iskoči iz „grla“ lijevka. Zbog svojeg spiralnog oblika, feder može, ovisno o smjeru vrtnje, gurati hranu dodatno prema gore ili dolje, a ovdje je naravno smjer vrtnje takav da dodatno olakšava sipanje hrane tako da ju gura prema dolje.

Za dodatan vizualan efekt, tri led diode smještene su unutar pleksiglas postolja, te se pale i gase u isto vrijeme kao i motor, čime se naglašava sam princip rada. Za zaštitu od prevelikog napona, na svaku diodu spojen je i otpornik od 75 Ω.

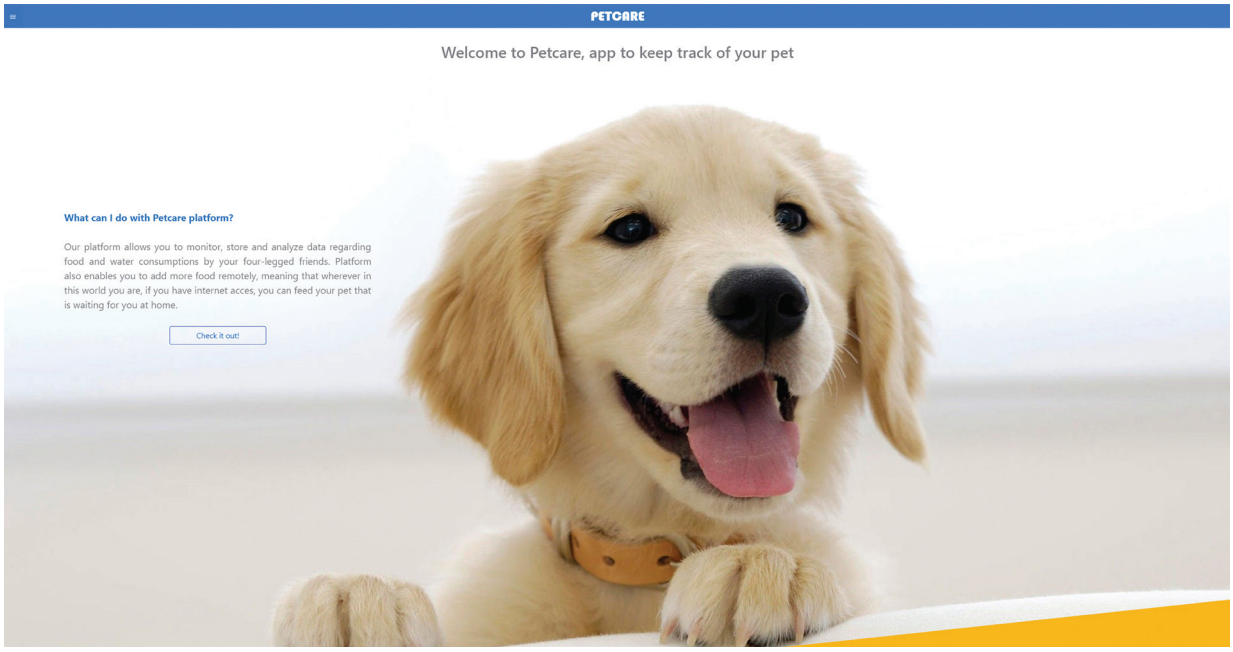
Za olakšani rad izrađena je i web aplikacija kroz koju korisnik vrlo lako može provjeriti kada se njegov kućni ljubimac hranio, koliko je hrane pojeo, da li ima još hrane u posudi te dodati još hrane, provjeriti iste podatke za vodu, također i provjeriti trenutnu, minimalnu i maksimalnu temperaturu tijekom dana. Moguće je dobiti graf i statistiku za određeni datum koji nas zanima korisnika.

Aplikaciji je moguće pristupiti na URL-u: https://petcare-app-ml.firebaseapp.com/

Backend aplikacija „hosta“ se na Heroku platformi, dok je frontend dio aplikacije „hostan“ preko Google-ove Firebase platforme.

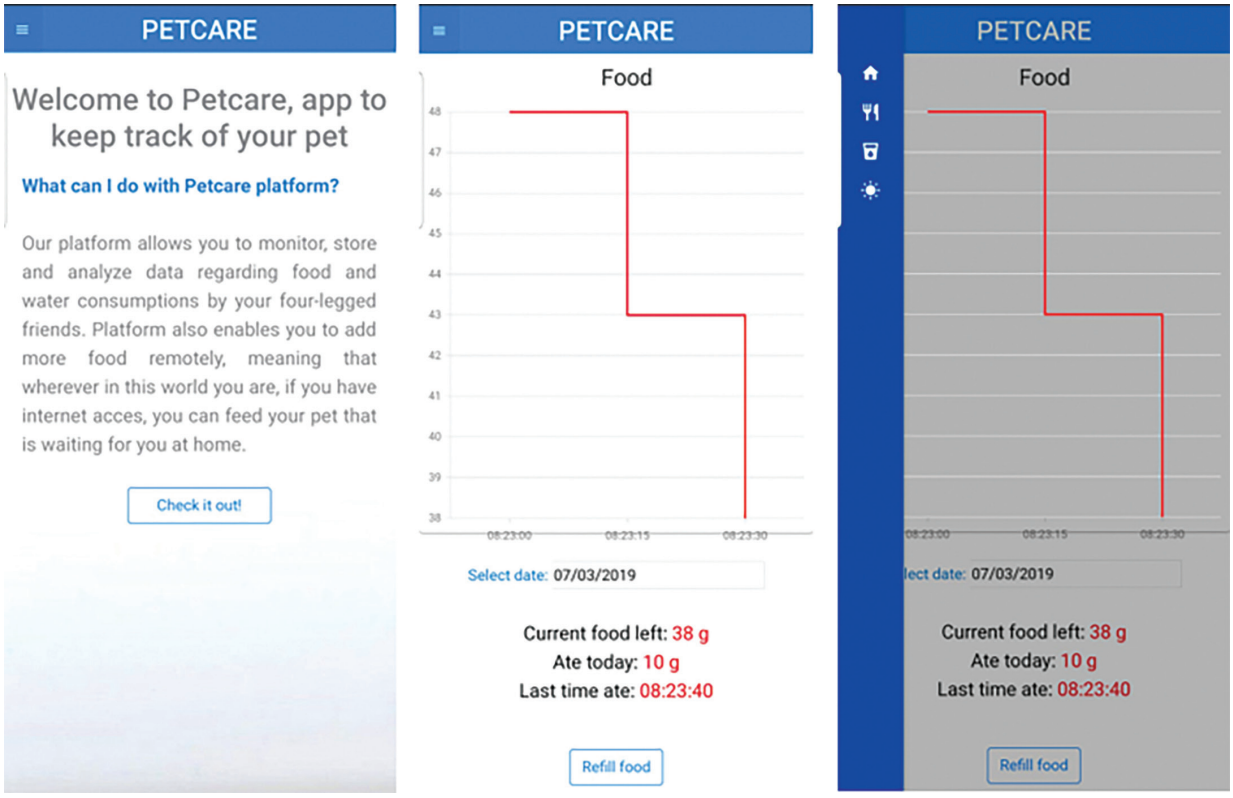
Cijela stranica je u minimalističkom i jednostavnom dizajnu, tako se na početnoj stranici nalazi kratki opis platforme, te tri tipke koje vode svaka na svoj ekran: hrana, piće i temperatura.

Kako bi korisnici mogli bez poteškoća koristiti aplikaciju i preko svojeg pametnog telefona, kreiran je i mobilni prikaz. Na slici 4 koja slijedi treba primijetiti kako je klizni meni s lijeve strane uži i sadrži samo ikone, statistički podaci ispod grafa nisu u istom redu, već svaki zasebno, te se tekst na početnoj stranici proteže preko cijelog ekrana.



Slika 3 *Izgled početne stranice*

Figure 3 *Homepage layout*



Slika 4 *Mobilni prikaz aplikacije*

Figure 4 *Mobile application display*

6. ZAKLJUČAK
6. CONCLUSION

Kombinacija mikrokontrolera i web aplikacije je pokazala se jako korisna, jer bez obzira gdje se nalazili i u bilo kojem trenutku možete provjeriti i upravljati nekim sustavom, definitivno čini korisnike zadovoljnim. Danas uz dostupne tehnologije, ovakve sustave zanimljivo nije teško razviti što ga čini idealnim i za „kućne" projekte, odnosno ljude koji vole razvijati ovakve sustave iz hobija. Važno je napomenuti kako su i komponente za ovakve sustave povoljni i lako dostupni. Modernim mikrokontroleri kao što su Arduino Uno te ESP32 u kombinaciji s web aplikacijom čine pametan sustav. Backend web aplikacije pisan je po najnovijim standardima Spring Boot programskog okvira za Javu, isto kao i frontend koji koristi trenutno najpopularniji JavaScript programski okvir na svijetu, a to je React. UI/UX dizajn web aplikacije vrlo je važan jer podiže razinu korisničkog iskustva, što dovodi do zadovoljstva korištenja aplikacije svojom jednostavnošću i funkcionalnosti. Web aplikacija također prilagođena je i za preglednike pametnih telefona, kako bi cijeli sustav bio dostupniji za korištenje. Ova aplikacija ima veliki potencijal jer bi uveliko pomogla svim zaposlenima a vlasnici su kućnih ljubimaca. Ovakvim rješenjem moguće je točno dozirati hranu za optimalni unos hrane životinja. Moguće je implementirati jako puno dodatnih statističkih podataka kao i druge funkcionalnosti koje bi korisnicima ovakve platforme bilo od velike koristi.

7. REFERENCE
7. REFERENCES

[1.] M.Margolis, „Arduino Cookbook", O`Reilly Media, 2nd edition, ISBN-10: 1449313876, 2011

[2.] S.Spanilescu, „ESP32 programming for the Internet of Things", ISBN:9780359282562, 2018

[3.] C.Walls, „Spring Boot in Action", Shelter Island, New York, ISBN-10 : 1617292540, 2016

[4.] A.Banks, E.Porcello, „Learning React", O`Reilly Media, 2nd edition, ISBN-10 : 1492051721, 2019

[5.] Techtutorialsx, 2019, URL:https://techtutorialsx.com/2017/04/24/esp32-connecting-to-a-wifi-network/

[6.] E-radionica.com, KKM: HX711 + load-cell, 2019, https://e-radionica.com/hr/blog/2019/04/08/kkm-hx711-load-cell/

[7.] International IOR Rectifier, 2019, https://www.infineon.com/dgdl/irf2807pbf.pdf?fileId=5546d462533600a4015355dea79e18f2

AUTORI · *AUTHORS*

● **Mario Lukec**
Rođen 13.01.1998. u Zagrebu. Završio je preddiplomski studij računarstva na Tehničkom veleučilištu u Zagrebu, a trenutno studira na specijalističkom studiju programskog inženjerstva na istom veleučilištu. Trenutno radi kao programski inženjer u tvrtci Reev (hrvatska podružnica zove se Digital Corporation for eMobility d.o.o.) gdje razvija platformu za električne punionice automobila.

Korespondencija · *Correspondence*
mario.lukec@tvz.hr

● **Ivica Dodig -** nepromjenjena biografija nalazi se u časopisu Polytechnic & Design Vol. 8, No. 2, 2020.

Korespondencija · *Correspondence*
ivica.dodig@tvz.hr

● **Jelena Kapelac Gulić -** nepromjenjena biografija nalazi se u časopisu Polytechnic & Design Vol. 8, No. 2, 2020.

Korespondencija · *Correspondence*
jelena.kapelac@tvz.hr

● **Brigitta Cafuta -** nepromjenjena biografija nalazi se u časopisu Polytechnic & Design Vol. 8, No. 2, 2020.

Korespondencija · *Correspondence*
brigitta.cafuta@tvz.hr

● **Ognjen Mitrović -** nepromjenjena biografija nalazi se u časopisu Polytechnic & Design Vol. 5, No. 2, 2017.

Korespondencija · *Correspondence*
ognjen.mitrovic@tvz.hr