Upravljanje servo motorima korištenjem Android aplikacije

Benković, Filip

Undergraduate thesis / Završni rad

2019

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University North / Sveučilište Sjever**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:122:594061

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-23**

Repository / Repozitorij:

University North Digital Repository

Sveučilište
Sjever

Završni rad br. 451/EL/2019

Upravljanje servo motorima korištenjem Android aplikacije

Filip Benković, 0873/336

Varaždinu, rujan 2019. godine

Sveučilište
Sjever

Odjel za Elektrotehniku

Završni rad br. 451/EL/2019

Upravljanje servo motorima korištenjem Android aplikacije

Student

Filip Benković, 0873/336

Mentor

Miroslav Horvatić, dipl.ing.

Varaždin, rujan 2019. godine

Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3, HR–42000 Varaždin

UNIVERSITY NORTH

Prijava završnog rada

Definiranje teme završnog rada i povjerenstva

ODJEL Odjel za elektrotehniku

STUDIJ preddiplomski stručni studij Elektrotehnika

PRISTUPNIK Filip Benković | MATIČNI BROJ 0873/336

DATUM 12.09.2019. | KOLEGIJ Automatsko upravljanje

NASLOV RADA Upravljanje servo motorima korištenjem Android aplikacije

NASLOV RADA NA ENGL. JEZIKU Servo motor control using Android application

MENTOR Miroslav Horvatić, dipl.ing. | ZVANJE predavač

ČLANOVI POVJERENSTVA

1. mr.sc. Ivan Šumiga, dipl.ing., viši predavač
2. mr.sc. Matija Mikac, dipl.ing., viši predavač
3. Miroslav Horvatić, dipl.ing., predavač
4. dr. sc. Dunja Srpak, dipl.ing., predavač- rezervni član
5.

Zadatak završnog rada

BROJ 451/EL/2019

OPIS

Realizirati sustav za upravljanje servo motorima korištenjem Android aplikacije, mrežne komunikacije i Arduino razvojnog sustava.

U radu je potrebno:
- osmisliti i realizirati Android aplikaciju i web stranicu za upis podataka na poslužitelj,
- osmisliti i realizirati mrežnu komunikaciju poslužitelja i Arduino razvojnog sustava,
- osmisliti i realizirati upravljanje servo motorima korištenjem Arduino razvojnog sustava.

ZADATAK URUČEN 19.09.2019

REPUBLIKA HRVATSKA 5 KOPRIVNICA SVEUČILIŠTE SJEVER

POTPIS MENTORA

Predgovor

Zahvaljujem se svome mentoru Miroslavu Horvatiću na njegovom trudu i strpljenju te smjernicama koje mi je davao oko završnog rada. Zahvaljujem se svojim roditeljima koji su mi omogućili studiranje te me podržavali tokom studiranja na Sveučilištu Sjever, te svim svojim bliskim ljudima koji su me podržavali.

Zahvaljujem se Sveučilištu Sjever, te svim profesorima na prenesenom znanju koje ću moći iskoristiti u daljnjem životu.

Sažetak

U ovom završnom radu realizirano je upravljanje servo motorima pomoću Android i Arduino razvojne platforme i Ethernet modula. Napravljena je web stranica i Android aplikacija pomoću kojih se vrijednosti za servo motore upisuju na poslužitelj u tekstualne datoteke. Potom se te vrijednosti uz pomoć Arduino razvojne platforme i Ethernet modula dohvaćaju i prenose na servo motore. Realizirano je i slanje podataka s Arduino razvojne platforme na poslužitelj. Nakon pritiska na tipku, prekidna rutina pokreće zapis stanja svjetleće diode u tekstualnu datoteku. Stanje svjetleće diode se kasnije iz tekstualne datoteke dohvaća i prikazuje na web stranici i Android aplikaciji. Dodana je još jedna prekidna rutina, koja nakon pokretanja, sprema realno vrijeme u tekstualnu datoteku uz pomoć RTC modula.

Ključne riječi: Arduino Ethernet shield, RTC shield, Mobilna aplikacija, Arduino, servo motor

Abstract

This paper contains example of contolling servo motor with Android software and Arduino development platform with Ethernet shield. A website and Android application has been created that sends values for servo motors that go to the server into text files. These values are then transmitted to servo motors using the Arduino development platform and Ethernet shield. Also sending data from the Arduino development platform to the server was implemented. After pressing a key, the interrupt routine starts to record the status of the LED into a text file. The status of the LED is later retrieved from the text file and displayed on the website and Android application. Another interrupt routine has been added, which, after startup, saves real time in a text file using the RTC shield.

Keywords: Arduino Ethernet shield, RTC shield, Mobile app, Arduino, servo motor

Popis korištenih kratica

HTML	HyperText Markup Language
RTC	Real Time Clock
USB	Universal Serial Bus
XML	EXtensible Markup Language
PHP	Hypertext Preprocessor
AJAX	Asynchronous JavaScript and XML
CSS	Cascading Style Sheets
MAC	media access control address
DHCP	Dynamic Host Configuration Protocol
HTTP	Hyper Text Transfer Protocol
IDE	Integrated Development Environment
PWM	Pulse width modulation

Sadržaj

1. Uvod 5
2. Arduino 6
2.1. Arduino razvojno okruženje 8
2.2. Ethernet modul 9
2.3. Real Time Clock modul 10
3. Android 12
3.1. Android Studio 12
4. Web programiranje 13
4.1. HyperText Markup Language 13
4.2. Cascadina Style Sheets 13
4.3. Javascript 14
4.4. Hypertext Preprocessor 14
5. Usmjerivač 15
5.1. Xampp 15
6. Praktični dio 16
6.1. Izrada web stranice 16
6.2. Spremanje vrijednosti 18
6.3. Izrada programa za Arduino 19
6.4. Izrada mobilne aplikacije 27
6.5. Sastavljanje makete 52
6.6. Spajanje na Arduino 53
7. Prikaz vrijednosti 56
8. Zaključak 57
9. Literatura 58

1. Uvod

Uz pomoć Arduino razvojne platforme moguće je automatizirati razne elektroničke uređaje i sklopove. Zbog dostupnosti dokumentacije i jednostavnosti korištenja Arduino razvojna platforma danas se koristi za upravljanje raznim sustavima. Za upravljanje amaterskim izvedbama 3D printerima često se koristi Arduino Mega platforma, dok je Arduino Uno pogodniji za manju kućnu automatizaciju. Pomoću modula (engl. shield) napravljenih za Arduino razvojnu platformu mogućnosti automatizacije korištenjem Arduino platforme se povećavaju.

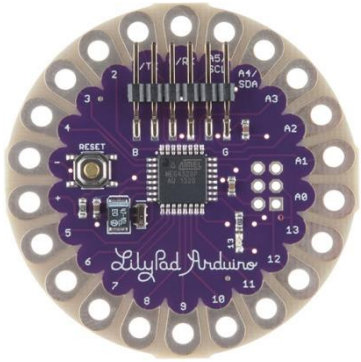
Ethernet modulom moguće je Arduino razvojnu platformu povezati s Internetom. To može biti korisno zato što tada Arduino može komunicirati s nekim poslužiteljem. Moguće je realizirati dohvaćanje nekih vrijednosti ili podataka sa web poslužitelja. Isto tako moguće je slati podatke na web poslužitelj. Navedeni podaci mogu se periodično upisivati u bazu podataka ili u neku tekstualnu datoteku. Podaci upisani na poslužitelj mogu se dohvatit na nekoj web stranici ili Mobilnoj aplikaciji. To može biti vrlo korisno kod kuće automatizacije za uključivanje ili isključivanje svjetla i slično. Također, moguće je realizirati upravljanje servo motorima uz pomoć mobilne aplikacije. U slučaju da je dostupan poslužitelj, može se realizirati upisivanje vrijednosti sa web stranice i mobilne aplikacije u npr. tekstualne datoteke koje se potom mogu dohvaćati sa Arduino razvojnom platformom i te vrijednosti prenositi na servo motore.

Problem ovakve komunikacije je vrijeme kašnjenja kod dohvaćanja ili slanja podataka između web poslužitelja i Ethernet modula. To vrijeme kašnjenja može se smanjiti korištenjem prekidnih rutina. Arduino Uno razvojna platforma može podržavati najviše dvije prekidne rutine.[54]

Za Arduino razvojnu platformu danas postoji mnogo različitih modula s raznim funkcionalnostima. Tako postoji Real Time Clock modul (RTC modul) pomoću njega se može kontinuirano računati realno vrijeme. Korištenjem RTC modula i mogućnosti Ethernet modula, trenutno vrijeme se može upisivati na poslužitelj. Upisivanje realnog vremena na poslužitelj može biti korisno ako se želi zabilježiti neka promjena na Arduino razvojnoj platformi te spremiti vrijeme kada se ta promjena dogodila.

2. Arduino

Arduino je otvorena razvojna platforma koja se koristi kod izgradnje elektroničkih projekata. Arduino se sastoji od fizičke programibilne pločice i od programskog dijela kojim se programira fizički dio. Arduino je nastao u Italiji 2005.g. Od početka razvijanja Arduino platforme pa sve do sada, Arduino je postao vrlo popularan. Popularnost leži u njegovoj jednostavnosti te niskoj cijeni, ali također i vrlo velikim mogućnostima i dobroj podršci. Program se na Arduino može prebaciti pomoću usb sučelja, a za to nije potreban vanjski programator kao za neke druge mikrokontrolere. Cijena Arduino razvojne platforme postala je vrlo pristupačna od 2005.g. pa do danas. Pruža se poprilično velika podrška na Internetu, od raznih video tutoriala pa sve do raznih foruma gdje ljudi razmjenjuju svoja iskustva i probleme. Za Arduino postoji pojednostavljena verzija C++ programskog jezika koja se koristi za njegovo programiranje. Mnoštvo već postojećih knjižica za razne probleme i za razne elektroničke uređaje i sklopove, utječu na jednostavnosti Arduino platforme. Njegova jednostavnost i niska cijena čine ga izvrsnim za početnike u elektronici. Također je pogodan kod manje kućne automatizacije. Danas postoji mnoštvo različitih Arduino platformi npr. LilyPad, Mega 2560, Micro, Mini, Pro, Zero, Uno, Nano, Leonardo itd. Sve te varijante razlikuju se po veličini, procesorskoj brzini, procesoru, broju digitalnih ulaza/izlaza, broju analognih ulaza, memoriji itd. Najpopularniji su Uno, Mega 2560 i Nano. Nano je zapravo manji i kompaktniji od Arduino Uno. Arduino Mega 2560 ina puno više ulaza/izlaza te veću memoriju.

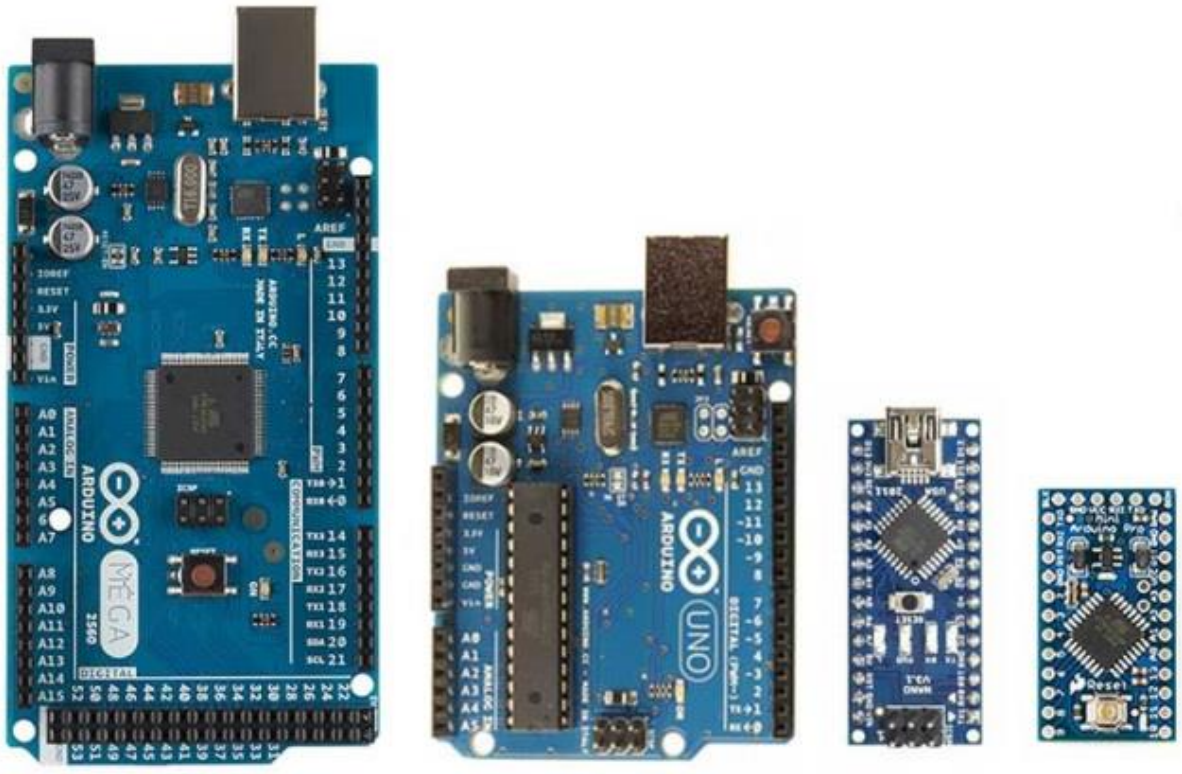


Slika 2.1 Izgled Arduino LilyPad pločice

Arduino Uno sastoji se od ATmega328p procesora, koji može raditi maksimalnom frekvencijom 16MHz, a koji sadrži 14 digitalnih ulaza/izlaza, od kojih 6 ima mogućnost PWM-a. Arduino Uno ima i 6 analognih ulaza, a sadrži 32kB memorije. Arduino Mega ima 54 digitalnih ulaza/izlaza, od kojih 15 ima mogućnost PWM-a, a sadrži 128KB memorije, Arduino Mega 2560 verzija sadrži 256KB memorije. Usporedba karakteristika i izgleda Arduino Uno, Arduino Nano i Arduino Mega mikrokontrolerskih sustava prikazana je na slikama 2.2 i 2.3.

Microcontroller Board Comparision Chart	Uno	Nano	Mega
Microcontroller	ATmega328P	ATmega328	ATmega1280
Clock Speed	16 MHz	16 MHz	16 MHz
I/O's	14	22	54
Analog Inputs	6	8	16
PWM's	6	6	15
Operating Voltage	5V	5V	5V
Recommended Supply Voltage	7-12V	7-12V	7-12V
Flash Memory	32KB	32KB	128KB
SRAM	2 KB	2 KB	8 KB
EEPROM	1 KB	1KB	4 KB

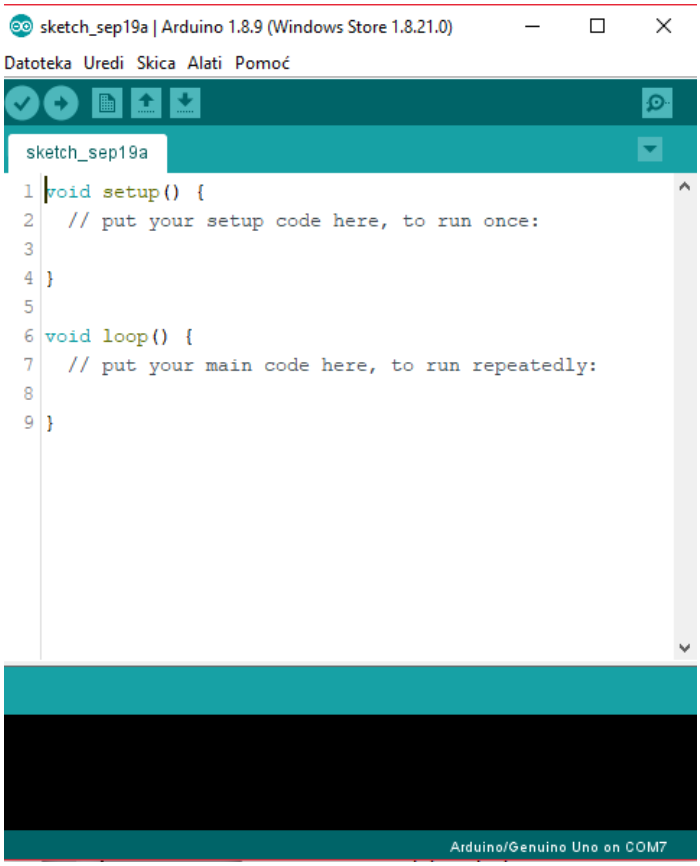
Slika 2.2 Usporedba Arduino Uno, Nano i Mega



Slika 2.3 Izgled Arduino Mega, Uno, Nano i Micro pločice

2.1. Arduino razvojno okruženje

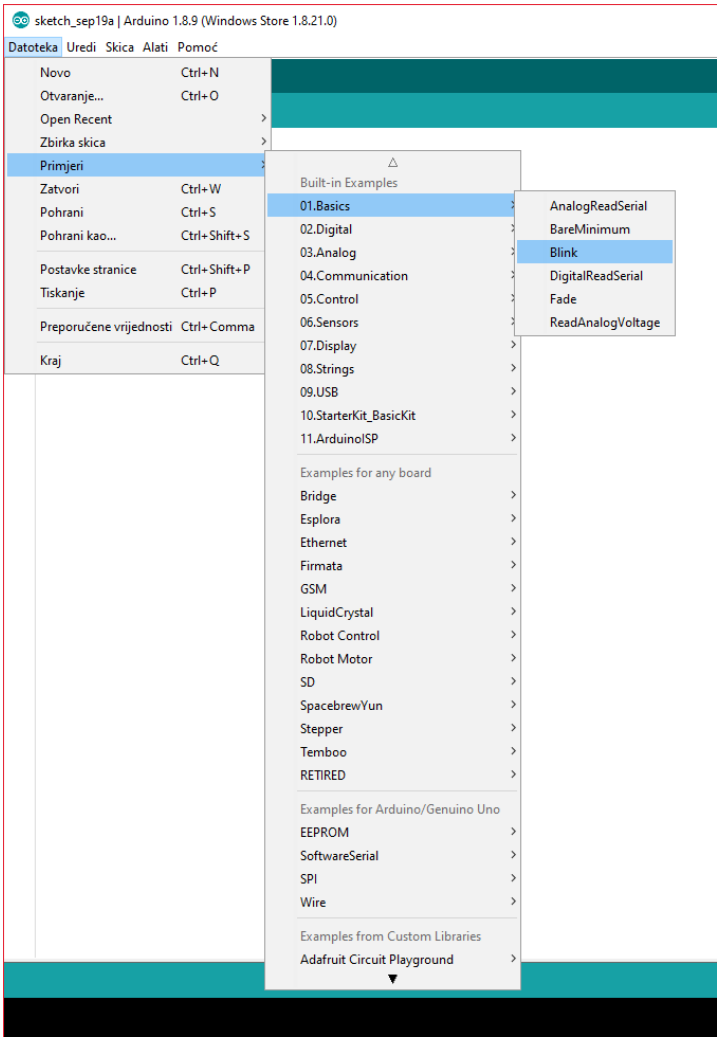
Arduino razvojno okruženje ili Arduino IDE je programsko okruženje pomoću kojeg se izrađuje program za Arduino pločicu. Pomoću tog okruženja programski kod se prebacuje na Arduino pločicu. Program se bazira na C++ programskom jeziku. Unutar samog okruženja dostupno je mnogo primjera pogodnih za učenje.



Slika 2.4 Izgled Arduino razvojnog okruženja

Programski kod se sastoji od dvije funkcije „void setup()“ i „void loop()“. Prva funkcija, odnosno „void setup()“ pokreće se samo kod spajanja Arduino pločice na napajanje ili kod prebacivanja programskog koda. Dakle, „void setup()“ se izvršava samo jednom, dok se „void loop()“ beskonačno ponavlja.

Unutar programskog okruženja primjeri su dostupni klikom na izbornik „Datoteka->Primjeri“, nakon čega se otvara mnoštvo primjera za učenje.



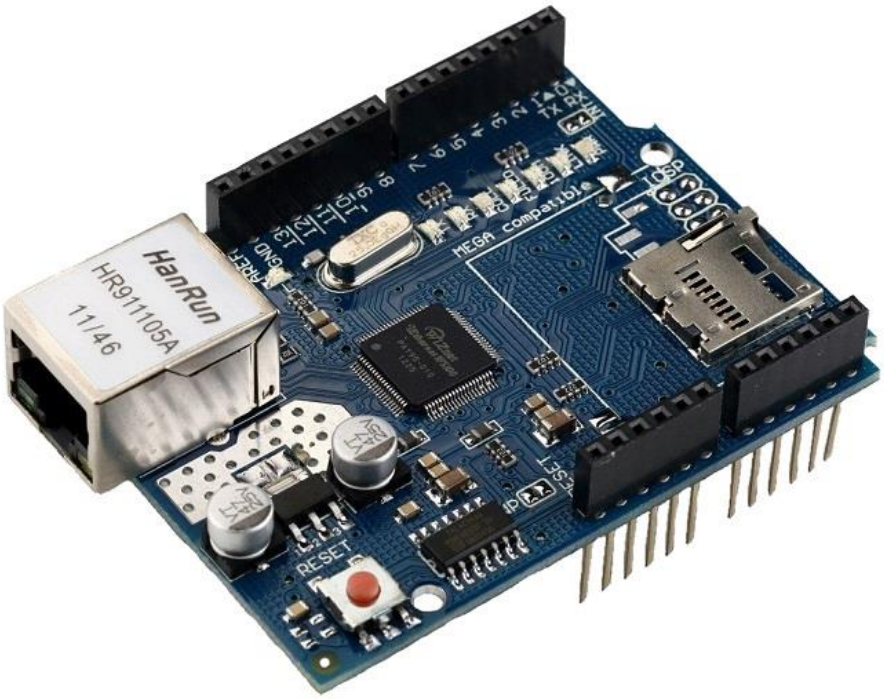
Slika 2.5 Prikaz programskih primjera u Arduino razvojnom okruženju

2.2. Ethernet modul

Za Arduino razvojnu platformu postoji mnogo raznih modula (engl. shield) koji se spajaju direktno na pločicu. Ti moduli omogućavaju neke dodatne funkcionalnosti koje sam mikrokontroler ne podržava. Danas su raspoloživi i mogu se koristiti Ethernet shield, Wi-fi shield, Dc motor control shield, Relay shield, LCD shield, smoke detector shield, bluetooth shield te mnogi drugi.

Slika 2.6 Prikaz relejnog štita za Arduino

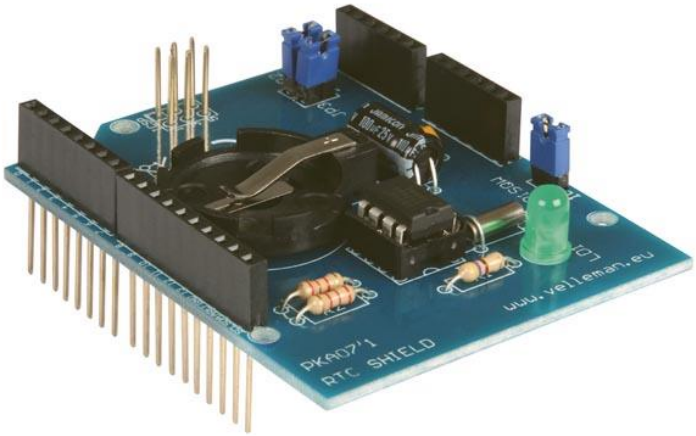
Ethernet modul omogućava spajanje Arduino razvoje platforme na računalnu mrežu. To može biti korisno kada je potrebna mogućnost slanja ili dohvaćanja podataka iz poslužitelja ili web stranice. Elektronički uređaj ili sklop može biti kontroliran web stranicom ili mobilnom aplikacijom pomoću Arduino razvojne platforme s Ethernet modulom. Sastoji se od Wiznet 5100 integriranog kruga koji podržava TCP i UDP. Ethernet modul se s Arduino pločicom spaja pomoću SPI komunikacije. Za spajanje koristi pinove 4, 10, 11, 12 i 13. Ethernet modul ima utor za SD karticu pomoću koje se mogu spremati ili čitati podaci. Ethernet modul se s računalom mrežom spaja pomoću RJ45 kabela.[56]



Slika 2.7 Izgled Arduino Ethernet modula

2.3. Real Time Clock modul

RTC modul za Arduino kontinuirano računa realno vrijeme, RTC modul realno vrijeme računa neovisno o napajanju Arduino razvojne platforme, zato što sadrži svoju zasebnu bateriju. To vrijeme RTC modul prosljeđuje na Arduino razvojnu platformu, koja onda može zabilježiti točno vrijeme neke promjene na Arduino razvojnoj platformi. Vrijeme se preko Ethernet modula može proslijediti na neki poslužitelj i snimiti u tekstualnu datoteku ili bazu podataka. RTC modul temelji se na Maxim-Dallas DS1307 integriranom krugu. Može brojiti sekunde,minute, sate, dane u mjesecu, dane u tjednu i godine. Ima mogućnost odabira između 12 ili 24 satnog sustava. Može se koristiti s drugim modulima.[27]



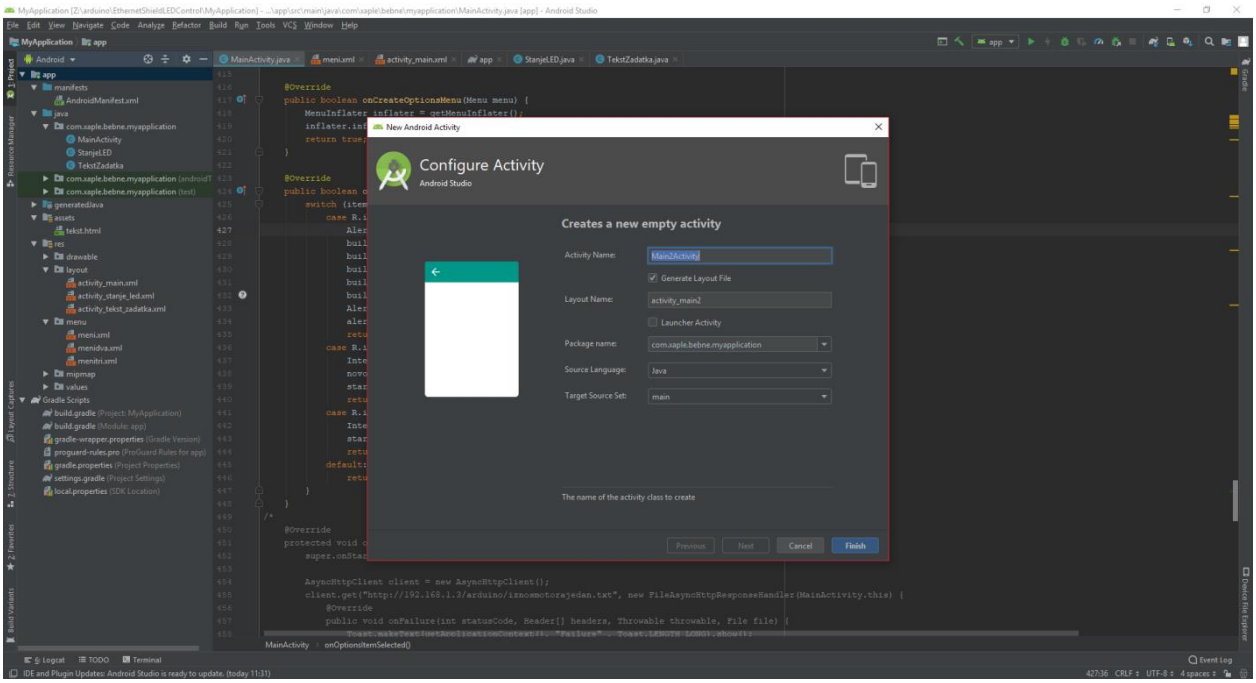
Slika 2.8 Izgled Arduino RTC shield-a [27]

3. Android

Android je otvoreni operacijski sustav napravljen za pametne mobilne uređaje, tablete, pametne televizore, sustave u automobilima i za pametne satove. Jezgra Android operacijskog sustava temelji se na Linux sustavu. Linux sustav je besplatni operacijski sustav sličan Unixu. Prvi uređaji s Android sustavom pojavili su se 2008.godine. Trenutna verzija Android sustava je 9.0, a naziv trenutne verzije je Android „Pie“.[7]

3.1. Android Studio

Android studio je službeno razvojno okruženje kod izgradnje aplikacija za Android sustave. Google je najavio Android studio 2013.g. Od 2019.g. preferirani programski jezik Android studioa je Kotlin. Programski jezik Java također je podržan od strane Android Studia. Android aplikacije su u početku programirane samo pomoću Jave programskog jezika, a kasnije se pojavio Kotlin programski jezik.



Slika 3.1 Izgled Android Studia-a razvojnog okruženja

4. Web programiranje

4.1. Hyper Text Markup Language

HTML je programski jezik za izgradnju web stranica. Njime se opisuje struktura web stranice npr. paragraf, naslov, tablica itd. HTML-om se definiraju svi elementi stranice. Elemente web stranice nazivaju se tagovi. HTML-om nije moguće izvršiti matematičku operaciju ili neki zadatak. Temeljna zadaća HTML-a je uputiti web preglednik kako prikazati neki „hipertext“ dokument. HTML datoteke su tekstualne datoteke s ekstenzijom .html ili .htm.[11]

Slika 4.1 Primjer HTML programskog kod[11]

4.2. Cascading style sheets

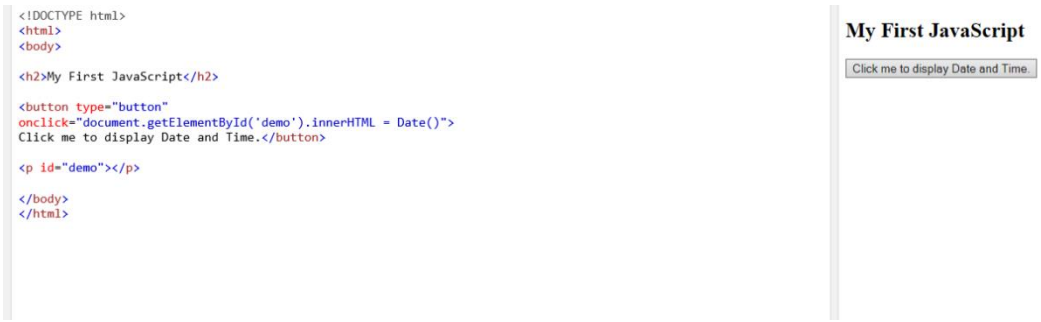
CSS je stilski jezik koji služi za oblikovanje stilova HTML elemenata. Pomoću CSS-a može se mijenjati boja teksta, font teksta kao i veličina teksta te mnoge druge mogućnosti. Moguće je promijeniti boju pozadine ili pomaknuti neki element web stranice po želji.[12]

Slika 4.2 Primjer programskog koda HTML-a s CSS-om[12]

4.3. Javascript

Javascript je programski jezik razvijen da bude sličan Javi programskom jeziku te da se izvršava u web pregledniku na klijentskoj strani. Javascript se koristi kako bi se programiralo ponašanje web stranice. JQuery je javascript knjižnica razvijena kako bi se olakšalo Javascript programiranje.

AJAX je set tehnika kod izgradnje web stranica na klijentskoj strani kako bi se postigla asinkrona web aplikacija. Pomoću AJAX-a web aplikacije mogu slati i primati podatke s poslužitelja bez ponovnog učitavanja web stranice. AJAX se izvršava u pozadini.[16]



Slika 4.3 Primjer HTML programskog koda s Javasript-om[16]

Nakon pritiska na gumb prikazuje se vrijeme i datum.

Slika 4.4 Prikaz HTML programskog koda s Javascript-om[16]

4.4. Hypertext Preprocessor

PHP je programski jezik namijenjen programiranju dinamičkih web stranica. PHP se izvršava na strani poslužitelja te omogućuje dinamičko upisivanje podataka u bazu podataka, odnosno bez ponovnog učitavanja web stranice.[26]

5. Usmjerivač

Usmjerivač (engl. router) je uređaj koji usmjeruje podatkovne pakete. Osnovni zadatak usmjerivača jest da pročita IP adresu u paketu te po svojoj tablici usmjeravanja usmjeri podatkovni paket. IP adresa je brojčana adresa računala na Internetu ili lokalnoj mreži. Sastoji se od 32 bita odnosno 4 okteta. Za dodjeljivanje IP adrese i mrežnih postavki zaslužan je DHCP (engl. Dynamic host configuration protocol). Navedeni protokol definira dodjeljivanje mrežnih postavki kao što su gateway i subnet maska. DHCP brine i o tome da u mreži nema sukoba adresa. Kako bi se web stranice mogle prezentirati zaslužan je HTTP (engl. Hyper Text Transfer Protocol) je protokol za komunikaciju između klijenta i poslužitelja. Usmjerivač služi za kreiranje računalnih mreža te za povezivanje i komunikaciju s Internetom.

MAC adresa je jedinstvena adresa svakog mrežnog uređaja. Sastoji se od 48 bita. Kada koristimo neke uređaje na lokalnoj mreži ili Internetu, IP adrese i MAC adrese su vezane.

Slika 5.1 Izgled usmjerivača

5.1. Xampp

Xampp je otvoreni i besplatni web poslužiteljski paket. Pomoću xampp-a može se testirati neki web poslužitelj prije puštanja poslužitelja u pogon. XAMPP ima mogućnost pokretanja apache HTTP poslužitelja na računalu. Može pokrenuti i MYSql bazu podataka. Xampp olakšava izgradnju web stranica jer se web stranica može testirati lokalno, ako zahtjeva komunikaciju s web poslužiteljem ili bazom podataka.

6. Praktični dio

Kako bi se Arduino razvojna platforma mogla programirati potrebno je instalirati Arduino IDE. Za izgradnju mobilnih aplikacija na Android-u potreban je Android Studio. Također, potrebno je instalirati Xampp koji će simulirati web poslužitelj i neki program za HTML, PHP datoteke, npr. Bluefish, ili Notepad++, tj. neki drugi.

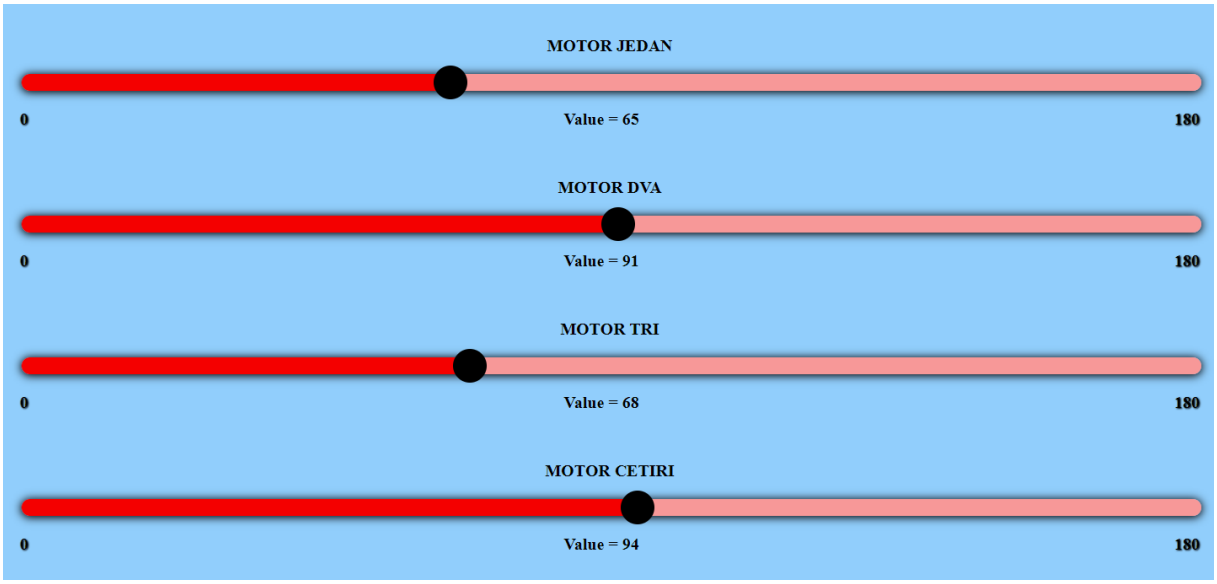
6.1. Izrada web stranice

```
<div id="boks">
<p id="para1">MOTOR JEDAN</p>
<div class="sliderjedan_div">
<input id="slide_jedan" type="range" min="1" max="179" step="1"
value="" class="slider_jedan_input">
</div>
<p id="paragraf"> 0 </p>
<p id="paragrafdva"> 180 </p>
<p id="para1"> Value <span id="sliderAmount_jedan"></span> </p>
<br />
<p id="para1">MOTOR DVA</p>
<div class="sliderjedan_div">
<input id="slide_dva" type="range" min="1" max="179" step="1"
value="" class="slider_jedan_input">
</div>
<p id="paragraf"> 0 </p>
<p id="paragrafdva"> 180 </p>
<p id="para1"> Value <span id="sliderAmount_dva"> </span></p>
<br />
<p id="para1">MOTOR TRI</p>
<div class="sliderjedan_div">
<input id="slide_tri" type="range" min="1" max="179" step="1"
value="" class="slider_jedan_input">
</div>
<p id="paragraf"> 0 </p>
<p id="paragrafdva"> 180 </p>
<p id="para1"> Value <span id="sliderAmount_tri"></span> </p>
<br />
<p id="para1">MOTOR CETIRI</p>
<div class="sliderjedan_div">
<input id="slide_cetiri" type="range" min="1" max="179" step="1"
class="slider_jedan_input">
</div>
<p id="paragraf"> 0 </p>
<p id="paragrafdva"> 180 </p>
<p id="para1"> Value <span id="sliderAmount_cetiri"></span></p>
</div>
```

Slika 6.1 HTML programski koda web stranice

Struktura web stranice izrađena je u HTML-u kao što je prikazano u programskom kodu na slici 6.1, te nakon toga stil dodan u CSS-u. Dakle, pomoću HTML-a dodana su četiri slider-a za četiri servo motora. Svaki slider može poprimiti vrijednost od 1 do 179.

Uz pomoć CSS-a može se promijeniti veličina slidera, boja slidera, a položaj teksta se može pomaknuti po želji. Veličina teksta te font, također zavise od CSS-a.



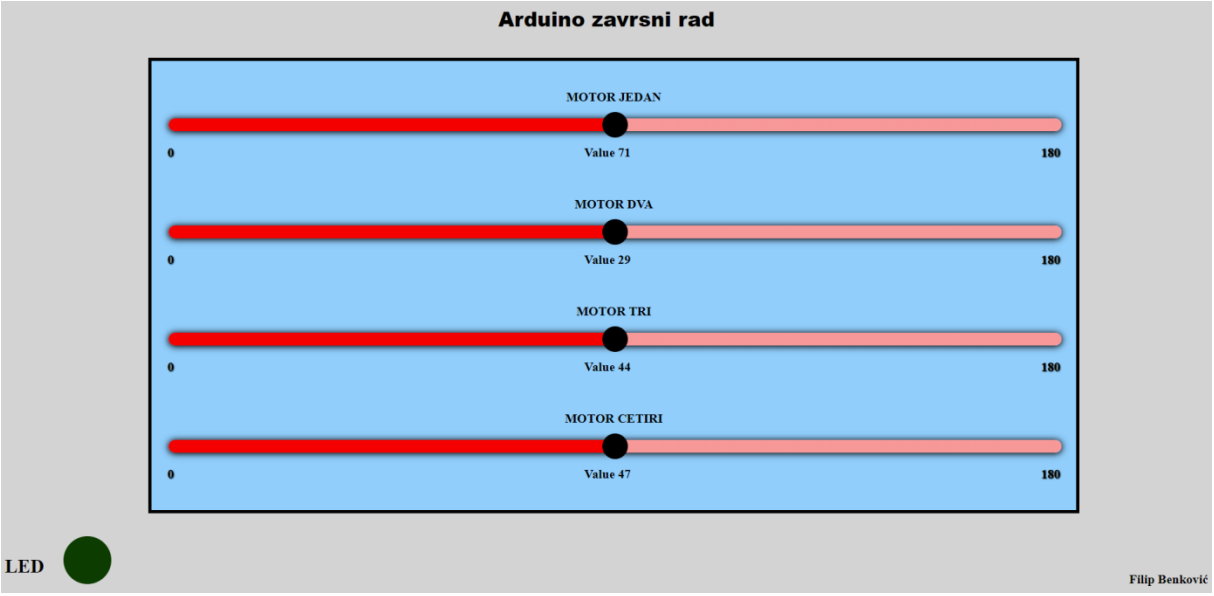
Slika 6.2 Početni izgled web stranice

Da bi se pomakom slider-a dinamički mijenjala boja, potrebno je u Javascript-u dodati programski kod kao na slici 6.3, koji mijenja boju slider-a zajedno s pomakom miša.

Nakon dodavanja još nekih elemenata HTML-a, a to su naslov, neki tekst i slično. Te dodavanja CSS-a za sve elemente HTML-a, dobiven je izgled konačne web stranice.

```
slide_jedan.addEventListener("mousemove", function () {
          var x = slide_jedan.value;
          var y = (x / 179) * 100;
var color = 'linear-gradient(90deg, rgb(255,0,0)' + y + '%,
rgb(255,153,153)' + y +    '%)';
          slide_jedan.style.background = color;
     })
```

Slika 6.3 Programski kod Javascript funkcije promjene boje slidera



Slika 6.4 Izgled gotove web stranice

6.2. Spremanje vrijednosti

Pomakom slidera poziva se PHP datoteka pomoću koje se trenutna vrijednost slider-a pohranjuje u tekstualnu datoteku.

```
slide_cetiri.onchange = function() {
    var iznos_cetiri = slide_cetiri.value;
    console.log("Vrijednost motora 4:" + iznos_cetiri);
    var xmlhttp = new XMLHttpRequest();
     xmlhttp.onreadystatechange = function () {
                if(xmlhttp.readyState == 4 && xmlhttp.status ==
200)
                {
                }
};
     xmlhttp.open("GET", "motorcetiri.php?vrijednost=" +
iznos_cetiri, true);
     xmlhttp.send();
```

Slika 6.5 Programski kod u kojem se poziva PHP datoteka

Javascript programski kod kao na slici 6.5 radi na način da čeka kad će se četvrtom slider-u promijeniti vrijednost. Kada se četvrtom slider-u promijeni vrijednost, pokrene se Javascript funkcija koja prvo pohrani vrijednost slidera u neku varijablu „iznos_cetiri“, a tu vrijednost ispisuje u konzoli. Korisno je u konzoli ispisati neke vrijednosti, tako da se kod ispitivanja web stranice u konzoli mogu pratiti promjene web stranice. Nakon toga, dodaje se „XMLHttpRequest“ te pomoću „xmlhttp.open“, poziva se PHP datoteka kojom se pohranjuju

vrijednosti u tekstualnu datoteku. Vrijednost se u PHP datoteku „motorcetiri.php“ prosljeđuje na način da se iza upitnika doda neki znakovni niz (engl. String) pomoću kojeg se u PHP datoteci traži vrijednost koja se zapisuje u tekstualnu datoteku. U ovom primjeru je „?vrijednost= + iznos_cetiri“, dakle u PHP datoteci se dohvaća sve nakon znaka jednakosti.

```
<?php
$varijabla = $_GET["vrijednost"]; // Dohvaćanje vrijednosti za
upisivanje
$textfile = "motorcetiriiznos.txt"; // Text document će biti
zapisana  vrijednost
 $fileLocation = "$textfile";
$fh = fopen($fileLocation, 'w   ') or die("Something went
wrong!");
$stringToWrite = "=$varijabla"; // Vrijednost za upisivanje
fwrite($fh, $stringToWrite); // Upisivanje
fclose($fh);
?>
```

Slika 6.6 Programski kod PHP datoteke

Kada se u PHP datoteci dohvati vrijednost pomoću „GET“ metode, vrijednost se upisuje u tekstualnu datoteku „motorcetiriiznos.txt“.

Sljedeći Javascript program prikazuje spremanje svih četiri vrijednosti motora u tekstualnu datoteku. Jednako kao i sa jednom vrijednošću samo se ovdje dodaju još tri.

```
var xmlhttp_dva = new XMLHttpRequest();
          xmlhttp_dva.onreadystatechange = function () {
               if(xmlhttp_dva.readyState == 4 &&
xmlhttp_dva.status == 200)
               {
               }
};
     xmlhttp_dva.open("GET", "vrijednosti.php?vrijednost=" +
iznos_jedan + "MD," + iznos_dva + "MT," + iznos_tri + "MC," +
iznos_cetiri, true);
     xmlhttp_dva.send();
}
```

Slika 6.7 Javascript programski koda za spremanje svih vrijednosti

6.3. Izrada programa za Arduino

Kako bi se mogao koristiti Ethernet modul, najprije je potrebno u Arduino razvojnom okruženju dodati knjižicu SPI i Ethernet.

```
#include <SPI.h>
#include <Ethernet.h>
```

Knjižica SPI je „Serial Peripheral Interface“, odnosno sinkronizirani serijski podatkovni protokol. Knjižica Ethernet služi za upravljanje Ethernet modulom.

Za Ethernet modul u Arduino razvojnom okruženju potrebno je definirati MAC adresu Ethernet modula i IP adresu. Ponekad je dovoljno dodati samo MAC adresu.

```
byte mac[] = { 0x90, 0xA2, 0xDA, 0x0F,
0xDE, 0xB3 };
IPAddress ip(192, 168, 1, 101);
```

Nakon što se definira MAC adresa i IP adresa, Arduino razvojno okruženje i Ethernet modul se povezuju s usmjerivačem.

```
// start the Ethernet connection:
  Serial.println("Initialize Ethernet with DHCP:");
  if (Ethernet.begin(mac) == 0) {
    Serial.println("Failed to configure Ethernet using DHCP");
    // Check for Ethernet hardware present
    if (Ethernet.hardwareStatus() == EthernetNoHardware) {
      Serial.println("Ethernet shield was not found.  Sorry,
can't run without hardware. :(");
      while (true) {
        delay(1); // do nothing, no point running without
Ethernet hardware
      }
    }
    if (Ethernet.linkStatus() == LinkOFF) {
      Serial.println("Ethernet cable is not connected.");
    }
    // try to congifure using IP address instead of DHCP:
    Ethernet.begin(mac, ip);
  } else {
    Serial.print("  DHCP assigned IP ");
    Serial.println(Ethernet.localIP());
  }
  // give the Ethernet shield a second to initialize:
  delay(1000);
  Serial.print("connecting to ");
  Serial.print(server);
  Serial.println("...");
```

Slika 6.8 Programski kod za dohvaćanja IP adrese

Arduino programski kod kao na slici 6.8 traži IP adresu Ethernet modula pomoću DHCP-a. Jednako tako prvo se provjerava da li je Ethernet modul spojen sa Arduino razvojnom platformom, zatim se provjerava da li je Ethernet kabel odnosno UTP kabel, priključen na Ethernet modul.

```
if (client.connect(server, 80)) {
    Serial.println("connected");
    client.print("GET /arduino/vrijednostislidera.txt?");
    client.println( " HTTP/1.1");
    client.println("Host: localhost");
    client.println("Connection: close");
    client.println();
    if(client.find("=")){
      int vrijjedan = client.parseInt();
      Serial.println(vrijjedan);
      if(vrijjedan > 0 && vrijjedan < 180)  {
        int pos = vrijjedan;
        myservo1.write(pos, 15, true);
      }
    }
```

Slika 6.9 Programski kod dohvaćanja datoteke s poslužitelja

U programskom kodu kao na slici 6.9, Ethernet modul radi kao klijent. Arduino se pomoću Ethernet modula spaja na poslužitelj i dohvaća vrijednosti iz tekstualne datoteke, a zatim vrijednosti prosljeđuje na servo motore. Navedeni programski kod je samo za jedan servo motor, ali za sve ostale elektromotore vrijedi isti princip. Arduino dohvati tekstualnu datoteku s poslužitelja u obliku „=34MD,75MT,25MC,63“. U Arduino razvojnom okruženju potrebno je iz tog znakovnog niza izdvojiti vrijednosti koje se potom prosljeđuju na servo motor.

Za prvi servo motor vrijednost se pomoću „client.find(„=“)“ izdvaja na način da se u znakovnom nizu traži znak jednakosti. Znak jednakosti može se pojaviti samo na početku znakovnog niza tako da znak jednakosti može poslužiti kao navigator gdje se nalazi vrijednost motora jedan. Nakon što se dobije vrijednost motora jedan, tu vrijednost je sada potrebno staviti u „int“ odnosno u cjelobrojni zapis. To se odrađuje tako da nekoj novoj varijabli „vrijjedan“ proslijedimo znakovni zapis kao cjelobrojni zapis pomoću „client.parseInt()“. Nakon brojevne vrijednosti prvog motora pojavljuje se indikator za drugi motor kao znakovni zapis. U cjelobrojni zapis moguće je prenijeti samo brojevni niz.

```
if(client.find("MD,"))  {
        int vrijdva = client.parseInt();
        Serial.println(vrijdva);
        if(vrijdva > 0 && vrijdva < 180){
            int posdva = vrijdva;
            myservo2.write(posdva, 15, true);
          }
    }
```

Slika 6.10 Programski kod za prosljeđivanje vrijednosti na servo motor

Za drugi servo motor je slično kao i za prvi, samo što se ovdje vrijednost traži pomoću indikatora „MD," te prosljeđuje u cjelobrojni zapis vrijednost koja se nalazi nakon indikatora.

```
if(client.find("MT,")){
        int vrijtri = client.parseInt();
        Serial.println(vrijtri);
        if(vrijtri > 0 && vrijtri < 180){
           int postri = vrijtri;
           myservo3.write(postri, 15, true);
          }
     }
    if(client.find("MC,")){
        int vrijcetiri= client.parseInt();
        Serial.println(vrijcetiri);
        if(vrijcetiri > 0 && vrijcetiri < 180){
           int poscetiri = vrijcetiri;
           myservo4.write(poscetiri, 15, true);
         }
    }
```

Slika 6.11 Programski kod za prosljeđivanje vrijednosti na servo motor

Za motor tri i četiri je također slično kao za motor jedan i dva, samo što su ovdje indikatori „MT," i „MC,"

Knjižica „VarSpeedServo" služi za komunikaciju s servo motorima. Bez te knjižice nije moguće dodijeliti servo motoru digitalni izlaz na Arduino razvojnoj platformi, ili prenijeti vrijednost na servo motor.

```
#include <VarSpeedServo.h>
```

Nakon dodavanja knjižice potrebno je definirati varijable za servo motore, te u „void setup()" dodijeliti servo motorima izlaze.

```
VarSpeedServo myservo1;
VarSpeedServo myservo2;
VarSpeedServo myservo3;
VarSpeedServo myservo4;
```

Izlazi dodijeljeni za servo motore označeni su od 5 do 8.

```
myservo1.attach(5);
myservo2.attach(6);
myservo3.attach(7);
myservo4.attach(8);
```

VarSpeedServo knjižica omogućava vrlo jednostavno korištenje servo motora. Tom knjižicom može se jednostavno kontrolirati brzinu motora. Vrijednost 15 označava brzinu servo motora, a ta brzina može biti u rasponu od 0 do 255.

```
myservo4.write(poscetiri, 15, true);
```

Kako bi se mogao koristiti RTC modul potrebno je dodati još dvije knjižice.

```
#include <Wire.h>
#include "RTClib.h"
```

Nakon toga se definira RTC te dodati popis dana u tjednu, kako bi se to kasnije moglo zapisati u tekstualnu datoteku.

```
RTC_DS1307 rtc;
char daysOfTheWeek[7][12] = {"Sunday", "Monday", "Tuesday",
"Wednesday", "Thursday", "Friday", "Saturday"};
```

Nakon definicije RTC-a i dodavanja popisa dana, potrebno je definirati varijable u koje će se spremati vrijeme.

```
String godinadva = "0";
String mjesecdva = "0";
String dandva = "0";
String satdva = "0";
String minutadva = "0";
String sekundadva = "0";
int godina;
int mjesec;
int dan;
int sat;
int minuta;
int sekunda;
int spremanjevremena = 0;
```

Unutar „void setup()“ definira se RTC modul i dodjeljuje vrijeme od kojeg se počinje dalje računati.

```
if (! rtc.begin()) {
    Serial.println("Couldn't find RTC");
    while (1);
  }
  if (! rtc.isrunning()) {
    Serial.println("RTC is NOT running!");
    // following line sets the RTC to the date & time this
sketch was compiled
    //rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
    // This line sets the RTC with an explicit date & time, for
example to set
    // January 21, 2014 at 3am you would call:
    // rtc.adjust(DateTime(2014, 1, 21, 3, 0, 0));
    // September 9, 2019 at 9am, 0, 0
    rtc.adjust(DateTime(2019, 9, 9, 9, 0, 0));
```

Slika 6.12 Programski kod za namještanje vremena RTC modula

Ako se postavi „rtc.adjust(DateTime(2019,9,9,9,0,0))“ kao što je prikazano na slici 6.12, tada RTC modul počinje računati od 9.rujna 2019.g. u 9 sati.

U glavnom programu koji se neprekidno ponavlja, dodaje se programski kod kao na slici 6.13. Pomoću ovog programskog koda, u ranije definirane varijable, pohranjuje se trenutno vrijeme iz RTC modula. Zatim se varijable prenose u znakovni zapis kako bi se mogle slati i pohranjivati na poslužitelju.

```
DateTime now = rtc.now();

  godina = now.year();
  mjesec = now.month();
  dan = now.day();
  sat = now.hour();
  minuta = now.minute();
  sekunda = now.second();

  godinadva = String(godina);
  mjesecdva = String(mjesec);
  dandva = String(dan);
  satdva = String(sat);
  minutadva = String(minuta);
  sekundadva = String(sekunda);
```

Slika 6.13 Programski kod pohranjivanja vremena iz RTC modula u varijable

```
String zagradajedan = "(";
  String zagradadva = ")";
if(spremanjevremena == 1){
  if (client.connect(server, 80)) {
    Serial.println("connected");
    client.print("GET /arduino/vrijeme.php?vrijednost=" +
godinadva + "/" + mjesecdva + "/" + dandva + zagradajedan +
daysOfTheWeek[now.dayOfTheWeek()] + zagradadva + satdva + ":" +
minutadva + ":" + sekundadva);
    client.println( " HTTP/1.1");
    client.println("Host: localhost");
    client.println("Connection: close");
    client.println();
    client.stop();
  } else {
    Serial.println();
    Serial.println("not connected");
    client.stop();
  }
  spremanjevremena = 0;
  }
```

Slika 6.14 Programski kod slanja vremena na poslužitelj

Kada se vrijednosti proslijede u varijable, potrebno ih je poslati na poslužitelj kao što je prikazano na slici 6.14, odnosno poziva se PHP datoteka koja sprema vrijeme u tekstualnu datoteku na poslužitelju. Kako bi se smanjilo vrijeme izvršavanja glavnog programa, potrebno je dodati varijablu „spremanjevremena“. Na opisani način se vrijeme ne sprema uvijek, nego samo kad ta varijabla poprimi vrijednost 1. Navedena varijabla poprimi vrijednost 1 unutar prekidne rutine koja se pokreće pritiskom na tipku.

Prekidnu rutinu potrebno je prvo definirati uz pomoć „attachInterrupt“ naredbe. Unutar te naredbe odabire se digitalni izlaz kojemu se dodjeljuje prekidna rutina, dodjeljuje se naziv prekidne rutine te da li će se prekidna rutina pokrenuti na uzlazni ili padajući brid.

```
attachInterrupt(digitalPinToInterrupt(3), vrijeme, RISING);
```

U ovoj prekidnoj rutini mijenja se samo stanje varijable.

```
void vrijeme(){
    spremanjevremena   =
1;
  }
```

Pritiskom na tipku varijabla se postavi u jedan, a trenutno vrijeme se pošalje u PHP datoteku koja poslanu vrijednost pohranjuje u tekstualnu datoteku. Na kraju slanja varijabla

„spremanjevremena“ postavi se opet u nulu, tako da se taj dio glavnog programa ne izvršava stalno, što pomaže u skraćivanju vremena trajanja glavnog programa.

PHP datoteka za spremanje vremena radi isto kao i PHP datoteke za spremanje vrijednosti slidera.

```
<?php
$varijabla = $_GET["vrijednost"]; // Dohvaćanje vremena
$textfile = "prikazvremena.txt"; // Text document u koji će biti
zapisano vrijeme
 $fileLocation = "$textfile";
$fh = fopen($fileLocation, 'w   ') or die("Something went
wrong!");
$stringToWrite = "=$varijabla";
fwrite($fh, $stringToWrite); // Upisivanje
fclose($fh);
?>
```

Slika 6.15 PHP datoteka koja pohranjuje vrijeme u tekstualnu datoteku

Spremljeno vrijeme se sprema u obliku:

```
=2019/9/9(Monday)14:4:39
```

Definirana je još jedna prekidna rutina uz pomoć „attachInterrupt“ naredbe.

```
attachInterrupt(digitalPinToInterrupt(2), buttonPressed, RISING);
```

Ova prekidna rutina se pokreće pomoću pritiska na tipku. Kada se tipka pritisne pokreće se prekidna rutina „buttonPressed“. U ovoj prekidnoj rutini prvo se postavlja varijabla „gumbpritisnut“ u vrijednost jedan. Ta varijabla služi za isto za što služi varijabla „spremanjevremena“. Zatim se provjerava da li je svjetleća dioda uključena. Ako je dioda uključena tada je isključujemo. Isključivanje i uključivanje svjetleće diode odrađuje se pomoću „digitalWrite“ naredbe, na način da se pomoću „HIGH“ uključuje, a pomoću „LOW“ isključuje. Varijabli „stanje“ se dodjeljuje vrijednost 1, ako je dioda uključena, ili vrijednost 0, ako je isključena. Vrijednost varijable „stanje“ pohranjuje se u tekstualnu datoteku.

```
void buttonPressed(){
  gumbpritisnut = 1;
    if(digitalRead(led) == HIGH){
        digitalWrite(led, LOW);
        stanje = "0";
      } else{
          digitalWrite(led,HIGH);
          stanje = "1";
        }
}
```

Slika 6.16 Prekidna rutina svjetleće diode

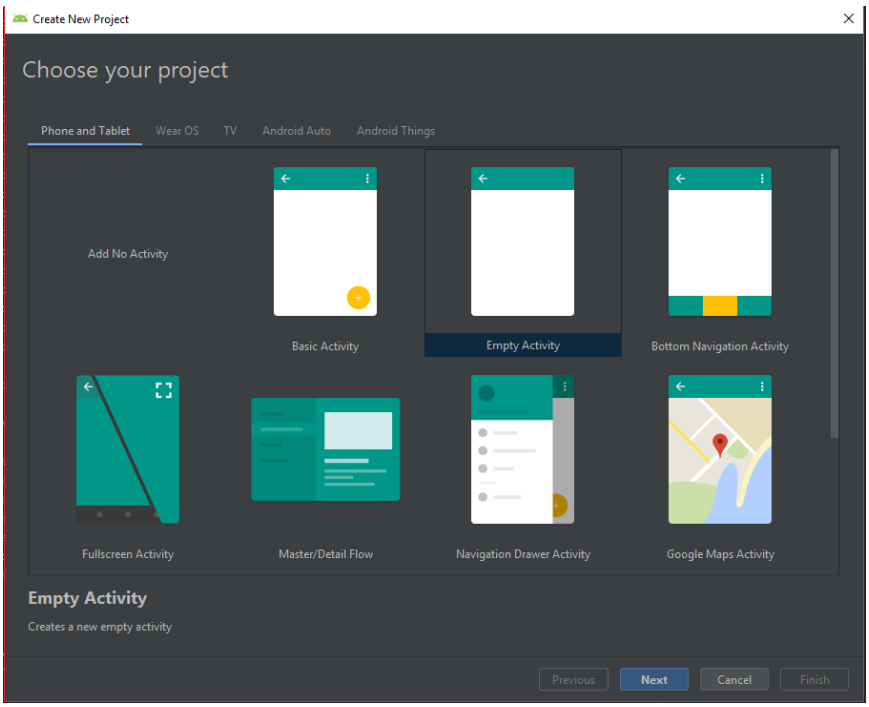
Spremanje stanja svjetleća dioda sprema se na sličan način kao i spremanje vremena. Kada je varijabla „gumbpritisnut“ postavljena u 1, tada se izvršava dio glavnog programa zaslužan za slanje stanja svjetlećih dioda na poslužitelj, odnosno u PHP datoteku koja to stanje pohrani u tekstualnu datoteku.

```
if(gumbpritisnut == 1){
  if (client.connect(server, 80)) {
    Serial.println("connected");
    client.print("GET /arduino/stanjegumba.php?vrijednost=" +
stanje);
    client.println( " HTTP/1.1");
    client.println("Host: localhost");
    client.println("Connection: close");
    client.println();
  client.stop();
  } else {
    Serial.println();
    Serial.println("not connected");
    client.stop();
  }
  gumbpritisnut = 0;
}
```

Slika 6.17 Programski kod slanja stanja svjetleće diode na poslužitelj

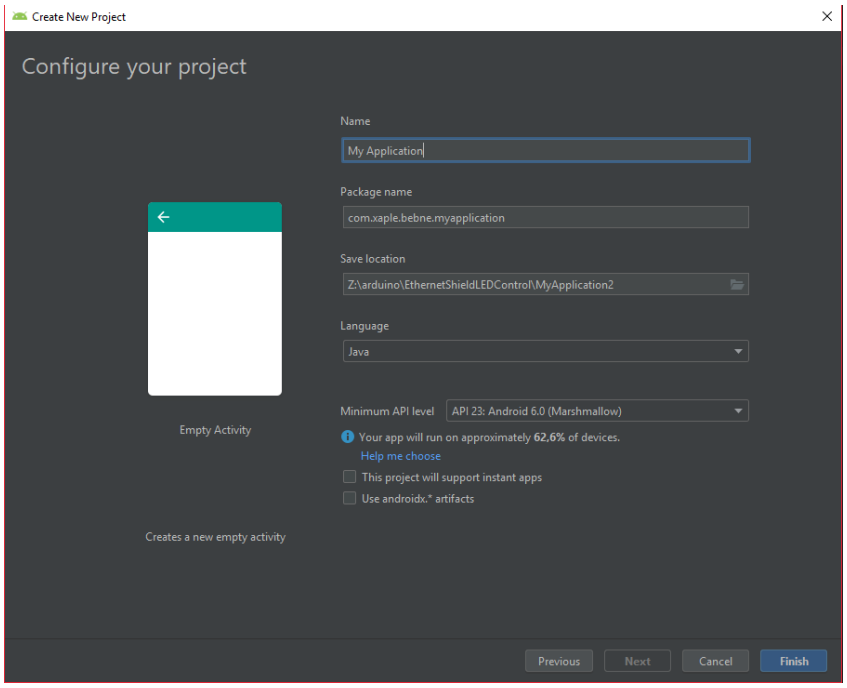
6.4. Izrada mobilne aplikacije

Mobilna aplikacija izrađena je za Android operacijski sustav. Program za izradu aplikacija koje podržava Android sustav zove se Android Studio. Kako bi se u Android Studio pokrenuo novi projekt, potrebno je otići na „File -> New -> New Project“, nakon čega se otvara prozor u kojemu se odabire početni oblik aplikacije. U ovoj aplikaciji odabrano je „Empty Activity“.



Slika 6.18 Prikaz stvaranja novog projekta u Android Studio

Nakon što se odabere izgled aplikacije, otvara se prozor u kojem je potrebno navesti ime aplikacije i mjesto gdje će se na računalu aplikacija pohraniti. Također je potrebno odabrati za koju verziju Android operacijskog sustava će se raditi aplikacija. Ova aplikacija napravljena je za Android 6.0 odnosno „Marshmallow“. Android Studio podržava sve Android sustave od Android 2.3 pa do Android 9.0. Može se odabrati i programski jezik u kojem će aplikacija biti programirana. Na odabir dolaze programski jezik Java i programski jezik Kotlin. Ova aplikacija napravljena je u programskom jeziku Java.



Slika 6.19 Prikaz odabira naziva aplikacije u Android Studio.

Nakon početnog konfiguriranja postavki buduće aplikacije, potrebno je dodati neke dijelove unutar „Manifest“ datoteke. „Manifest“ datoteka je xml datoteka, a sadrži osnovne informacije o aplikaciji. Unutar „Manifest“ datoteke se dodjeljuju dopuštenja aplikacije. Ako aplikacija koristi Internet, potrebno je aplikaciji unutar „Manifest“ datoteke omogućiti pristupanje Internet. Ako aplikacija koristi neki senzor, također je potrebno prvo unutar „Manifest“ datoteke omogućiti pristup tom senzoru. Unutar „Manifest“ datoteke moguće je promijeniti naziv aplikacije i izgled ikone aplikacije.

Unutar „Manifest“ datoteke potrebno je dodati:

```
<uses-permission android:name="android.permission.INTERNET" />
```

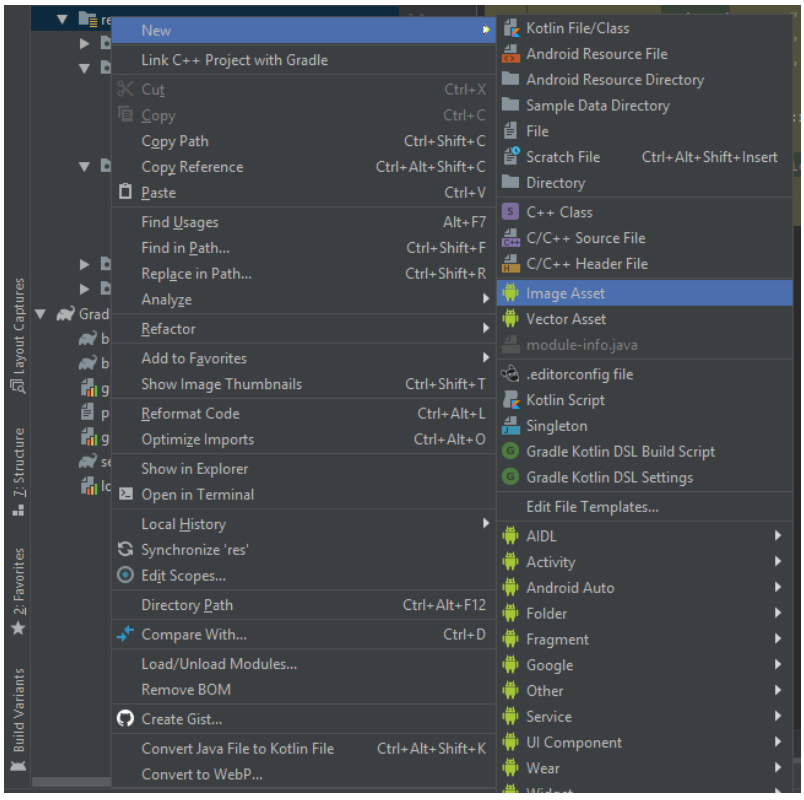
Čime se aplikaciji dozvoljava korištenje Interneta.

Unutar „Manifest“ datoteke može se promijeniti izgled ikone aplikacije.

```
android:icon="@mipmap/ic_launcher_dva"
```

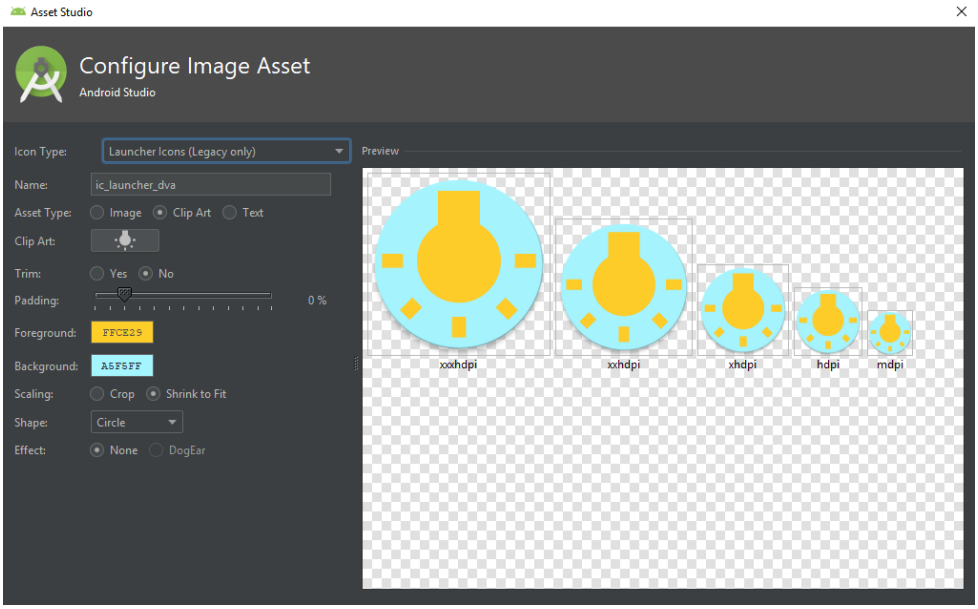
Najprije je potrebno dodati novu ikonu kako bi se stara ikona mogla promijeniti.

Ikone za aplikaciju ili za izbornik mogu se dodati na način da se pritisne desni klik miša na „res“, a potom na „New“, te na „Image Asset“.



Slika 6.20 Prikaz dodavanje ikona za aplikaciju

Otvara se prozor u kojem se mogu odabrati neke ikonice koje već postoje unutar Android Studia. Može im se mijenjati boja i veličina.



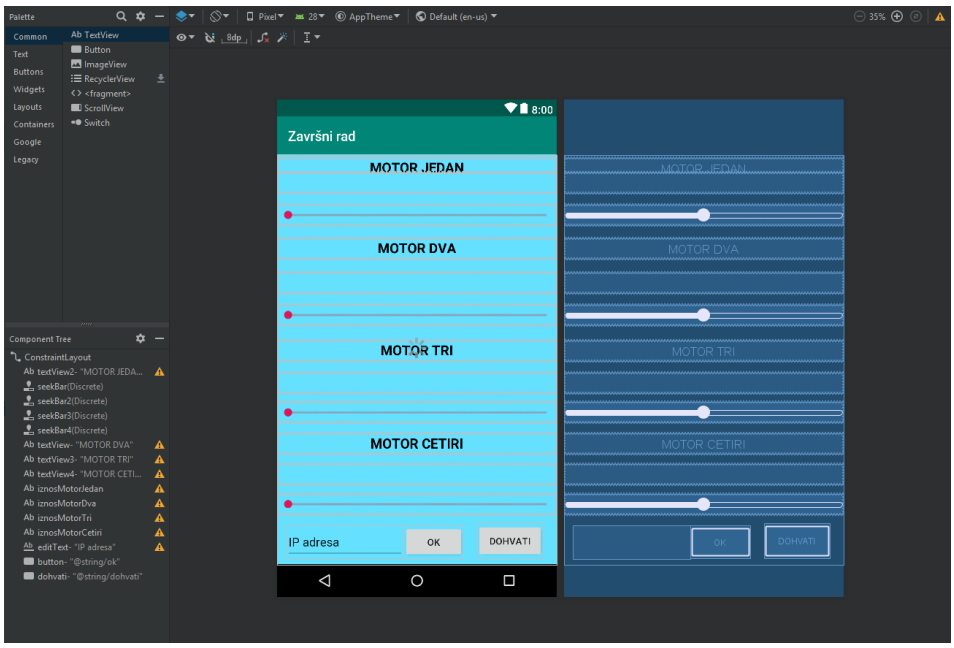
Slika 6.21 Prikaz dodavanja ikona za aplikaciju

Unutar datoteke „build.gradle(Modlue:app)“ potrebno je dodati „Dependency“. Tako se dodaje vanjska knjižica u Andriod Studio. U ovom slučaju dodano je:

```
implementation 'com.loopj.android:android-async-http:1.4.9'
```

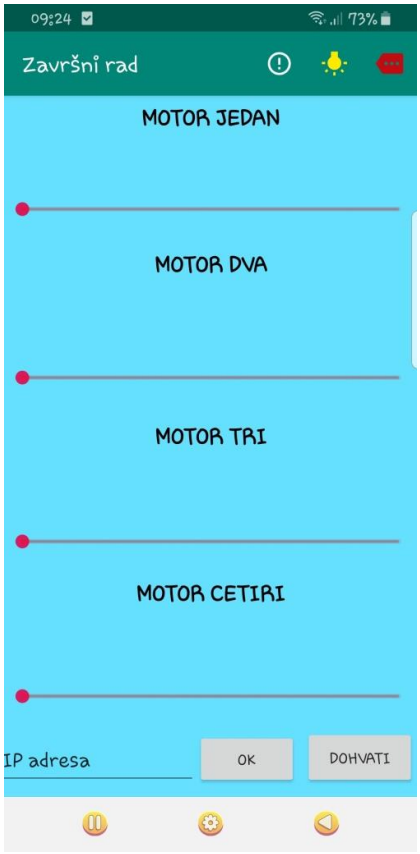
Potrebno je dodati ovaj dio zbog HTTP konekcije koja će biti potrebna kasnije, prilikom spajanja na poslužitelj i dohvaćanja vrijednosti.

Nakon toga je potrebno dodati izgled aplikacije. To se može pomoću xml programskog koda ili pomoću dizajnera. U ovoj aplikaciji neke stvari su odrađene pomoću dizajnera a neke pomoću xml-a.



Slika 6.22 Izgled početnog zaslona u Android Studio

Elementi aplikacije kao što su slider-i, gumb, polje za unos itd, mogu se jednostavno unijeti na zaslon aplikacije. Svakom elementu dodijeli se id. Id je potreban kako bi se u glavnom programu taj element mogao koristiti, budući da ga je potrebno pronaći pomoću id-a. Pomoću dizajnera ovo se može lako posložiti. Sve navedeno može se promijeniti i u xml-u samo što tamo ima puno više pisanja. Neke dijelove je jednostavnije promijeniti u xml-u, kao što su boja teksta, ili poravnanje teksta.



Slika 6.23 Izgled početnog zaslona na mobilnom uređaju

```
<TextView
    android:id="@+id/iznosMotorTri"
    android:layout_width="0dp"
    android:layout_height="0dp"
    android:layout_marginBottom="13dp"
    android:textAlignment="center"
    android:textColor="#000000"
    android:textSize="15dp"
    android:textStyle="bold"
    app:layout_constraintBottom_toTopOf="@+id/seekBar3"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView3" />

<TextView
    android:id="@+id/iznosMotorCetiri"
    android:layout_width="0dp"
    android:layout_height="0dp"
    android:layout_marginBottom="14dp"
    android:textAlignment="center"
    android:textColor="#000000"
    android:textSize="15dp"
    android:textStyle="bold"
    app:layout_constraintBottom_toTopOf="@+id/seekBar4"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView4" />
```

Slika 6.24 Izgled xml programskog koda unutar Android Studia

Naziv datoteke koja sadrži izgled početnog zaslona je „activity_main.xml“.

Java datoteka koja je povezana s datotekom „activity_main.xml“ zove se „MainActivity.Java“.

Unutar Java datoteke dodaje se funkcionalnost aplikacije, odnosno „MainActivity.Java“ povezana je uz početni zaslon tako da se dodaje funkcionalnost početnog zaslona.

Najprije je potrebno definirati varijable za slider.

```
private SeekBar slider, sliderDva, sliderTri, sliderCetiri;
```

Unutar „onCreate“ metode varijable slidera treba povezati s id-om iz xml-a.

```
slider=(SeekBar)findViewById(R.id.seekBar);
sliderDva=(SeekBar)findViewById(R.id.seekBar2);
sliderTri=(SeekBar)findViewById(R.id.seekBar3);
sliderCetiri=(SeekBar
findViewById(R.id.seekBar4);
```

Elementi iz xml-a se sa Java programskim kodom mogu povezati vrlo jednostavno, pomoću „findViewById“.

Nakon definiranja slider-a, potrebno je povezati i tekstualne elemente koji služe za prikaz vrijednosti pojedinih slider-a.

```
final TextView iznosMotorJedan = (TextView)
findViewById(R.id.iznosMotorJedan);
final TextView iznosMotorDva = (TextView)
findViewById(R.id.iznosMotorDva);
final TextView iznosMotorTri = (TextView)
findViewById(R.id.iznosMotorTri);
final TextView iznosMotorCetiri = (TextView)
findViewById(R.id.iznosMotorCetiri);
```

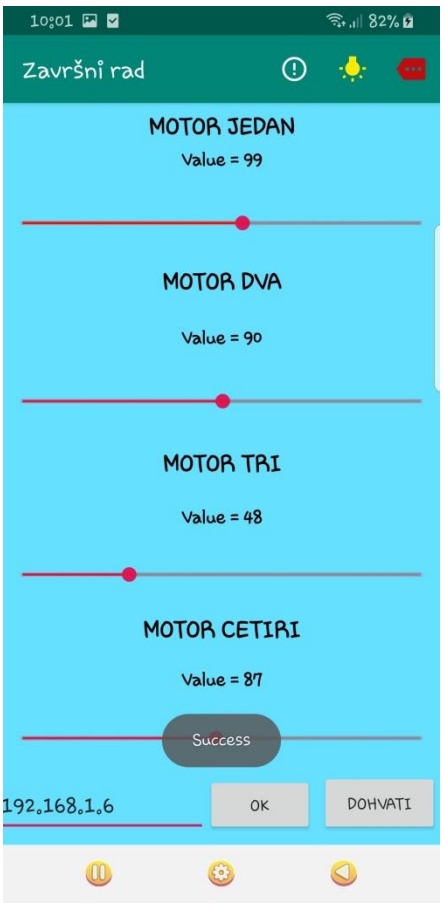
```
slider.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener() {
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress,
boolean fromUser) {
        iznosMotorJedan.setText("Value = " + progress);

    }
    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {
    }
```

Postavljanjem „setOnSeekBarChangeListener“ aktivira se pokretanje slidera prilikom promjene položaja slidera. Nadalje prilikom stvaranja „setOnSeekBarChangeListener“ Android Studio dodaje dijelove vezane uz to. Android Studio dodaje nekoliko dodatnih metoda kao što su „onProgressChanged“, „onStartTrackingTouch“ i „onStopTrackingTouch“. Unutar „onProgressChanged“ mijenja se vrijednost tekstualnom elementu povezanom s slider-om. U taj tekstualni element ispisujemo vrijednost slider-a.

```
@Override
public void onStopTrackingTouch(SeekBar seekBar) {
    vrijednost_jedan = seekBar.getProgress();
    AsyncHttpClient client = new AsyncHttpClient();
    client.get("http://" + adresa +
"/arduino/vrijednosti.php?vrijednost=" + vrijednost_jedan +
"MD," + vrijednost_dva + "MT," + vrijednost_tri + "MC," +
vrijednost_cetiri, new AsyncHttpResponseHandler() {
        @Override
        public void onStart() {
            // called before request is started
            Toast.makeText(getApplicationContext(), "Start",
Toast.LENGTH_LONG).show();
        }
        @Override
        public void onSuccess(int statusCode, Header[] headers,
byte[] responseBody) {
            Toast.makeText(getApplicationContext(), "Success",
Toast.LENGTH_LONG).show();
        }
        @Override
        public void onFailure(int statusCode, Header[] headers,
byte[] responseBody, Throwable error) {
            Toast.makeText(getApplicationContext(), "Failure",
Toast.LENGTH_LONG).show();
        }
```

Slika 6.26 Programski kod slanja vrijednost slidera na poslužitelj

U „onStopTrackingTouch“ dodaje se slanje vrijednosti slider-a na poslužitelj kao što je prikazano na slici 6.26. U ovoj metodi je najprikladnije poslati vrijednost, zato što se ova metoda pokrene tek kada se prst sa makne sa slider-a, odnosno s zaslona mobilnog uređaja. Vrijednost slider-a sprema se u varijablu „vrijednost_jedan“ pomoću „seekBar.getProgress()“. Zatim započinje HTTP konekcija s poslužiteljem gdje je potrebna, u ovom slučaju, IP adresa poslužitelja. Adresa poslužitelja se unosi pomoću gumba i polja za unos. Slanje na poslužitelj se odrađuje slično kao kod web stranice. U istu tekstualnu datoteku upisuju se sve četiri vrijednosti. Poziva se datoteka „vrijednosti.php“ pomoću koje se vrijednosti upisuju u tekstualnu datoteku. Android Studio stvori metode „onStart“, „onSuccess“, „onFailure“ i „onRetry“. Pomoću „toast“ poruke može se pratiti događanje HTTP konekcija, odnosno da li je konekcija započela, da li je konekcija uspjela ili neuspjela. „Toast“ poruka je poruka koja se ispisuje na zaslonu mobilnog uređaja. Ova poruka služi za vrlo kratke i brze poruke na aplikacijama.



Slika 6.25 Izgled početnog zaslona s toast porukom

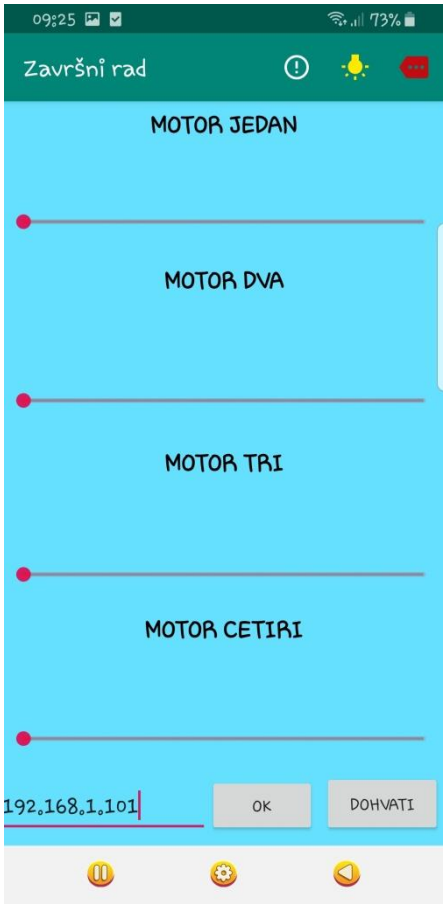
```
AsyncHttpClient client_dva = new AsyncHttpClient();
client_dva.get("http://" + adresa
+"/arduino/motorjedan.php?vrijednost=" + vrijednost_jedan,  new
AsyncHttpResponseHandler() {
    @Override
    public void onSuccess(int statusCode, Header[] headers,
byte[] responseBody) {

    }

    @Override
    public void onFailure(int statusCode, Header[] headers,
byte[] responseBody, Throwable error) {

    }
});
```

Slika 6.27 Programski kod slanja jedne vrijednosti na poslužitelj

Uz prvi slider postoji još jedna HTTP konekcija kao na slici 6.27 u kojoj se šalje ista vrijednost u drugu tekstualnu datoteku, koja služi samo za vrijednost prvog slidera. Kada se dohvaća samo vrijednost prvog slidera, lakše je obaviti dohvaćanje samo jedne vrijednosti.

```
Button button = (Button) findViewById(R.id.button);

editText = (EditText) findViewById(R.id.editText);

button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        adresa = editText.getText().toString();
    }
});
```

Preko id-a iz xml dijela početnog zaslona, potrebno je povezati gumb „ok“ i polje za unos. Polje za unos se koristi kako bi se napisala adresa poslužitelja, a sa gumbom „ok“ se ta adresa potvrđuje.



Slika 6.28 Izgled početnog zaslona s upisanom IP adresom

Nakon pritiska gumba „ok“ izvršava se „setOnClickListener“ dio programa. U tom dijelu programa se adresa upisana u polje za unos prenosi u varijablu „adresa“. Pomoću te adrese formira se ukupna adresa kamo se šalje ili prima vrijednost.

Koristi se još jedan gumb kojim se dohvaćaju vrijednosti koje se nalaze u tekstualnim datotekama na poslužitelju. Ako se vrijednosti promijene u web stranici, a u aplikaciji ne, tada se u aplikaciju moraju učitati vrijednosti koje su promijenjene u web stranici. Kako bi se učitale te vrijednosti koristi se ovaj dodatni gumb. Kada se unese i potvrdi adresa poslužitelja, može se pritisnuti na gumb „dohvati“. Vrijednosti se dohvate s poslužitelja i prenose na slider-e.

```
Button dohvati = (Button)
findViewById(R.id.dohvati);
```

Gumb s nazivom „dohvati“ postavlja se „setOnClickListener“ metoda koja se pokrene kad se pritisne na gumb.

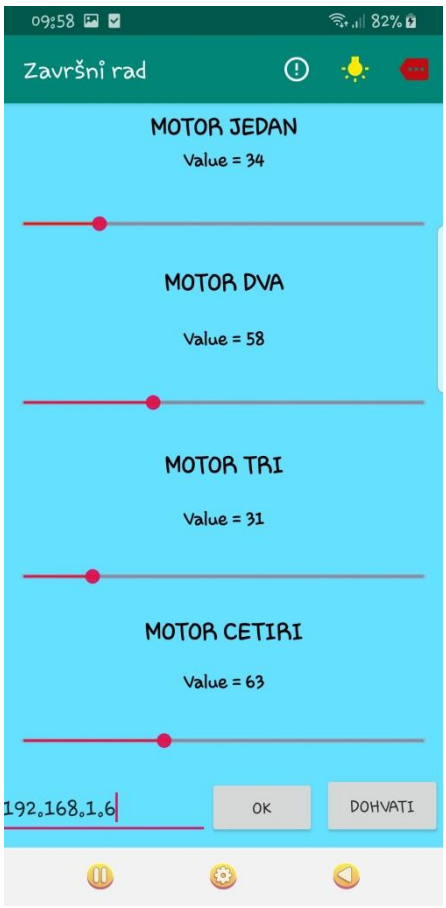
```
dohvati.setOnClickListener(new
View.OnClickListener() {
    @Override
    public void onClick(View v) {
}
```

Unutar „setOnClickListener“ metode dohvaćaju se vrijednosti s poslužitelja.

```
AsyncHttpClient client = new AsyncHttpClient();
client.get("http://" + adresa + "/arduino/iznosmotorajedan.txt",
new FileAsyncHttpResponseHandler(MainActivity.this) {
    @Override
    public void onFailure(int statusCode, Header[] headers,
Throwable throwable, File file) {
        Toast.makeText(getApplicationContext(), "Failure" ,
Toast.LENGTH_LONG).show();
    }
    @Override
    public void onSuccess(int statusCode, Header[] headers, File
file) {
        try {
            BufferedReader in = new BufferedReader(new
FileReader(file));
            String str;
            while ((str = in.readLine()) != null) {
                // str is one line of text; readLine() strips the
newline character(s)
                String novo = str.substring(1);
                int vrij = Integer.parseInt(novo);
                //Toast.makeText(getApplicationContext(), novo ,
Toast.LENGTH_LONG).show();
                slider.setProgress(vrij);
            }
            in.close();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
});
```

Slika 6.29 Programski kod dohvaćanja vrijednosti s poslužitelja

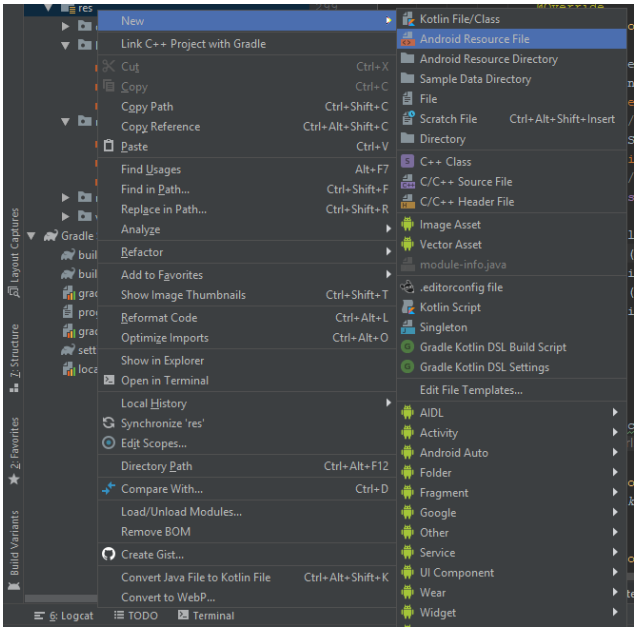
U programskom kodu na slici 6.29 se pomoću HTTP konekcije dohvaća tekstualna datoteka s poslužitelja u kojoj se nalazi vrijednost prvog motora. Nakon uspješnog dohvaćanja tekstualne datoteke potrebno je sadržaj te datoteke premjestiti u znakovni niz i potom ga premjestiti u cjelobrojni zapis. Na kraju se ta vrijednost postavi na slider pomoću „setProgress“.



Slika 6.30 Izgled početnog zaslona nakon dohvaćanja vrijednosti s poslužitelja

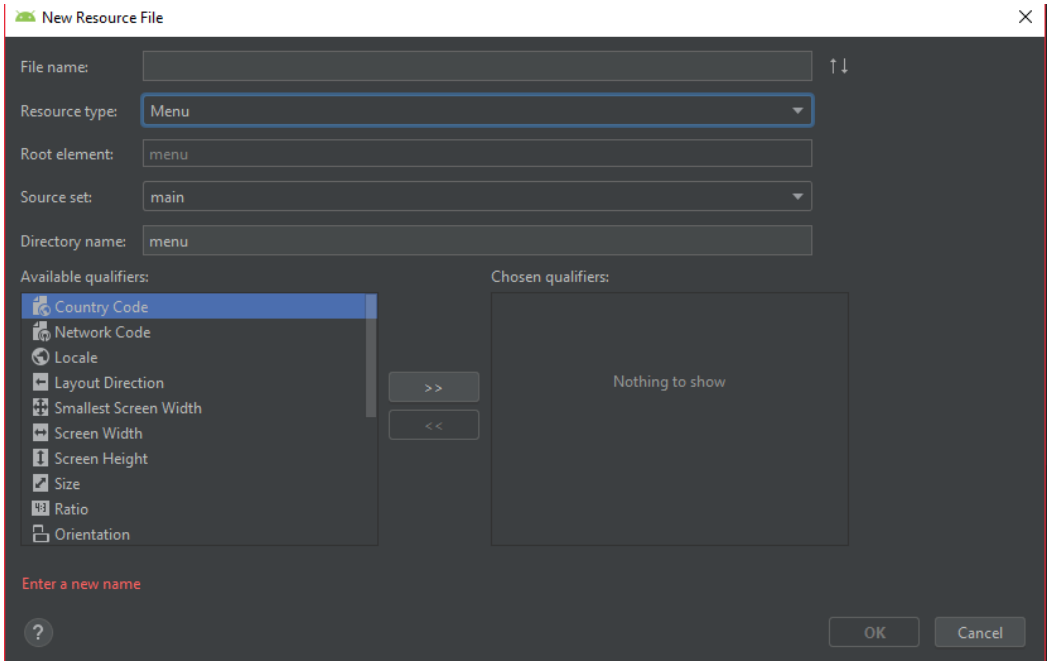
U aplikaciju je dodan izbornik, kojim se može pristupiti nekim dodatnim dijelovima. Moguće je pokrenuti i novi zaslon na kojem se vidi stanje svjetleće diode.

Zaslon se dodaje tako da se desnim klikom miša klikne na mapu „res“ te potom na „New“ i na kraju na „Android Resource File“.



Slika 6.31 Dodavanje datoteke u Android Studio

Navedenim postupkom otvara se novi prozor u kojem je potrebno odabrati „Menu“ kod polja „Resource type“.



Slika 6.32 Dodavanje datoteke izbornika u Android Studio

Kada „Menu“ odabira otvara se nova xml datoteka zaslužna za izbornik. U ovoj datoteci može se uređivati izbornik, dodavati ikone i ikonama dodati id.

Elemente izbornika najlakše je dodati unutar xml-a kao što je prikazano na slici 6.33, a ne dizajnera kao kod početnog zaslona. Pomoću „android:id“ dodaje se id pomoću kojeg se element može pronaći pomoću Jave. Pomoću „android:icon“ dodaje se ikona koja je prethodno bila dodana kao što je dodana i ikonica za aplikaciju. Pomoću „app:showAsAction“ može se kontrolirati kad će se te ikone od izbornika prikazivati na traci za izbornik. Ako se postavi „always“ ikone će se uvijek prikazivati. Ako se postavi „ifroom“ tada će se prikazati samo ako ima dovoljno mjesta.

```
<item
    android:id="@+id/oProgramu"
    android:icon="@drawable/detalji"
    android:title="o Programu"
    app:showAsAction="always"></item>

<item
    android:id="@+id/slika"
    android:icon="@drawable/slika"
    android:title="LED ON"
    app:showAsAction="always"></item>
<item
    android:id="@+id/zadatak"
    android:icon="@drawable/zadatak"
    android:title="Tekst zadatka"
    app:showAsAction="always"></item>
```

Slika 6.33 Xml programski kod izbornika

Kako bi se izbornik mogao koristit i vidjeti u našoj aplikaciji, mora se pomoću Jave uključiti u početni zaslon.

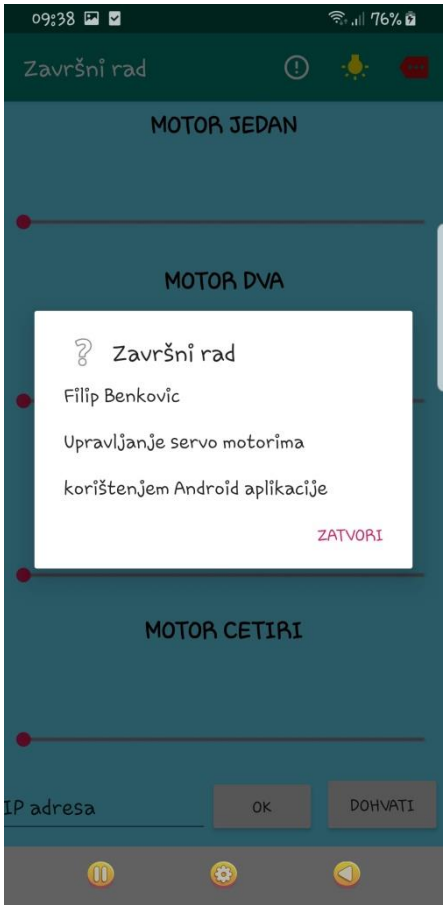
```
@Override
public boolean onCreateOptionsMenu(Menu
menu) {
    MenuInflater inflater =
getMenuInflater();
    inflater.inflate(R.menu.meni, menu);
    return true;
}
```

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()){
        case R.id.oProgramu:
            AlertDialog.Builder builder = new
AlertDialog.Builder(this);
            builder.setTitle("Završni rad");
            builder.setMessage("Filip Benkovic\n\nUpravljanje
servo motorima\n\npomoću android-a.");
            builder.setCancelable(false);
            builder.setPositiveButton("Zatvori", null);
            builder.setIcon(android.R.drawable.ic_menu_help);
            AlertDialog alert = builder.create();
            alert.show();
            return true;
        case R.id.slika:
            Intent novo = new
Intent(MainActivity.this,StanjeLED.class);
            novo.putExtra("key", adresa);
            startActivity(novo);
            return true;
        case R.id.zadatak:
            Intent tekstZadatka = new
Intent(MainActivity.this,TekstZadatka.class);
            startActivity(tekstZadatka);
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}
```

Slika 6.34 Dodavanje izbornika u Javi

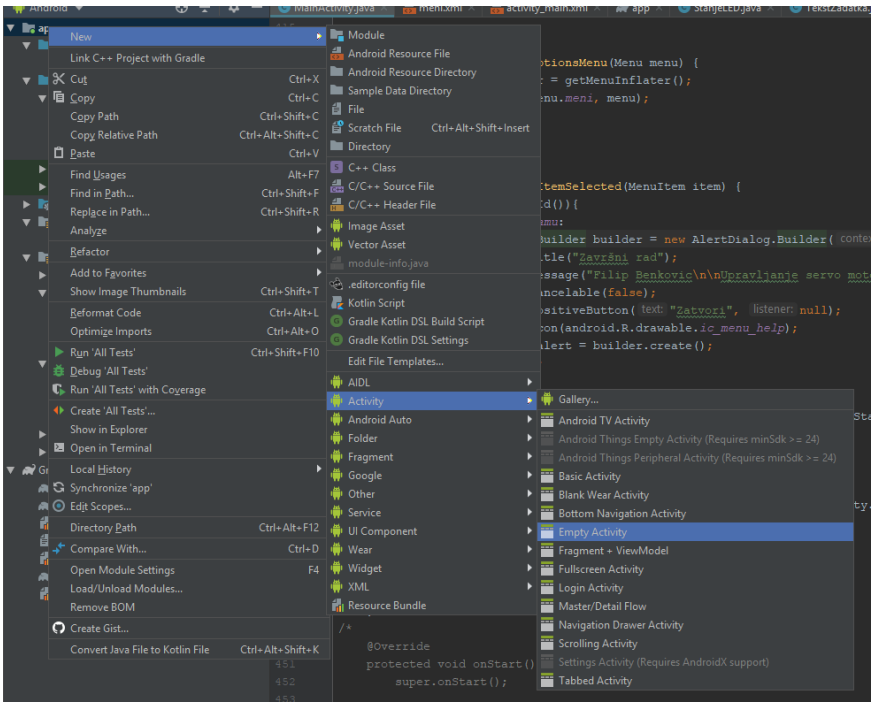
Kako bi se mogli koristiti dijelove izbornika, potrebno ih je u Javi programirati. Kako bi se izvršio neki zadatak pritiskom na ikonicu izbornika, potrebno je u Javi dodati funkcionalnosti pojedinih ikonica. Klikom na jednu od ikonica pokreće se „Alert dialog“. To je prozorčić koji se prikaže na zaslonu. Iduća ikonica pokreće novi „Activity“, odnosno novi zaslon na kojem se

može vidjeti stanje svjetleće diode. Zadnja ikonica izbornika pokreće novi zaslon u kojem piše zadatak završnog rada.



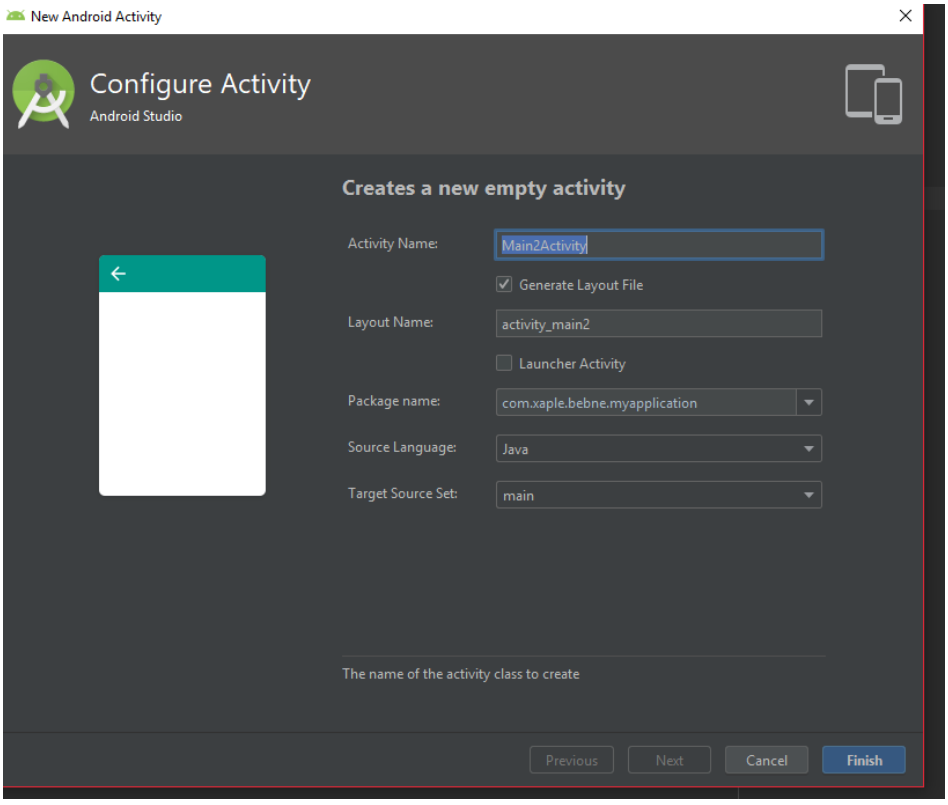
Slika 6.35 Izgled „Alert dialog-a“

Novi zaslon ili „activity“ pokreće se pomoću „startActivity“. Najprije je potrebno kreirati novi „activity“. To se radi tako da se na „app“ mapu pritisne desni klik miša te se odabire „Activity“ i potom „activity“ koji se želi dodati. U ovom slučaju to je „Empty Activity“.



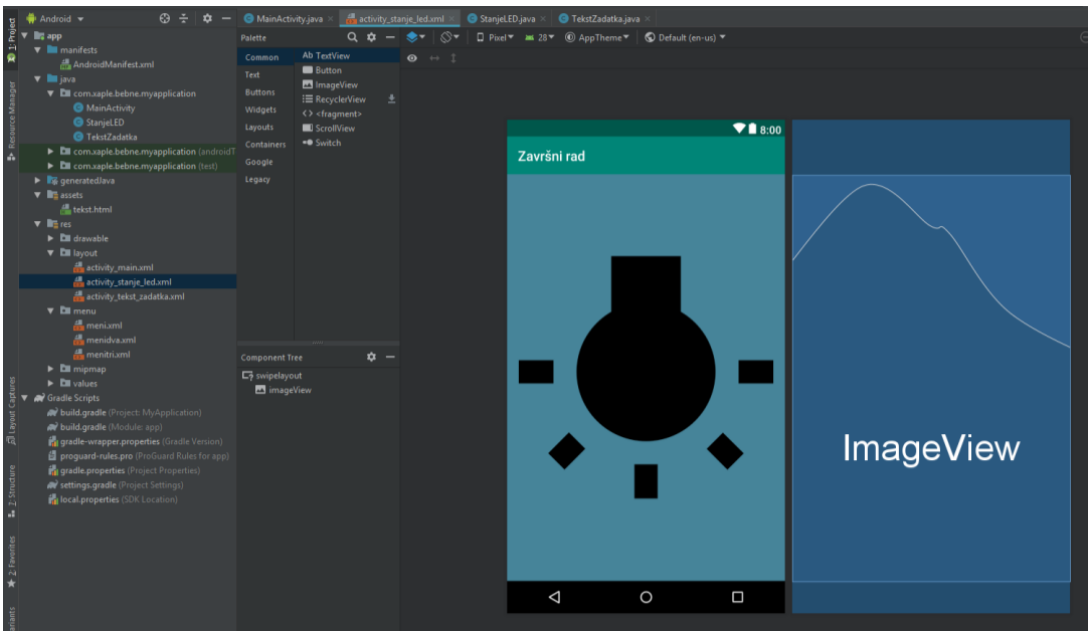
Slika 6.36 Dodavanje novog Activity-a

Nakon toga otvara se novi prozor u kojem se može promijeniti naziv novog „Activity-a“.

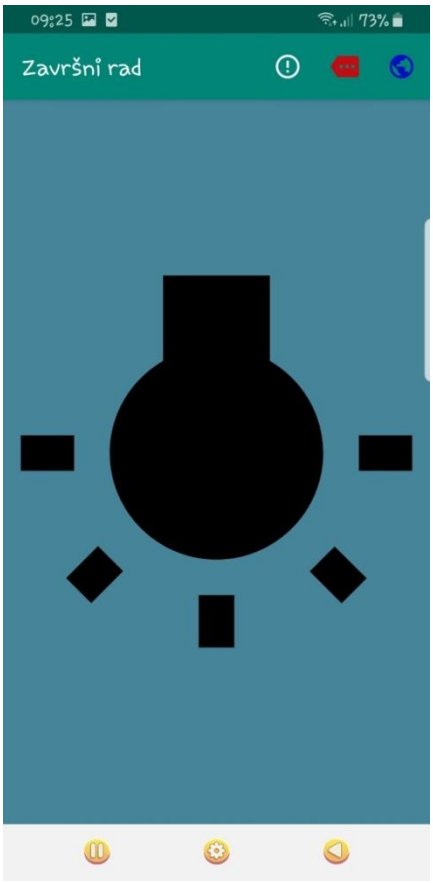


Slika 6.37 Dodavanje naziva novom „Activity-u“

Datoteka koja sadrži izgled zaslona zove se „activity_stanje_led.xml“. Ovdje je u dizajneru dodana slika crne žaruljice. Ova slika se pomoću Jave promijeni u žutu žaruljicu ovisno o stanju svjetleće diode. Slika se postavlja u „ImageView“ element.



Slika 6.38 Izgled zaslona s žaruljicom



Slika 6.39 Izgled zaslona mobilnog uređaja s žaruljom

U Java programskom kodu koji je povezan s „activity_stanje_led.xml“ najprije se preko id-a „ImageView-a“ poveže element iz xml-a s Javom.

```
final ImageView imageView = (ImageView)
findViewById(R.id.imageView);
```

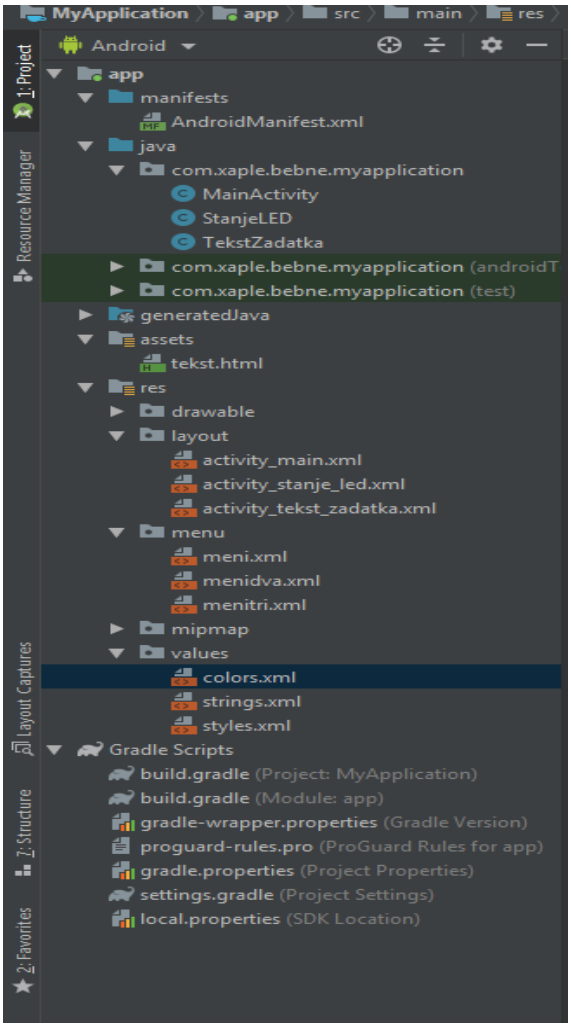
```
AsyncHttpClient clientcetiri = new AsyncHttpClient();
clientcetiri.get("http://" + value +
"/arduino/vrijednostgumba.txt", new
FileAsyncHttpResponseHandler(StanjeLED.this) {
    @Override
    public void onFailure(int statusCode, Header[] headers,
Throwable throwable, File file) {
        Toast.makeText(getApplicationContext(), "Failure" ,
Toast.LENGTH_LONG).show();
    }
    @Override
    public void onSuccess(int statusCode, Header[] headers, File
file) {
        try {
            BufferedReader in = new BufferedReader(new
FileReader(file));
            String str;
            while ((str = in.readLine()) != null) {
                // str is one line of text; readLine() strips
the newline character(s)
                String novo = str.substring(1);
                int vrij = Integer.parseInt(novo);
                //Toast.makeText(getApplicationContext(), novo ,
Toast.LENGTH_LONG).show();
                if(vrij == 1){

imageView.setImageResource(R.drawable.slika);
                } else{

imageView.setImageResource(R.drawable.slika_dva);
                }
            }
            in.close();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
});
```

Slika 6.40 Programski kod dohvaćanja stanja svjetleće diode

Pomoću HTTP konekcije kao što je prikazano na slici 6.40, dohvaća se tekstualna datoteka u kojoj se nalazi vrijednost stanja svjetleće diode. To se odradi na jednaki način kao kod dohvaćanja vrijednosti motora. Nakon što se dohvati vrijednost svjetleće diode, vrijednost se pohranjuje u neku varijablu. Prema vrijednosti varijable mijenja se slika na zaslonu. Ako je vrijednost 1 tada je slika žuta, a ako je vrijednost 0, slika je crna. Ovaj dio se odradi prilikom otvaranja ovog „Activity-a“ odnosno zaslona. Pritiskom na ikonicu u izborniku početnog zaslona otvara se ovaj zaslon s slikom žaruljice.

Ako se stanje svjetleće diode promijeni kada je prikazan zaslon s žaruljicom, to stanje se ne prikazuje trenutno. Ako se želi saznati da li se promijenilo stanje ili ne, mora se povući zaslon prema dolje. Tada se ponovno odradi dohvaćanje tekstualne datoteke s poslužitelja.

Kako bi se mogle dodijeliti boje na kružić za učitavanje, potrebno ih je dodati u Android Studio. Dodaju se u datoteku „colors.xml“ koja se nalazi unutar mape „values“. Unutar te datoteke dodaju se boje za daljnje korištenje.



Slika 6.41 Datoteka „colors.xml“

```
<color name="bojaCrvena">#ff0000</color>
<color name="bojaPlava">#0000ff</color>
<color name="bojaZuta">#ffff00</color>
```

```
final SwipeRefreshLayout swipeRefreshLayout =
(SwipeRefreshLayout) findViewById(R.id.swipelayout);
swipeRefreshLayout.setColorSchemeResources(R.color.bojaCrvena,
R.color.bojaPlava, R.color.bojaZuta);
swipeRefreshLayout.setOnRefreshListener(new
SwipeRefreshLayout.OnRefreshListener() {
    @Override
    public void onRefresh() {
        swipeRefreshLayout.setRefreshing(true);
        Handler handler = new Handler();
        handler.postDelayed(new Runnable() {
            @Override
            public void run() {
                AsyncHttpClient clientcetiri = new
AsyncHttpClient();
                clientcetiri.get("http://" + value +
"/arduino/vrijednostgumba.txt", new
FileAsyncHttpResponseHandler(StanjeLED.this) {
                    @Override
                    public void onFailure(int statusCode,
Header[] headers, Throwable throwable, File file) {
                        Toast.makeText(getApplicationContext(),
"Failure" , Toast.LENGTH_LONG).show();
                    }
                    @Override
                    public void onSuccess(int statusCode,
Header[] headers, File file) {
                        try {
                            BufferedReader in = new
BufferedReader(new FileReader(file));
                            String str;
                            while ((str = in.readLine()) !=
null) {
                                // str is one line of text;
readLine() strips the newline character(s)
                                String novo = str.substring(1);
                                int vrij =
Integer.parseInt(novo);

//Toast.makeText(getApplicationContext(), novo ,
Toast.LENGTH_LONG).show();
                                if(vrij == 1){

imageView.setImageResource(R.drawable.slika);
                                } else{

imageView.setImageResource(R.drawable.slika_dva);
                                }
                            }
                            in.close();
                        } catch (FileNotFoundException e) {
                            e.printStackTrace();
                        } catch (IOException e) {
                            e.printStackTrace();
                        }
                    }
                });
```

```
swipeRefreshLayout.setRefreshing(false);
            }
        },2000);
    }
});
```

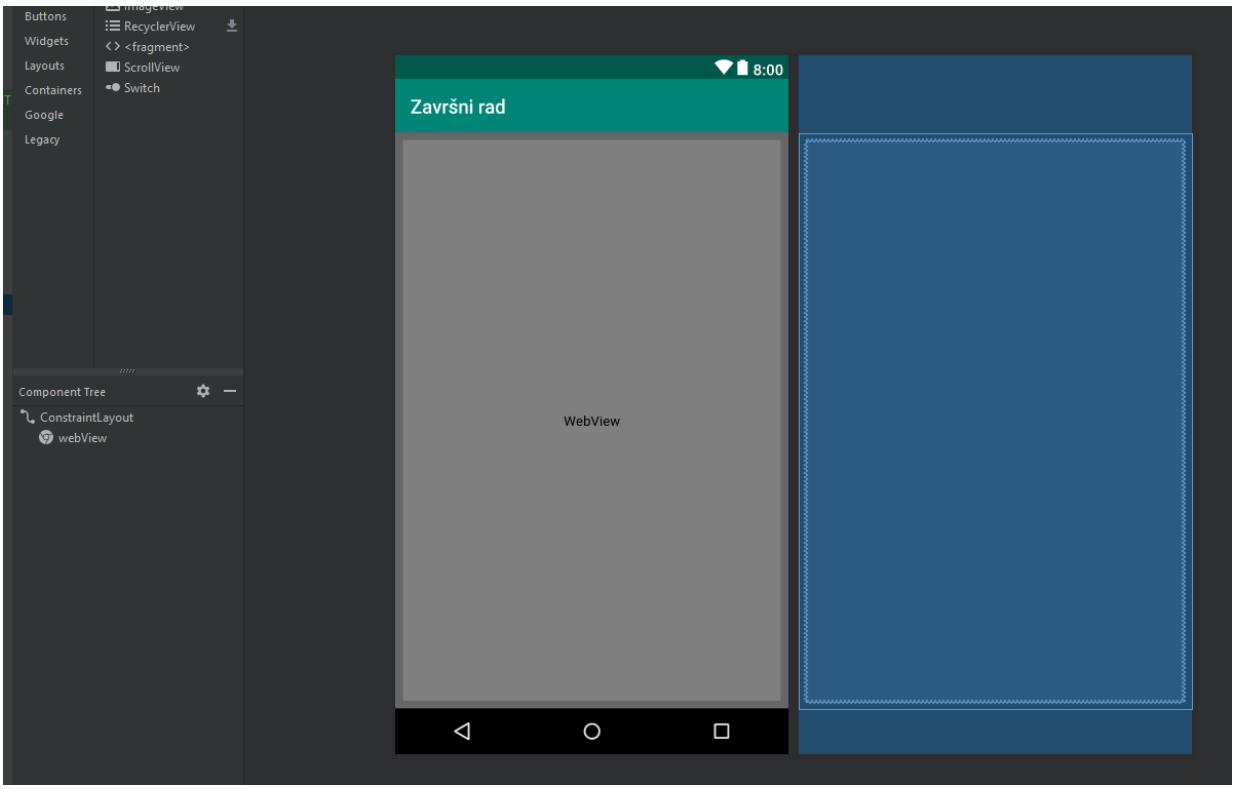
Slika 6.42 Programski kod promjene slike ovisno o stanju svjetleće diode

Na zaslon sa žaruljicom dodan je izbornik. Pomoću izbornika može se pokrenuti „Alert dialog“, vratiti na početni zaslon ili otvoriti zaslon na kojem se nalazi tekst zadatka završnog rada.

```
<item
        android:id="@+id/oProgramu"
        android:icon="@drawable/detalji"
        android:title="o Programu"
        app:showAsAction="always"></item>

<item
    android:id="@+id/zadatak"
    android:icon="@drawable/zadatak"
    android:title="LED ON"
    app:showAsAction="always"></item>

<item
    android:id="@+id/glavna"
    android:icon="@drawable/glavna"
    android:title="Glavna Strana"
    app:showAsAction="always"></item>
```

Slika 6.43 Programski kod izbornika

Posljednji dodani zaslon je zaslon na kojem se nalazi zadatak završnog rada. Na njemu se nalazi „WebView“ na koji se može dodati HTML dokument. Tako da „WebView“ ne otvara neku stranicu na Internetu, nego otvara HTML datoteku iz memorije.



Slika 6.44 Izgled zaslona s tekstom u Android Studio

Datoteka koja sadrži izgled zaslona zove se „activity_tekst_zadatka.xml“. Povezana je s Java datotekom „TekstZadatka.java“.

Unutar Java datoteke se povezuje „WebView" pomoću id-a, a Te na njega dodaje HTML datoteka koja će se otvarati.

```
protected void onCreate(Bundle savedInstanceState) {

    value = getIntent().getStringExtra("key2");
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_tekst_zadatka);

    WebView webView = (WebView) findViewById(R.id.webView);

    String filename = "tekst.html";

    webView.loadUrl("file:///android_asset/" + filename);
    //webView.getSettings().setJavaScriptEnabled(true);

}
```

Datoteka koja se dodaje u Android Studio prethodno je kreirana na računalu i prekopirana je u „assets“ mapu.

Slika 6.45 Mapa „assets“ i datoteka „tekst.html“

HTML datoteka koja je dodana u mapu „assets“ zove se „tekst.html“.

Slika 6.46 Izgled zaslona mobilnog uređaja na kojem je ispisan tekst

```
<!DOCTYPE html>
<HTML>
   <HEAD>
      <TITLE>Naslov rada</TITLE>
   </HEAD>
   <style type="text/css">
      .tekstjedan{
         font-weight: bold;
         font-size: 200%;
         text-align: center;
         font-family: "Comic Sans MS", cursive, sans-serif;
      }
      .tekstdva{
         font-weight: bold;
         font-size: 100%;
         text-align: left;
         font-family: "Comic Sans MS", cursive, sans-serif;
      }
      .teksttri{
         font-style: italic;
         font-size: 100%;
         font-family: "Comic Sans MS", cursive, sans-serif;
      }
   </style>
   <BODY style="background-color: lightyellow">

   <h1 class="tekstjedan">Upravljanje servo motorima korištenjem
Android aplikacije</h1>

   <p class="tekstdva">Realizirati sustav za upravljanje servo
motorima korištenjem Android aplikacije, mrežne komunikacije i
Arduino razvojnog sustava</p>

   <p class="teksttri">U radu je potrebno: <br> -osmisliti i
realizirati Android aplikaciju i web stranicu za upis podataka
na poslužitelj
                  <br> -osmisliti i realizirati mrežnu
komunikaciju poslužitelja i Arduino razvojnog sustava
                  <br> -osmisliti i realizirati upravljanje
servo motorima korištenjem Arduino razvojnog sustava</p>

   </BODY>
</HTML>
```

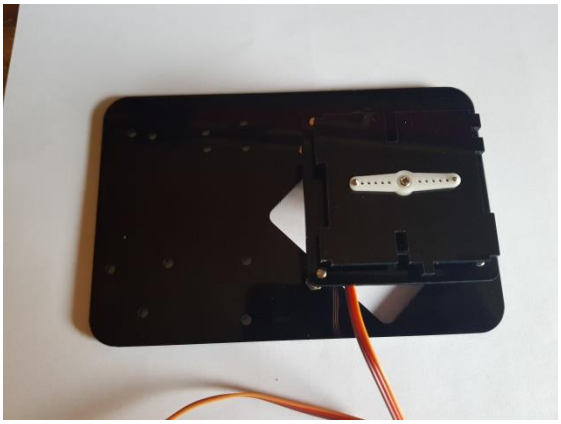
Slika 6.47 HTML programski kod

Na zaslonu koji prikazuje tekst zadataka u „WebView“ prikazu, nalazi se izbornik kojim se može vratiti na početni zaslon, na zaslon sa žaruljicom ili se može uključiti „Alert dialog“.

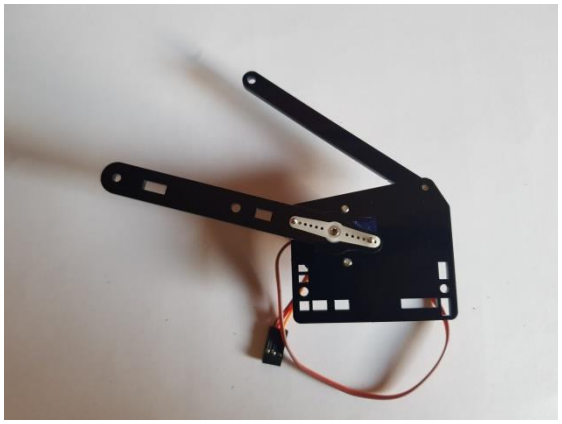
6.5. Sastavljanje makete

Slika 6.47 Sastavljanje podnožja makete

Slika 6.48 Izgled servo motora

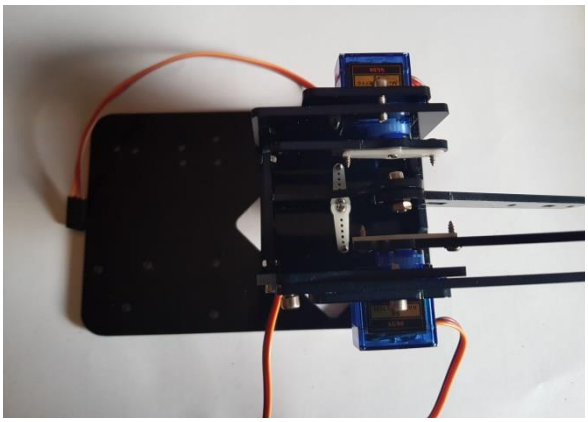


Slika 6.49 Sastavljanje podnožja makete s prvim motorom



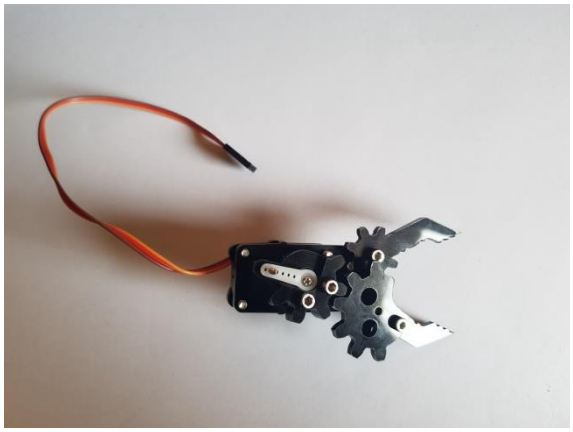
Slika 6.50 Sastavljanje drugog motora

Slika 6.51 Sastavljanje trećeg motora



Slika 6.52 Sastavljanje triju motora

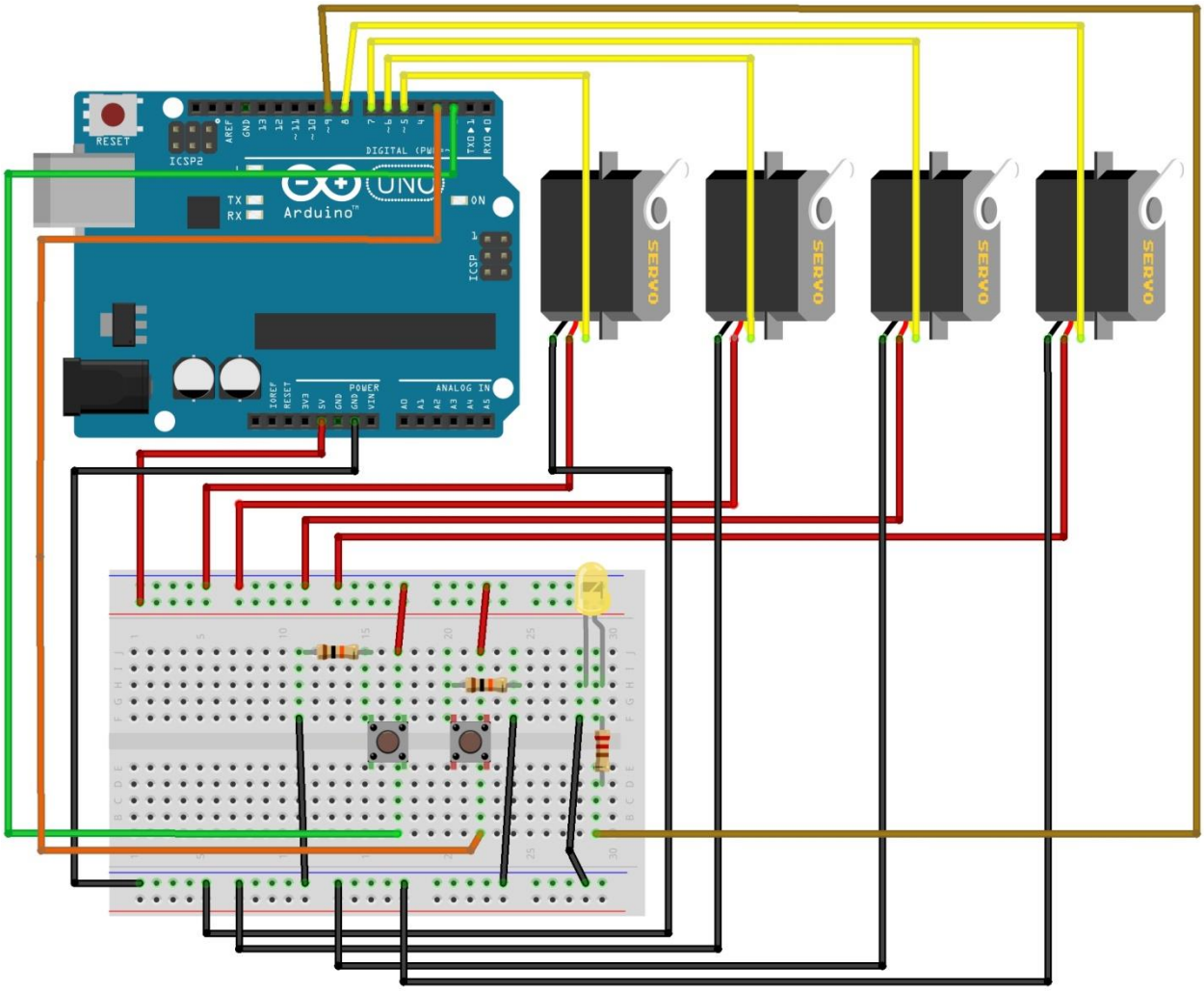
Slika 6.53 Izgled komponenata ruke makete



Slika 6.54 Sastavljanje četvrtog motora

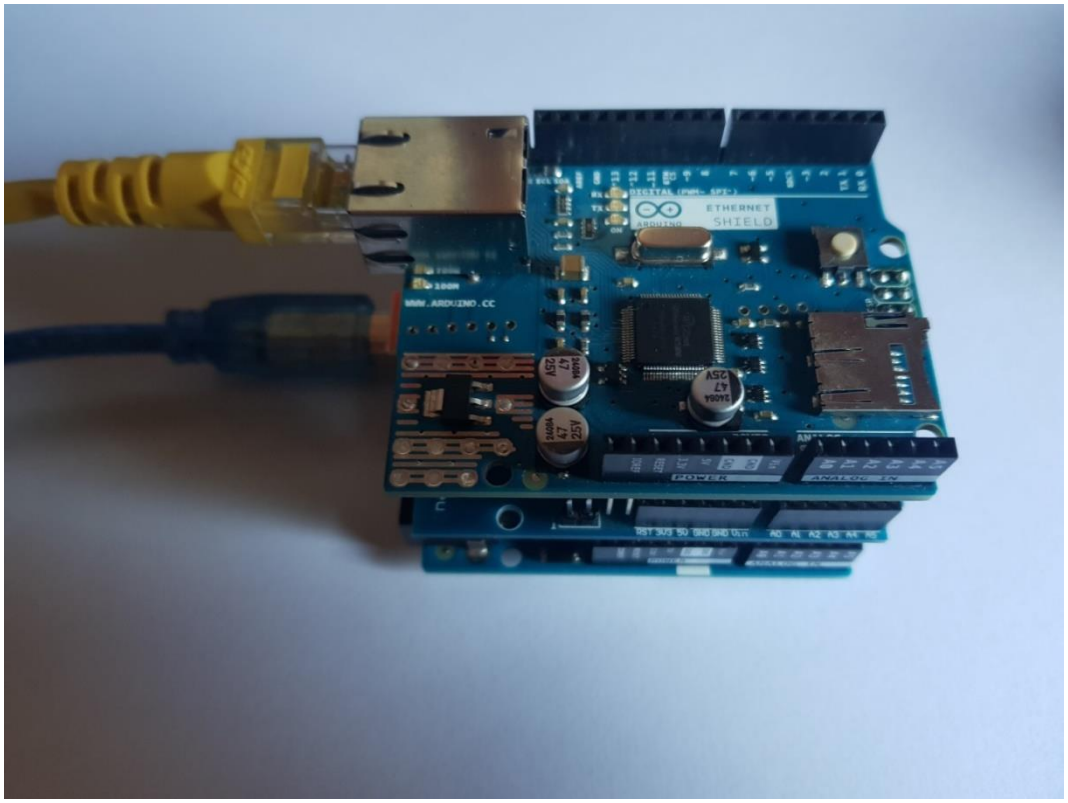
6.6. Spajanje na Arduino

Shema spajanja Arduino razvojne platforme sa servo motorima, tipkama i svjetlećom diodom.

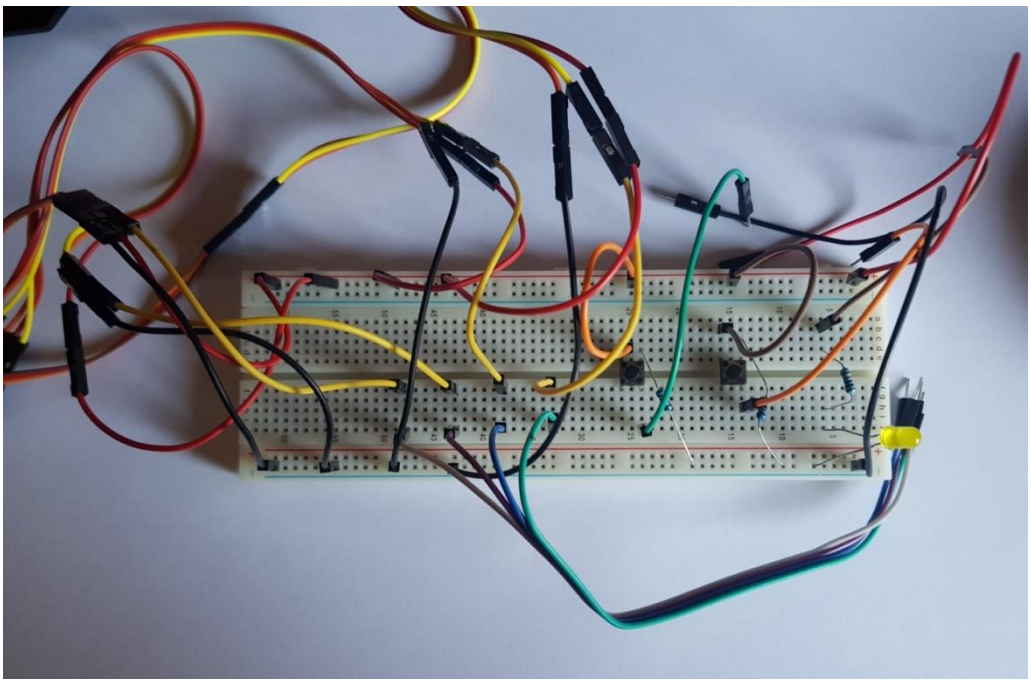


Slika 6.55 Shema spajanja

Slika 6.56 Arduino, Ethernet modul i RTC modul



Slika 6.57 Arduino, Ethernet modul, RTC modul sa kablovima



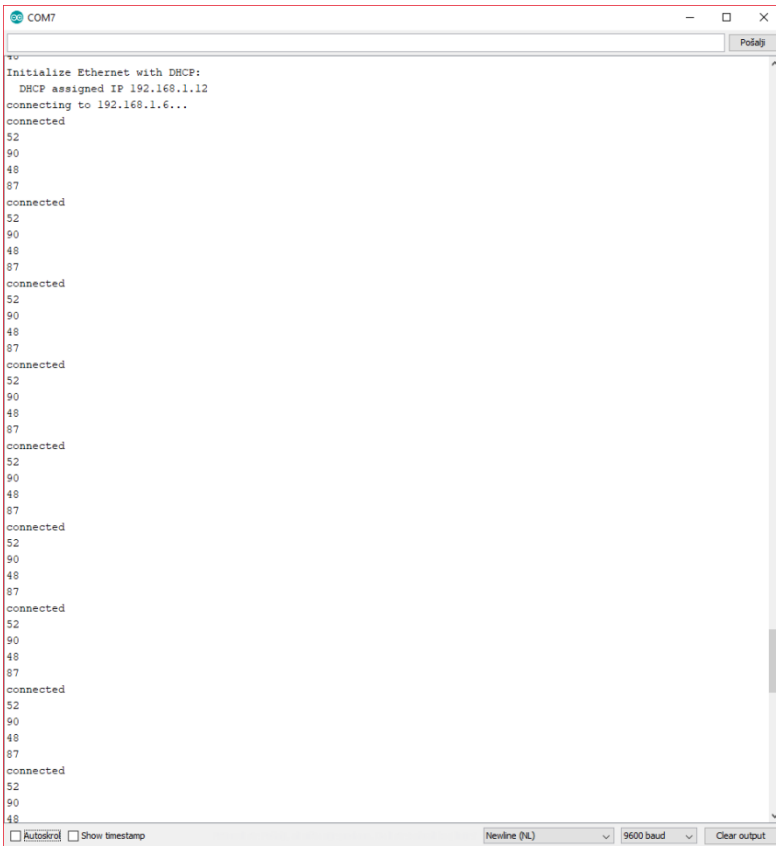
Slika 6.58 Spajanje na eksperimentalnu pločicu

Slika 6.59 Izgled usmjerivača

Ethernet modul se UTP kablom spaja s usmjerivačem. Računalo i mobilni uređaj se na usmjerivač spajaju pomoću bežične mreže.

7. Prikaz vrijednosti

Prilikom promjene vrijednosti na Android aplikaciji ili na web stranici, te vrijednosti se prenose u tekstualne datoteke na poslužitelju. Pomoću Arduino platforme te vrijednost se dohvaćaju i prenose na servo motore. U Serial monitor-u Arduino platforme može se pratiti prijenos tih vrijednosti.



Slika 7.1 Prikaz vrijednosti u Serial monitor-u

8. Zaključak

Zbog jednostavnosti, niske cijene i mnogobrojnoj Internet podršci, Arduino razvojna platforma postaje sve popularnija. Mnoštvo različitih dostupnih knjižica olakšavaju primjenu ove platforme u području automatizacije.

U radu je prikazano kako se korištenjem Arduino Uno razvojne platforme i Ethernet modula može postići komunikacija s poslužiteljem. Vrijednosti se iz poslužitelja mogu primati i slati. Pomoću neke web stranice ili mobilne aplikacije mogu se vrijednosti slati prema poslužitelju te se upisivati u tekstualne datoteke. Pomoću web stranice ili mobilne aplikacije može se kontrolirati stanja na izlazima Arduino razvoje platforme.

Problem koji se pojavljuje je vrijeme kašnjenja. Vrijeme kašnjenja se povećava spajanjem na poslužitelj. U radu je pokazano kako je vrijeme kašnjenja moguće smanjiti korištenjem prekidnih rutina.

Pomoću Arduino razvojne platforme i Ethernet modula moguće je kontrolirati servo motore na način da se s poslužitelja dohvaćaju vrijednosti i prosljeđuju na servo motore. Zbog vremena kašnjenja dolazi do malog zakašnjenja, što znači da ako se promijeni vrijednost u web stranici ili mobilnoj aplikaciji ta vrijednost se ne prenosi na servo motor u realnom vremenu.

9. Literatura

[1] https://learn.sparkfun.com/tutorials/what-is-an-arduino/all , dostupno 23.9.2019.
[2] https://hr.wikipedia.org/wiki/Arduino , dostupno 23.9.2019.
[3] https://en.wikipedia.org/wiki/Arduino , dostupno 23.9.2019.
[4] https://www.arduino.cc/en/products.compare , dostupno 23.9.2019.
[5] https://randomnerdtutorials.com/25-arduino-shields/ , dostupno 23.9.2019.
[6] https://hr.wikipedia.org/wiki/Linux , dostupno 23.9.2019.
[7] https://hr.wikipedia.org/wiki/Android_(operacijski_sustav) , dostupno 23.9.2019.
[8] https://en.wikipedia.org/wiki/XAMPP , dostupno 23.9.2019.
[9] https://www.apachefriends.org/index.html , dostupno 23.9.2019.
[10] https://hr.wikipedia.org/wiki/HTML , dostupno 23.9.2019.
[11] https://www.w3schools.com/html/html_intro.asp , dostupno 23.9.2019.
[12] https://www.w3schools.com/css/default.asp , dostupno 23.9.2019.
[13] https://hr.wikipedia.org/wiki/CSS , dostupno 23.9.2019.
[14] https://en.wikipedia.org/wiki/JavaScript , dostupno 23.9.2019.
[15] https://hr.wikipedia.org/wiki/JavaScript , dostupno 23.9.2019.
[16] https://www.w3schools.com/js/ , dostupno 23.9.2019.
[17] https://en.wikipedia.org/wiki/Ajax_(programming) , dostupno 23.9.2019.
[18] https://www.w3schools.com/xml/ajax_intro.asp , dostupno 23.9.2019.
[19] https://www.w3schools.com/xml/ajax_intro.asp , dostupno 23.9.2019.
[20] https://www.w3schools.com/jquery/ , dostupno 23.9.2019.
[21] https://hr.wikipedia.org/wiki/DHCP , dostupno 23.9.2019.
[22] https://en.wikipedia.org/wiki/Router_(computing) , dostupno 23.9.2019.
[23] https://hr.wikipedia.org/wiki/Usmjerivač , dostupno 23.9.2019.
[24] https://hr.wikipedia.org/wiki/HTTP, dostupno 23.9.2019.
[25] https://hr.wikipedia.org/wiki/PHP, dostupno 23.9.2019.
[26] https://www.w3schools.com/php/, dostupno 23.9.2019.
[27] https://www.velleman.eu/products/view/?id=418862 , dostupno 23.9.2019.
[28] https://hr.wikipedia.org/wiki/IP_broj, dostupno 23.9.2019.
[29] https://en.wikipedia.org/wiki/IP_address, dostupno 23.9.2019.
[30] Matija Buden: Realizacija sustava za komunikaciju sa analognim i digitalnim osjetilima korištenjem Arduino razvojne platforme, Završni rad, Varaždin, 2014.
[31] Michael Margolis : Aruino Cookbook , ožujak 2011
[32] https://stackoverflow.com/questions/9789283/how-to-get-javascript-variable-value-in-php , dostupno 23.9.2019.
[33] https://stackoverflow.com/questions/35406508/get-slider-value-to-php?rq=1 , dostupno 23.9.2019.
[34] https://www.youtube.com/watch?v=rjmtYkRK1nM , dostupno 23.9.2019.
[35] https://www.youtube.com/watch?v=BrpiNUf2XCk , dostupno 23.9.2019.
[36] https://randomnerdtutorials.com/arduino-webserver-with-an-arduino-ethernet-shield/ , dostupno 23.9.2019.
[37] https://www.instructables.com/id/Control-Arduino-using-android-app/ , dostupno 23.9.2019.
[38] https://www.instructables.com/id/Arduino-Android-LED-control-Using-Ethernet-Shield/ , dostupno 23.9.2019..
[39] https://www.instructables.com/id/Arduino-Ethernet-Shield-Tutorial/ , dostupno 23.9.2019.
[40] https://alselectro.wordpress.com/2016/10/30/arduino-ethernet-shield-led-onoff-from-webpage/ , dostupno 23.9.2019.
[41] https://randomnerdtutorials.com/esp32-servo-motor-web-server-arduino-ide/ , dostupno 23.9.2019.
[42] https://www.youtube.com/watch?v=_WqfNyE_pt8 , dostupno 23.9.2019.
[43] https://stackoverflow.com/questions/41512383/communication-between-arduino-and-android-via-ethernet-shield , dostupno 23.9.2019.
[44] https://loopj.com/android-async-http/ , dostupno 23.9.2019.
[45] https://tronixstuff.com/2013/12/06/arduino-tutorials-chapter-16-ethernet/ , dostupno 23.9.2019.
[46] https://forum.arduino.cc/index.php?topic=167395.0 , dostupno 23.9.2019.
[47] https://sindrelindstad.com/projects/how-to-led-arduino-php-proc/, dostupno 23.9.2019.

[48] https://seeeddoc.github.io/Ethernet_Shield_V2.4/ , dostupno 23.9.2019.
[49] https://arduino.stackexchange.com/questions/19986/unable-to-connect-ethernet-shield-w5100-to-localhost , dostupno 23.9.2019.
[50] https://forum.arduino.cc/index.php?topic=326941.0 , dostupno 23.9.2019.
[51] https://circuitdigest.com/microcontroller-projects/arduino-interrupt-tutorial-with-examples , dostupno 23.9.2019.
[52] https://forum.arduino.cc/index.php?topic=186467.0 , dostupno 23.9.2019.
[53] https://www.youtube.com/watch?v=_rXvpZkJrUE , dostupno 23.9.2019.
[54] https://www.arduino.cc/reference/en/language/functions/external-interrupts/attachinterrupt/ , dostupno 23.9.2019.
[55] https://www.arduino.cc/en/reference/ethernet, dostupno 23.9.2019.
[56] https://www.mouser.com/catalog/specsheets/A000056_DATASHEET.pdf , dostupno 23.9.2019.
[57] https://www.velleman.eu/downloads/0/infosheets/datasheet_ka07_uk.pdf , dostupno 23.9.2019.

Popis slika

Slika 2.1 Izgled Arduino LilyPad pločice Izvor:
https://www.sparkfun.com/products/13342
Slika 2.2 Usporedba Arduino Uno, Nano i Mega Izvor:
http://smartmicrocontroller.com/arduino-board-types-comparison-chart/
Slika 2.3 Izgled Arduino Mega, Uno, Nano i Micro pločice Izvor:
https://blog.techiehunter.org/tag/arduino-nano/
Slika 2.4 Izgled Arduino razvojnog okruženja Izvor: Snimka zaslona
Slika 2.5 Prikaz programskih primjera u Arduino razvojnom okruženju Izvor: Snimka zaslona
Slika 2.6 Prikaz relejnog modula za Arduino Izvor:
https://www.amazon.com/Seeed-Arduino-Relay-Shield-V2-0/dp/B00BOZ7V02
Slika 2.7 Izgled Arduino Ethernet modula Izvor:
https://grobotronics.com/arduino-ethernet-shield-rev3-compatible.html?sl=en
Slika 2.8 Izgled Arduino RTC modula Izvor:
https://www.velleman.eu/products/view/?id=418862
Slika 3.1 Izgled Android Studia-a razvojnog okruženja Izvor: Snimka zaslona
Slika 4.1 Primjer HTML programskog koda Izvor:
https://www.w3schools.com/html/default.asp
Slika 4.2 Primjer programskog koda HTML-a s CSS-om Izvor:
https://www.w3schools.com/css/default.asp
Slika 4.3 Primjer HTML programskog koda s Javasript-om Izvor:
https://www.w3schools.com/js/default.asp
Slika 4.4 Prikaz HTML programskog koda s Javascript-om Izvor:
https://www.w3schools.com/js/default.asp
Slika 5.1 Izgled usmjerivača Izvor:
http://fs.airlive.com/manual/AirLive_WL-5460APv2_Manual.pdf

Slika 6.1 HTML programski koda web stranice
Slika 6.2 Početni izgled web stranice
Slika 6.3 Programski kod Javascript funkcije promjene boje slidera
Slika 6.4 Izgled gotove web stranice
Slika 6.5 Programski kod u kojem se poziva PHP datoteka
Slika 6.6 Programski kod PHP datoteke
Slika 6.7 Javascript programski koda za spremanje svih vrijednosti

Slika 6.8 Programski kod za dohvaćanja IP adrese
Slika 6.9 Programski kod dohvaćanja datoteke s poslužitelja
Slika 6.9 Programski kod dohvaćanja datoteke s poslužitelja
Slika 6.10 Programski kod za prosljeđivanje vrijednosti na servo motor
Slika 6.11 Programski kod za prosljeđivanje vrijednosti na servo motor
Slika 6.12 Programski kod za namještanje vremena RTC modula
Slika 6.13 Programski kod pohranjivanja vremena iz RTC modula u varijable
Slika 6.14 Programski kod slanja vremena na poslužitelj
Slika 6.15 PHP datoteka koja pohranjuje vrijeme u tekstualnu datoteku
Slika 6.16 Prekidna rutina svjetleće diode
Slika 6.17 Programski kod slanja stanja svjetleće diode na poslužitelj
Slika 6.18 Prikaz stvaranja novog projekta u Android Studio
Slika 6.19 Prikaz odabira naziva aplikacije u Android Studio.
Slika 6.20 Prikaz dodavanje ikona za aplikaciju
Slika 6.21 Prikaz dodavanja ikona za aplikaciju
Slika 6.22 Izgled početnog zaslona u Android Studio
Slika 6.23 Izgled početnog zaslona na mobilnom uređaju
Slika 6.24 Izgled xml programskog koda unutar Android Studia
Slika 6.25 Izgled početnog zaslona s toast porukom
Slika 6.26 Programski kod slanja vrijednost slidera na poslužitelj
Slika 6.27 Programski kod slanja jedne vrijednosti na poslužitelj
Slika 6.28 Izgled početnog zaslona s upisanom IP adresom
Slika 6.29 Programski kod dohvaćanja vrijednosti s poslužitelja
Slika 6.30 Izgled početnog zaslona nakon dohvaćanja vrijednosti s poslužitelja
Slika 6.31 Dodavanje datoteke u Android Studio
Slika 6.32 Dodavanje datoteke izbornika u Android Studio
Slika 6.33 Xml programski kod izbornika
Slika 6.34 Dodavanje izbornika u Javi
Slika 6.35 Izgled „Alert dialog-a“
Slika 6.36 Dodavanje novog Activity-a
Slika 6.37 Dodavanje naziva novom „Activity-u“
Slika 6.38 Izgled zaslona s žaruljicom
Slika 6.39 Izgled zaslona mobilnog uređaja s žaruljom
Slika 6.40 Programski kod dohvaćanja stanja svjetleće diode
Slika 6.41 Datoteka „colors.xml“

Slika 6.42 Programski kod promjene slike ovisno o stanju svjetleće diode
Slika 6.43 Programski kod izbornika
Slika 6.44 Izgled zaslona s tekstom u Android Studio
Slika 6.45 Mapa „assets“ i datoteka „tekst.html“
Slika 6.46 Izgled zaslona mobilnog uređaja na kojem je ispisan tekst
Slika 6.47 HTML programski kod
Slika 6.47 Sastavljanje podnožja makete
Slika 6.48 Izgled servo motora
Slika 6.49 Sastavljanje podnožja makete s prvim motorom
Slika 6.50 Sastavljanje drugog motora
Slika 6.51 Sastavljanje trećeg motora
Slika 6.52 Sastavljanje triju motora
Slika 6.53 Izgled komponenata ruke makete
Slika 6.54 Sastavljanje četvrtog motora
Slika 6.55 Shema spajanja Izvor:
https://fritzing.org/home/
Slika 6.56 Arduino, Ethernet modul i RTC modul
Slika 6.57 Arduino, Ethernet modul, RTC modul sa kablovima
Slika 6.58 Spajanje na eksperimentalnu pločicu
Slika 6.59 Izgled usmjerivača
Slika 7.1 Prikaz vrijednosti u Serial monitor-u Izvor: Snimka zaslona

UNIVERSITY
NORTH

Sveučilište
Sjever

SVEUČILIŠTE
SJEVER

IZJAVA O AUTORSTVU
I
SUGLASNOST ZA JAVNU OBJAVU

Završni/diplomski rad isključivo je autorsko djelo studenta koji je isti izradio te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, izvora s interneta, i drugih izvora) bez navođenja izvora i autora navedenih radova. Svi dijelovi tuđih radova moraju biti pravilno navedeni i citirani. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom, odnosno nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, Filip Benković *(ime i prezime)* pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog (*obrisati nepotrebno*) rada pod naslovom Upravljanje servo motorima korištenjem Android aplikacije *(upisati naslov)* te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:
(upisati ime i prezime)

FILIP BENKOVIĆ, [signature]
(vlastoručni potpis)

Sukladno Zakonu o znanstvenoj djelatnost i visokom obrazovanju završne/diplomske radove sveučilišta su dužna trajno objaviti na javnoj internetskoj bazi sveučilišne knjižnice u sastavu sveučilišta te kopirati u javnu internetsku bazu završnih/diplomskih radova Nacionalne i sveučilišne knjižnice. Završni radovi istovrsnih umjetničkih studija koji se realiziraju kroz umjetnička ostvarenja objavljuju se na odgovarajući način.

Ja, Filip Benković *(ime i prezime)* neopozivo izjavljujem da sam suglasan/na s javnom objavom završnog/diplomskog (*obrisati nepotrebno*) rada pod naslovom Upravljanje servo motorima korištenjem Android aplikacije *(upisati naslov)* čiji sam autor/ica.

Student/ica:
(upisati ime i prezime)

FILIP BENKOVIĆ, [signature]
(vlastoručni potpis)