INTERNETSKA APLIKACIJA ZA UPRAVLJANJE PAMETNOM KUĆOM

Bešker, Ivo

SVEUČILIŠTE U SPLITU

SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

PRIJEDIPLOMSKI STRUČNI STUDIJ ELEKTRONIKA

Ivo Bešker

ZAVRŠNI RAD

INTERNETSKA APLIKACIJA ZA UPRAVLJANJE PAMETNOM KUĆOM

Split, rujan 2023.

SVEUČILIŠTE U SPLITU

SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

PRIJEDIPLOMSKI STRUČNI STUDIJ ELEKTRONIKA

Predmet: Elektronički sklopovi

Ivo Bešker

ZAVRŠNI RAD

Kandidat: Ivo Bešker
Naslov rada: Internetska aplikacija za upravljanje pametnom kućom
Mentor: Tonko Kovačević

Split, rujan 2023.

Sadržaj

Sažetak 1

Summary 2

1. Uvod 3

2. Uređaji 4

2.1 Raspberry PI 4

2.2 Industrial shields M-Duino 38AR+ 5

2.3 *Hardware-ski* koncept projekta 6

3. Korišteni programi i programski jezici 9

3.1 Apache 2 poslužitelj 9

3.2 Raspbian 9

3.3 PHP 10

3.4 Javascript i jQery 10

3.5 MariaDB 11

3.6 Arduino IDE 11

4. Algoritam pametne kuće 12

5. M-Duino 38AR+ PLC Arduino program 14

5.1 Deklaracija varijabli 14

5.2 Inicijalizacija 18

5.3 Obrada HTTP zahtjeva 22

5.4 Aktiviranje trošila i ažuriranje senzora 26

5.5 Provjera stanja tipkala trošila ili alarma 29

5.6 Obrada naredbi poslanih sa korisničkog sučelja 31

6. Raspberry PI 33

6.1 CRON JOB 33

6.2 MariaDB 37

7. Korisničko sučelje 38

7.1 Uvod 38
7.2 Navigacija 39
7.3 Učitavanje prostorija, trošila i senzora 40
7.4 Slanje naredbe na M-Duino PLC 47
7.5 Osvježavanje stanja trošila i senzora iz baze podataka 48
7.6 PHP skripa za komunikaciju sa bazom podataka 51
8. Zaključak 54
9. Literatura 55
10. Popis slika 56
11. Popis tablica 57

Sažetak

Internetska aplikacija za upravljanje pametnom kućom

Sustavi „pametne kuće“ su već dobrano ukorijenjeni u način izgradnje stambenih jedinica, ali isto tako i u brodogradnji i javnim objektima. Tehnička osnova na kojoj su sustavi pametne kuće izrađeni su se mijenjali sa razvojem tehnologije. Te su evoluirali od jednostavnih industrijskih standarda do, u današnje vrijeme, korištenjem pretežno ethernet mreža, odnosno TCP komunikacije.

U ovom radu će biti predstavljena verzija pametne kuće, koja za temelj ima web sučelje. Kao osnova, koristiti će se TCP[1] komunikacija, odnosno HTTP[2] protokol. Isto tako biti će opisan način komunikacije između raznih uređaja unutar pametne kuće.

Također ću predstaviti specifičnosti na koje se može naići prilikom projektiranja kontrolera za pametne kuće. Koji uređaji se mogu koristiti za izradu kontrolera pametne kuće, koji se programski paketi mogu koristiti, i specifičnosti glede očuvanja korištenih uređaja u smislu zaštite memorija, sigurnosti podataka i mogućih nadogradnji sustava.

Ključne riječi: Internetska aplikacija, HTTP, komunikacija, pametna kuća, web sučelje, projektiranje

Summary

Internet application for smart home management

"Smart home" systems are already well established in the construction of residential units, but also in shipbuilding and public buildings. The technical basis on which smart home systems are built has changed with the development of technology. These have evolved from simple industrial standards to, nowadays, using mostly ethernet networks, and TCP communication.

This paper will present a version of the smart home, which is based on a web interface. As a basis, TCP communication and HTTP protocol, will be used. The method of communication between various devices within the smart home will also be described.

I will also present the specifics that can be encountered when designing controllers for smart homes. Which devices can be used to create a smart home controller, which software packages can be used, and specifics regarding the preservation of used devices in terms of memory protection, data security, and possible system upgrades.

Key words: Internet application, HTTP, communication, smart house, web interface, design

1. Uvod

Za izradu aplikacije za upravljanje pametnom kućom, prvo bi trebalo izabrati hardware-sku osnovu na kojoj će se izvoditi aplikacija. U ovom slučaju, istu aplikaciju će koristiti i moje kolege studenti, stoga sam morao predvidjeti i mogućnost nadogradnje sustava.

Osnovne pretpostavke svih sustava pametne kuće su da korisnik mora biti zadovoljan vizualnim i funkcionalnim dijelom upravljanja pametnom kućom. Podijelio sam *hardware-ski* dio završnog projekta na više korištenih uređaja. Svaki od uređaja ima za svrhu zadovoljiti određeni dio osnovnih postavki. Vizualni dio će odrađivati jedan uređaj, dok će funkcionalni dio, odrađivati drugi uređaj. Također, radi se o internetskoj aplikaciji, koja koristi ethernet mrežu, te je korištena jednostavna mrežna oprema za komunikaciju između uređaja.

Sustavi pametnih kuća su osmišljeni da upravljanje domom dovedu na novu razinu komfora i funkcionalnosti. Komfor se očituje i u vizualnom prikazu upravljačkog sučelja. Izgled upravljačkog sučelja je samo jedan aspekt upravljanja koji se odnosi na dizajn istog. Ali da bi izgled upravljačkog sučelja uopce funkcionirao na način na koji je dizajniran, internetska aplikacija mora bit pomno izrađena da predvidi sve moguće nedostatke, na koje se može naići korištenjem iste.

Programski dio internetske aplikacije se sastoji od operativnog sustava, baze podataka, te programskih skripti. Svi djelovi aplikacije su povezani u jednu cjelinu, koja dalje komunicira sa ostalim uređajima u lokalnoj mreži. Aplikacija nadzire korišenje pametne kuće od strane korisnika, radi reduciranja opterećenja *hardware-a* i općenito mreže. Aplikacija, prikupljanjem informacija sa uređaja na mreži i iz baze podataka, modificira korisničko sučelje, tako da korisnik ima skoro pa trenutnu informaciju o stanju trošila ili senzora instaliranih u sustavu pametne kuće.

2. Uređaji

2.1 Raspberry PI

Osnovno računalo za izradu WEB sučelja je Raspberry PI [3]. Izabrao sam ovaj uređaj jer posjeduje operativni sustav, na kojem je jednostavno instalirati kvalitetan WEB poslužitelj. Na taj način je moguće izraditi aplikaciju koja može vizualno biti prihvatljiva za moderno upravljačko sučelje pametne kuće. Korišten je model Raspberry PI 2 B V1.1. Raspberry PI posjeduje odličnu programsku podršku, jer njegov operativni sustav je izvedenica Linux[4] operativnog sustava. Uređaj posjeduje mrežno sučelje za povezivanje sa ostalim uređajima na mreži.

Video izlaz i USB ulazi daju mogućnost Raspberry PI računalu da se može podesiti direktno, korištenjem monitora, tipkovnice i miša. ROM memorija na računalu je izvedena u obliku Micro SD kartice. Za ovakav tip projekta je dovoljna količina memorije od 32GB, te nije potreban dodatni vanjski hard disk.

Raspberry PI se može koristiti i kao dodatni relejni modul, ili dodatni modul sa analognim ulazima. To može proširiti ulazno izlazne mogućnosti pametne kuće. Takvi ulazi i izlazi se lako mogu kontrolirati pomoću implementiranih naredbi.

Tablica 2.1 - Raspbarry PI karakteristike

Procesor:	A 900MHz quad-core ARM Cortex-A7
Radna memorija:	1GB
Mreža:	100 BASE Ethernet
USB:	4 USB portova
I/O:	40 pinova
Video:	HDMI
Audio:	3,5mm utor
Memorija:	Micro SD

2.2 Industrial shields M-Duino 38AR+

M-Duino 38AR+ je PLC, proizvod tvrtke Industrial shields[5], baziran na Arduino[6] platformi. PLC tog tipa je korišten za upravljanje trošilima i za čitanje podataka sa senzora. Osim programske podrške od strane proizvođača, ovaj PLC ima mogućnost korištenja raznih pogodnosti koje pruža programska podrška od strane Arduino platforme. Na kućištu PLC-a su izvedeni priključci za digitalne ulaze i digitalne izlaze, analogne ulaze i izlaze, te kao na svakom PLC-u, priključci za komunikacijske mreže. Među njima je i RJ-45 konektor za priključak na ethernet mrežu. Što je ključno za izradu ovog tipa sustava pametne kuće, gdje su svi uređaji spojeni u jednu mrežu, zajedno sa korisničkim (upravljačkim) uređajima. Ovaj PLC komunicira preko HTTP protokola sa Raspberry PI uređajem, te zajedno koordiniraju mrežni promet i upravljanje krajnjim uređajima. Na PLC se mogu spojiti i tipkala, kao direktno upravljanje osnovnim elementima pametne kuće. Kao što su, rasvjeta, ventilacija.

M-Duino 38AR+ posjeduje preko 20 I/O ulaza i izlaza, koji se mogu podesiti u razne modove rada. Od komunikacijskih priključaka, osim navedenog mrežnog priključka, PLC sadrži dva TTL UART priključka, SPI priključak, I2C priključak, RS232 i RS485 priključke. Dodatne komunikacijske mreže, omogućavaju proširenje količine upravljivih uređaja, odnosno senzora u sustavu pametne kuće.

Tablica 2.2 – M-Duino 38AR+

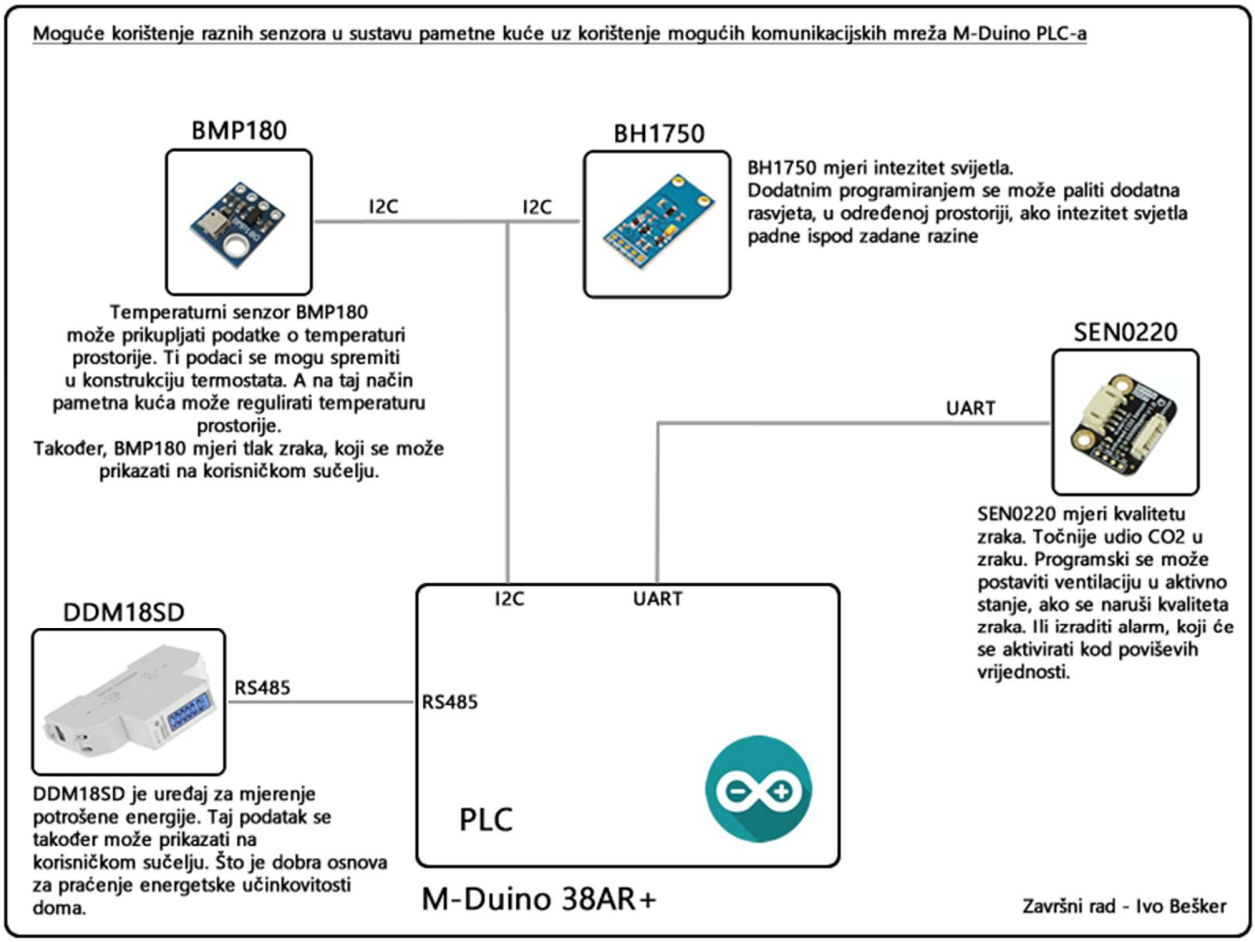
Vcc =	11.4 VDC – 30 VDC
Ic =	700 mA (na 24 VDC)
Pd [uz potrošnju pinova] =	~16.8 W
Radna temperatura =	-20°C – 60°C
Maksimalna frekvencija oscilatora =	16 Mhz

2.3 *Hardware-ski* koncept projekta

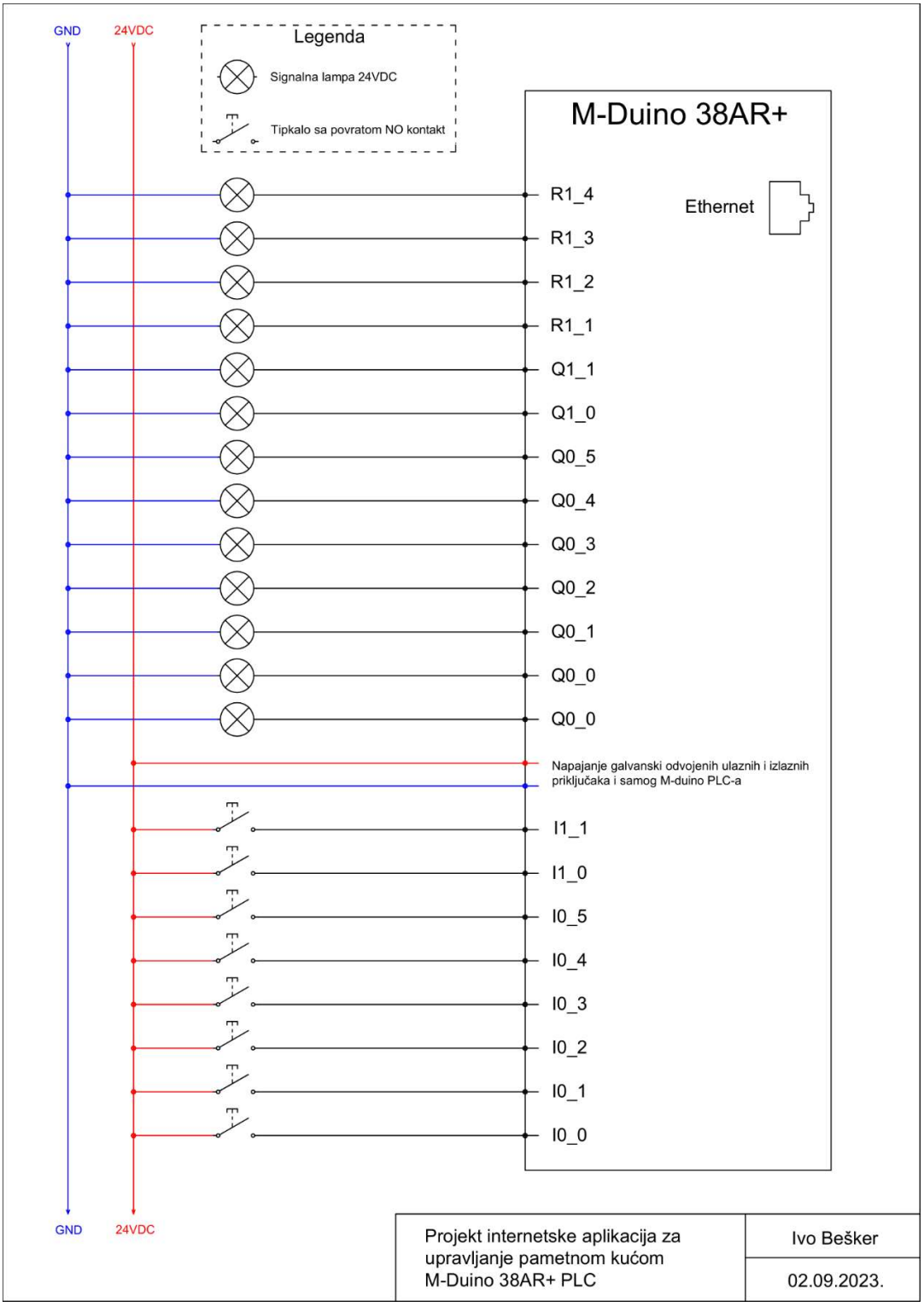
Kontroleri u pametnim kućama su uređaji, koji se u načelu sastoje od samo jdnog uređaja koji nadgleda sve ostale uređaje u sustavu. U ovom radu taj kontroler je izveden sa dva uređaja. Raspberry PI i M-Duino 38AR+ su kombinacija koja predstavlja kontroler pametne kuće. Oba uređaja imaju instaliran WEB poslužitelj, jer se komunikacija odvija preko HTTP protokola. Na WEB poslužitelju na Raspberry PI računalu je glavna WEB aplikacija. Dok se na M-Duino 38AR+ PLC-u HTTP komunikacija koristi za slanje podataka o stanju svih trošila i senzora. Trošilima i senzorima se konstantno provjerava stanje. Da li su aktivirani, ili nisu i da li su se promijenile vrijednosti na senzorima.

Kod pametnih kuća se preporuca da glavni uređaji, kao što je glavni kontroler, imaju statičke mrežne postavke. Tako i u ovoj postavki, Raspberry PI i M-Duino PLC imaju statičke IP adrese. Dok su korisnici spojeni na istu LAN mrežu putem etherneta ili bežičnim putem, ali njihove mrežne postavke su dodijeljene od strane DHCP poslužitelja na mreži. U ovom slučaju to je GSM/GPRS bežični usmjerivač D-LINK DVR-960 sa DHCP poslužiteljem, koji služi za povezivanje kontrolera pametne kuće, modula i korisnika, preko različitih medija prijenosa, te za spajanje na sustav preko interneta. IP adrese Raspberry PI uređaja i M-Duino PLC-a su stavljene na listu fiksnih IP adresa, da ne bi došlo do mrežnih konflikata.

M-Duino PLC zahtjeva 24VDC napajanje. Napajanje M-Duino PLC-a je izvedeno sa Schrack LP 746201-A napajanjem. Raspberry PI zahtjeva napajanje od 5VDC/3A, koje dobiva direktno iz originalnog adaptera.



Slika 2.1 - Mogućnost spajanja senzora u sustavu pametne kuće



Slika 2.2 - Nacrt pokazne ploče upravljanja pametnom kućom

3. Korišteni programi i programski jezici

3.1 Apache 2 poslužitelj

Apache 2 [7] poslužitelj je programski paket predviđen za rad na linux i windows[8] operativnim sustavima. To je HTTP poslužitelj koji ima za svrhu izvršavati glavnu aplikaciju ovog projekta. Apache 2 je najpopularniji poslužitelj otvorenog koda. Sigurnost, efikasnost i nadogradnja su prednosti ovog HTTP servisa.

U ovom projektu Apache 2 poslužitelj je centralni dio kontrolera pametne kuće. Sve skripte vezano za prikaz upravljačkog sučelja pametne kuće se nalaze na tom poslužitelju. Apache 2 poslužitelj omogućuje korištenje određenih programskih jezika, koji služe za izradu skripti koje se izvode na samom poslužitelju. Što je bitno za pristup npr. bazi podataka na sigurniji način. Također, na serveru je podešen naziv servera, koji može olakšati pristup poslužitelju putem WEB preglednika. Umjesto upisivanja IP adrese poslužitelja, potrebno je upisati ime servera, kao npr. u ovom projektu, http://pametnakuca/, te pristupiti upravljačkom sučelju pametne kuće.

3.2 Raspbian

Raspbian[9] je linux operativni sustav, koji je instaliran na Rapsberry PI računalu. To je izvedenica Debian[10] Linux operativnog sustava. Raspbian omogućuje instalaciju niza programa/modula vezano za ovaj projekt. Međuostalim i navedenog Apache 2 poslužitelja. Ovaj projekt koristi razne mogućnosti ovog operativnog sustava, kao npr. „cron job“, koji koristi za periodično pozivanje M-Duino PLC-a i očitanje stanja trošila i senzora.

Također na raspbian operativnom sustavu je moguće podesiti daljinsko upravljanje Raspberry PI uređajem. Tako da prilikom dizajniranja sustava pametne kuće, moguće su jednostavne izmjene na glavnom poslužitelju i ostalim programskim dodacima.

3.3 PHP

PHP[11] je programski jezik u kojem je napisan dio skripti za ovaj projekt. PHP je instaliran na Raspbian operativnom sustavu, te povezan sa Apache 2 poslužiteljem. Na taj način je omogućeno njegovo korištenje na poslužitelju.

Većina skripti vezanih za komunikaciju sa bazom podataka su napisane u PHP-u. PHP je jezik kojim se pišu skripte koje se izvode na poslužitelju, te su sigurnije za pristup bazi podataka, a i ima ugrađene jednostavne procedure za obradu podataka. Kad PHP skripta na poslijetku pribavi podatke iz baze podataka, proslijeđuje iste skriptama koje se izvode na korisničkoj strani (u pregledniku), koji se onda obrađuju i pripremaju za prikazivanje korisniku.

3.4 Javascript i jQery

PHP je programski jezik koji se izvodi na poslužitelju. No za prikaz i dinamičke promjene izgleda WEB sučelja pametne kuće sam koristio javascript[12] programski jezik, kao i javascript biblioteku zvanu jQuery[13]. WEB stranica je napisana u HTML-u[14], odnosno, *HyperText Markup Language.* A određeni dio stranice se prilikom njenog učitavanja ispiše pomoću PHP-a. Sve naknadne promjene, na već učitanoj stranici se, izvode pomoću skripte koja se izvodi na pregledniku korisnika. U ovom slučaju skripte napisane u javascript-u.

Tako npr., promjena stanja lampe, se prikazuje promjenom ikone. Taj dio funkcionalnosti je programiran u javascript-u.

3.5 MariaDB

MariaDB[15] je MySQL[16] poslužitelj koji sam koristio za izradu baze podataka. MariaDB je poslužitelj kojeg je lako podesiti da se može daljinski upravljati, te je lako direktno kontrolirati baze podataka i tablice u bazama. To uvelike pomaže prilikom razvoja ovakvih sustava, gdje se kasnije može istu bazu podataka premjestiti na neki drugi poslužitelj.

MariaDB je poslužitelj koji je otvorenog koda, te ga je jednostavno koristiti i instalirati na Linux platformama, kao što je Raspberry PI. Iako baza podataka nije velika, ali je njen način pohranjivanja podataka bitan za očuvanje glavne memorije računala. U ovom slučaju Raspberry PI računala.

3.6 Arduino IDE

Arduino IDE[17] je programski paket koji sam koristio za kompajliranje programa i programiranje M-Duino PLC-a. Jednostavan je za korištenje i posjeduje veliku programsku podršku. To se odnosi i na programsku podršku za module, koji se mogu koristiti sa arduino platformom i na programsku podršku za pomoćne programe.

Industrial shields M-Duino 38AR+ posjeduje Arduino biblioteke koje pojednostavljuju podešavanje PLC-a. Kao što su, na primjer, I/O priključci, koji se automatski podešavaju kod korištenja.

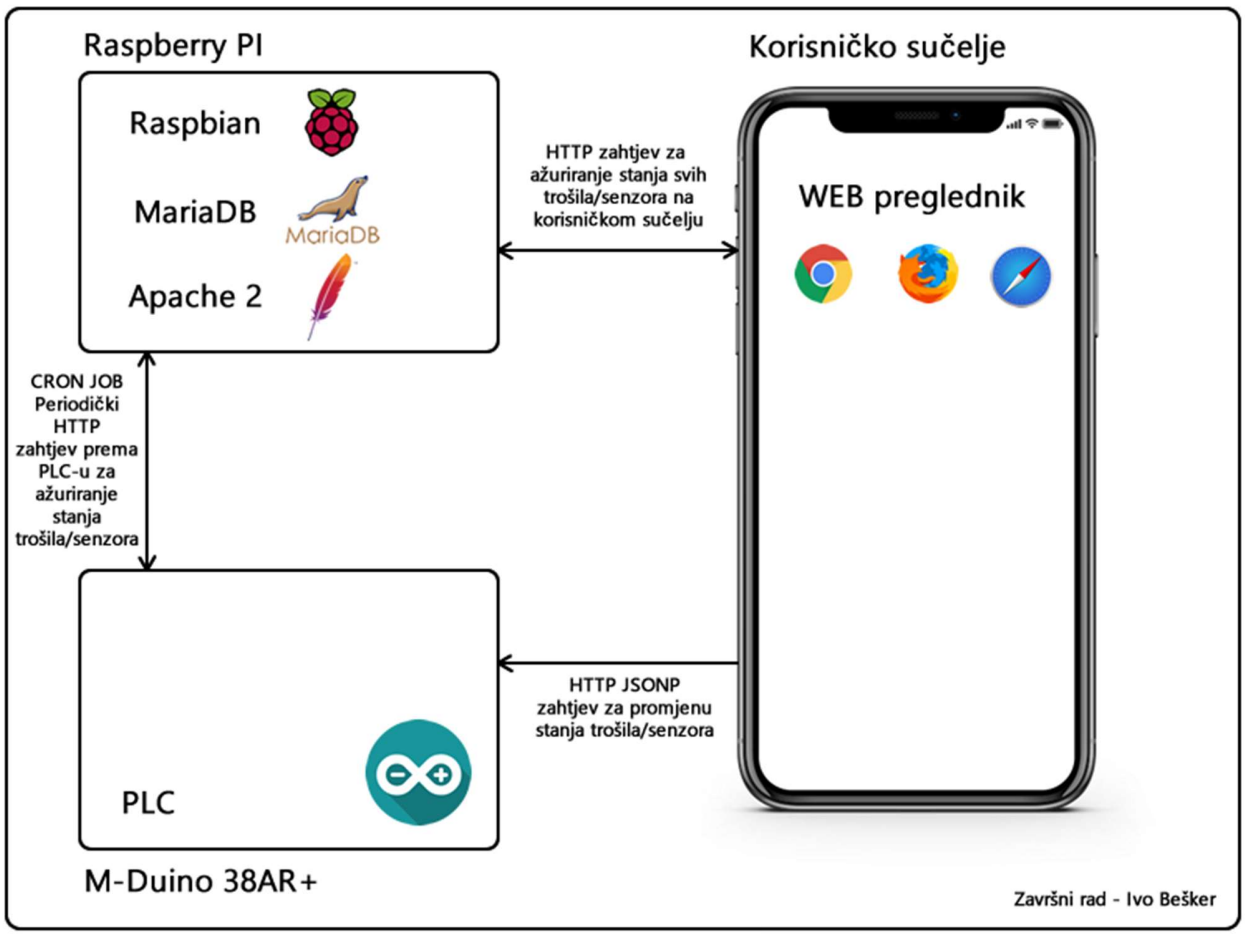
4. Algoritam pametne kuće

Komunikacijski algoritam u projektu je kombinacija tehnologija za komunikaciju. Radi se o komunikaciji između dva WEB poslužitelja. Također se vrši komunikacija između korisničkog pretraživača i M-Duino PLC-a. A sam Raspbian operativni sustav, pomoću periodičkog pozivanja M-Duino PLC-a, obnavlja podatke o stanju trošila i senzora u bazi podataka.

Baza podataka na Raspberry PI računalu služi za sve korisnike pametne kuće, a može ih biti mnogo u jednom trenutku, tako da imaju zajedničku točku za provjeru stanja svih trošila i senzora. I kojoj također mogu brzo pristupiti. Te podatke je prikupio operativni sustav na Rapberry PI računalu. Konstrukcija pametne kuće, odnosno svi programirani katovi, prostorije i trošila/senzori, se nalazi na M-Duino PLC-u. Te podatke također periodički ažurira Raspberry PI u navedenoj bazu podataka. Sva komunikacija se odvija HTTP GET zahtjevom.

Komunikacija između korisničkog sučelja i M-Duino PLC-a se odvija kada korisnik npr. želi upaliti svijetlo. Tada korisničko sučelje direktno šalje PLC-u naredbu ili naredbe, koje PLC obradi, te zapiše u memorijski prostor na PLC-u. Ta komunikacija se odvija između različitih domena, te se koristi HTTP JSONP zahtjev. Zapisani podaci se kasnije u komunikaciji između PLC-a i Raspberry PI računala ažuriraju u bazi podataka na Raspberry PI računalu.

Komunikacija između Raspberry PI računala i korisničkih preglednika se odvija pomoću HTTP GET zahtjeva (WEB poslužitelj - korisnik). Korisničko sučelje periodički zatraži podatke o svim trošilima i senzorima iz aktualizirane baze podataka na Raspberry PI računalu. Korisnički preglednici podatke dobiju u određenom formatu. Podatke obrade, te dinamički, na već učitanoj WEB stranici, izmjenjuju prikazane ikone stanja trošila, ili izmjenjuju prikaz podataka sa senzora.



Slika 4.1 - Grafički prikaz komunikacijskog algoritma pametne kuće

5. M-Duino 38AR+ PLC Arduino program

5.1 Deklaracija varijabli

Kao što je navedeno, ukupno se aplikacija sastoji od više skripti. Na M-Duino PLC-u program je napisan u C-u. Skripte koje se izvode na WEB poslužitelju, na Raspberry PI računalu, su napisane u PHP jeziku. Dok su skripte koje se izvode na korisničkom pregledniku napisane u javascript-u. Sve te skripte komuniciraju međusobno preko HTTP GET zahtjeva.

Arduino program je u načelu jednostavan, jer se može koristiti dosta već napisanog koda. Na početku programa treba uključiti zaglavlje svih dodatnih programa/klasa koje ću koristiti na M-Duino PLC-u. Zatim slijedi definiranje konstanti. Kao npr. broj trošila koje će se koristiti u projektu. Zatim slijedi deklaracija i inicijalizacija varijabli koje su vezane za ethernet vezu. Tu su definirane IP adresa, MAC adresa i ostale postavke mreže.

Zatim slijedi deklaracija varijabli vezanih za projekt pametne kuće. Varijabla *int inicijalizacija* služi kada se M-Duino PLC inicijalno pokrene i kada dobije prvi periodički zahtjev sa Raspberry PI računala, da pošalje konstrukciju pametne kuće Raspberry PI računalu. To znači da PLC šalje programirane katove, prostorije, trošila, senzore i stanje trošila i senzora. Ta varijabla se poništava nakon tog prvog zahtjeva. Svi slijedeći zahtjevi sa Raspberry PI računala prema PLC-u, generiraju samo stanje trošila i senzora.

Slijedeća deklaracija/definicija je deklaracija/definicija polja svih I/O priključaka koji će se koristiti. I/O priključci imaju svoj broj (*index*) koji je u opsegu 0-255. Industrial shields zaglavlje je definiralo njihov, za čovjeka prihvatljiviji zapis, koji se zapisuje u ova polja. Polja će kasnije služiti da se pomoću petlje prolazi kroz sve I/O priključke i utvrdi njihovo stanje, ili promijeni isto. Tu su i polja sa stanjima na ulaznim priključcima PLC-a. Da se kasnije može izvesti programsko tipkalo za fizičku kontrolu trošila spojenih na PLC-u. Sva navedena polja, ovisno da li su namjenjena za ulazne ili izlazne pinove, moraju biti iste dužine.

Na kraju je definirana struktura za sva trošila ili za svaki senzor koji će se koristiti u projektu. Id predstavlja index svakog trošla/senzora. Tip predstavlja vrstu trošila/senzora, tako da korisničko sučelje može izabrati pravilnu vizualnu reprezentaciju istih. Boolena predstavlja stanje na trošilima, ili senzorima koji imaju diskretno stanje. Npr. rasvjeta, alarm, vatrodojava... U polju podatak je zapisan podatak iz senzora. Zapisan je u *int* varijabli, jer se podaci spremaju sa jednom decimalom. Tako da korisničko sučelje ovaj podatak dijeli sa deset (10) i dobije podatak sa jednom decimalom. Funkcije u projektu koje koriste ovaj podatak, trebaju uzeti u obzir i kakav je ovo tip podatka. Podatak_set služi za spremanje podataka vezanih za recimo termostat. Ili slične uređaje, gdje treba postaviti vrijednost koju pametna kuća mora zadovoljiti. Mod podatak služi za spremanje moda rada uređaja. Npr. mod rada termostata može biti off, heat, ili cool. Isto tako ovo polje može služiti za upisivanje naziva mjerne jedinice fizikalne veličine koju senzor mjeri. U kat podatak se upisuje naziv kata na kojem se nalazi prostorija. Ovaj podatak služi za generiranje navigacije na korisničkom sučelju. Prostor je podatak koji također služi za generirenje navigacije na korisničkom sučelju, ali i za izradu panela prostora na korisničkom sučelju. Ime podatak služi za upisivanje naziva trošila, ili kod senzora za upisivanje što se mjeri senzorom. Taj podataka se kasnije spaja sa vrijednošću podatka sa senzora i mjernom jedinicom. Kod služi za upisivanje index-a I/O priključka, na kojem se nalazi trošilo, ili više index-a I/O priključaka spojenih točkom (.), za recimo upavljanje klimom. Ventilokonvektor zahtjeva više relejnih izlaza. Ovaj podatak se može koristiti na više načina, ovisno o funkciji koja ga koristi na PLC-u. U ovo polje se može upisati i riječ koja npr. označava neki vanjski relejni modul, i omogućiti komunikaciju sa istim. Kod_s podatak se koristi za upisivanje index-a ulaznog I/O priključka. To se koristi ako se želi pridružiti tipkalo za paljenje rasvjete. Ili, može se koristiti isto tipkalo za kontrolu više trošila.

Na kraju slijedi deklaracija polja struktura trošila/senzora, ovisno o konstanti BROJ_TROSILA_SENZORA.

```
#include <string.h>
#include <SPI.h>  //SPI komunikacija za MMC karticu i Ethernet vezu
#include <Ethernet.h>
#include <SD.h>
#include <MQ135.h>
#include "MQ7.h"
#include "MHZ19.h"
#include "SdsDustSensor.h"
#include <Wire.h>
#include <BMx280I2C.h>
#include <math.h>

//KONSTANTE
#define BROJ_TROSILA_SENZORA 10 //Ovdje se definira broj trošila/senzora ---
- zatim se svaki senzor u setup() inicijalizira
#define I2C_ADDRESS 0x76 //I2C adresa bma280
#define SD_KARTICA_CS_PIN 53
// size of buffer used to capture HTTP requests
#define REQ_BUF_SZ 128

// MAC address from Ethernet shield sticker under board
byte mac[] = {
   0xDE,
   0xAD,
   0xBE,
   0xEF,
   0xFE,
   0xED
};
IPAddress ip(192, 168, 1, 121); // IP address, may need to change depending on
network
IPAddress gateway(192, 168, 1, 1);
IPAddress myDns(192, 168, 1, 1);
IPAddress subnet(255, 2555, 255, 1);
EthernetServer server(80); // create a server at port 80

char HTTP_req[REQ_BUF_SZ] = {
   0
}; // buffered HTTP request stored as null terminated string
char req_index = 0;

//Inicijalizacija je varijabla koja se resetira nakon prvog slanja strukture tr
ošila/senzora pametne kuće
int inicijalizacija = 1;
```

```
//Definicija digitalnih izlaza
uint8_t digitalni_izlazi[] = {Q0_0,Q0_1,Q0_2,Q0_3,Q0_4,Q0_5,Q0_6,Q0_7,R1_1,R1_2
,R1_3,R1_4,R1_5,R1_6,R1_7,R1_8};

//Definicija digitalnih ulaza
uint8_t digitalni_ulazi[] = {I0_0,I0_1,I0_2,I0_3,I0_4,I1_0,I1_1,I1_2,I1_3,I1_4}
;
uint8_t zadnje_stanje_tipkala[] = {0,0,0,0,0,0,0,0,0,0};
uint8_t stanje[] = {0,0,0,0,0,0,0,0,0,0};

/*
  Generalna struktura za sva trošila
  Ovisno o tipu trošila program će drugačije obraditi podatke
  boolean podatak se upisuje za trošila/senzore sa diskretnim stanjem
  podatak, odnosno podatak_set, se upisuje za trošila/senzore sa numeričkim pod
acima
  mod, kat i prostor su polja znakova koja opisuju lokaciju i ostale podatke o
trošili/senzoru
  mod može služiti i za senzore tipa termostat, gdje se može upisati mod rada i
stog. off/heat/cool
*/
struct trosilo {
   uint8_t id;
   uint8_t tip;
   uint8_t boolean;
   int podatak;
   int podatak_set;
   String mod;
   String kat;
   String prostor;
   String ime;
   String kod;
   String kod_s;
};

struct trosilo trosila[BROJ_TROSILA_SENZORA];
int odgoda_senzora = 0;
```

5.2 Inicijalizacija

U slijedeće segmentu programa se inicijaliziraju trošila i senzori koji će biti u projektu. Sva inicijalizacija se događa unutar setup() funkcije.

Nakon trošila i senzora, slijedi inicijalizacija ethernet modula. Zatim slijedi pokretanje WEB poslužitelja na M-duino PLC-u, zajedno sa provjerom funkcionalnosti MMC memorijske kartice na PLC-u i inicijalizacija serijske komunikacije za prikaz rada programa u hodu i detektiranje grešaka.

```
#include void setup() {
   /*
     Potrebno je upisati podatke za svako trošilo/senzor koliko ih je navedeno
u: BROJ_TROSILA_SENZORA
     Tip trošila/senzora pokrece funkciju za obradu podataka trošila odnosno se
nzora
     Kod i
kod_s označavaju relej ili digitalni/analogni ulaz, koji se kasnije mogu korist
iti za čitanje ili pisanje podataka u I/O
     U polje mod se može upisati mjerna jedinica za tip 4 trošila/senzora, ili
mod rada termostata
     Ako se pod "prostor" upiše "xxx" to trošilo/senzor će se prikazati u svim
prostorijama
     U polje "kod" se sa delimiterima (.) upisuje index tri releja za tri brzin
e i relej ventila ventilokonvektora, kod tipova
3 (termostat), a kod rasvjete samo index izlaznog I/O: uint8_t digitalni_izlazi
     Polje kod se može koristiti za dodavanje specijalnih podataka, koje služe
za slanje naredbi na ostale relejne module u mreži.
Tipovi trošila/senzora:
     1 - svjetlo
     2 - ventilator
     3 - termostat
     4 - senzor opcenito
     5 - alarm
     6 - vatrodojava*/
   trosila[0].id = 1;
   trosila[0].tip = 1;
   trosila[0].boolean = 0;
   trosila[0].podatak = 0;
   trosila[0].podatak_set = 0;
```

```
trosila[0].mod = "as";
trosila[0].kat = "Prizemlje";
trosila[0].prostor = "Dnevni boravak";
trosila[0].ime = "Glavno";
trosila[0].kod = "0"; //Q0_0
trosila[0].kod_s = "0"; //I0_0
//#################################
trosila[1].id = 2;
trosila[1].tip = 2;
trosila[1].boolean = 0;
trosila[1].podatak = 0;
trosila[1].podatak_set = 0;
trosila[1].mod = "au";
trosila[1].kat = "Prizemlje";
trosila[1].prostor = "WC";
trosila[1].ime = "Ventilacija";
trosila[1].kod = "1"; //Q0_1
trosila[1].kod_s = "1"; //I0_1
//#################################
trosila[2].id = 3;
trosila[2].tip = 1;
trosila[2].boolean = 0;
trosila[2].podatak = 0;
trosila[2].podatak_set = 0;
trosila[2].mod = "as";
trosila[2].kat = "Prizemlje";
trosila[2].prostor = "Hodnik";
trosila[2].ime = "Glavno";
trosila[2].kod = "2"; //QQ_2
trosila[2].kod_s = "2"; //I0_2
//#################################
trosila[3].id = 4;
trosila[3].tip = 1;
trosila[3].boolean = 0;
trosila[3].podatak = 0;
trosila[3].podatak_set = 0;
trosila[3].mod = "at";
trosila[3].kat = "Prvi kat";
trosila[3].prostor = "WC";
trosila[3].ime = "Glavno";
trosila[3].kod = "3"; //Q0_3
trosila[3].kod_s = "3"; //I0_3
//#################################
trosila[4].id = 5;
```

```
   trosila[4].tip = 3;
   trosila[4].boolean = 0;
   trosila[4].podatak = 250; //npr 25C *10
   trosila[4].podatak_set = 280;
   trosila[4].mod = "off"; //off, cool, heat
   trosila[4].kat = "Prizemlje";
   trosila[4].prostor = "Dnevni boravak";
   trosila[4].ime = "Temostat";
   trosila[4].kod = "8.9.10.11"; //R1_1, R1_2, R1_3, R1_4 - Kod termostata ide
redoslijed releja za paljenje ventilokonvektora, a obrada se vrši u funkciji za
 paljenje releja
   trosila[4].kod_s = "";
   //#################################
   trosila[5].id = 6;
   trosila[5].tip = 4;
   trosila[5].boolean = 0;
   trosila[5].podatak = 10230;
   trosila[5].podatak_set = 0;
   trosila[5].mod = "hpa";
   trosila[5].kat = "Prizemlje";
   trosila[5].prostor = "xxx";
   trosila[5].ime = "Tlak zraka";
   trosila[5].kod = "";
   trosila[5].kod_s = "";
   //#################################
   trosila[6].id = 7;
   trosila[6].tip = 4;
   trosila[6].boolean = 0;
   trosila[6].podatak = 350;
   trosila[6].podatak_set = 0;
   trosila[6].mod = "%";
   trosila[6].kat = "Prizemlje";
   trosila[6].prostor = "xxx";
   trosila[6].ime = "Vlažnost";
   trosila[6].kod = "";
   trosila[6].kod_s = "";
   //#################################
   trosila[7].id = 8;
   trosila[7].tip = 1;
   trosila[7].boolean = 0;
   trosila[7].podatak = 0;
   trosila[7].podatak_set = 0;
   trosila[7].mod = "ajme";
   trosila[7].kat = "Prizemlje";
```

```
   trosila[7].prostor = "WC";
   trosila[7].ime = "Glavno";
   trosila[7].kod = "4"; //Q0_4
   trosila[7].kod_s = "4"; //I0_4
   //#################################
   trosila[8].id = 9;
   trosila[8].tip = 5;
   trosila[8].boolean = 1;
   trosila[8].podatak = 0;
   trosila[8].podatak_set = 0;
   trosila[8].mod = "%";
   trosila[8].kat = "Prizemlje";
   trosila[8].prostor = "xxx";
   trosila[8].ime = "Alarm";
   trosila[8].kod = "5"; //Q0_5 Relej zujalice, korititi ga mogu svi alarmi
   trosila[8].kod_s = "5"; //I1_0 Alarm zahtjeva manualno alarmiranje
   //#################################
   trosila[9].id = 10;
   trosila[9].tip = 6;
   trosila[9].boolean = 1;
   trosila[9].podatak = 0;
   trosila[9].podatak_set = 0;
   trosila[9].mod = "%";
   trosila[9].kat = "Prizemlje";
   trosila[9].prostor = "xxx";
   trosila[9].ime = "Vatrodojava";
   trosila[9].kod = "5"; //Q0_5 Relej zujalice, korititi ga mogu svi alarmi
   trosila[9].kod_s = "6"; //I1_1 Vanjska pobuda za vatrodojavu
   //#######################Završetak inicijalizacije trošila/senzora

   Ethernet.begin(mac, ip, myDns, gateway, subnet);
   server.begin(); // start to listen for clients

   // initialize SD card
   if (!SD.begin(SD_KARTICA_CS_PIN)) {
      return; // init failed
   }

   //Serial1.begin(9600);
   Serial.begin(9600);
}
```

5.3 Obrada HTTP zahtjeva

Funkcija ethernet_poziv() je funkcija koja obrađuje sve zahtjeve koji dolaze sa mreže prema M-Duino PLC-u. Svaki HTTP zahtjev ima formu u kojoj se nalaze podaci. U primjeru HTTP GET zahtjeva ispod, vidi se podatak „GET /postavke“. WEB poslužitelj mora moći pronaći taj dio zahtjeva i iz riječi „postavke“ zaključiti što treba odgovoriti pošiljatelju. Upravo tome je namjenjena funkcija ethernet_poziv(). Ona prolazi kroz svaki zahtjev, obrađuje ga i donosi odluku što napraviti sa zahtjevom i kakav odgovor će vratiti pošiljatelju. Funkcija ima spremljen zahtjev u varijabli HTTP_req. Podaci u varijabli HTTP_req se obrade u funkciji StrContains(), koja vrati naziv naredbe u zahtjevu. I zatim se ispitivanjem utvrdi što PLC treba uraditi. U ovom projektu, PLC ima nekoliko funkcija koje mora odraditi. Mora moći poslati generičke odgovore pošiljatelju, kao što su da je sve u redu i da nije moguće pronaći naredbu/file. Također PLC mora moći vratiti podatke o konstrukciji pametne kuće (katovi, prostori, trošila i senzori), vratiti podatke o stanju trošila ili senzora, te izmjeniti stanje trošila ili senzora.

```
GET /postavke HTTP/2
Host: 192.168.1.121
User-Agent: curl/7.54.0
Accept: */*
```

```
void etheret_poziv(EthernetClient client) {
   if (client) { // got client?
      boolean currentLineIsBlank = true;
      while (client.connected()) {
         if (client.available()) { // client data available to read
            char c = client.read(); // read 1 byte (character) from client
            // buffer first part of HTTP request in HTTP_req array (string)
            // leave last element in array as 0 to null terminate string (REQ_B
UF_SZ - 1)
            if (req_index < (REQ_BUF_SZ - 1)) {
               HTTP_req[req_index] = c; // save HTTP request character
               req_index++;
            }
            //Serial.print(c);    // print HTTP request character to serial mon
itor
            // last line of client request is blank and ends with \n
```

```
            // respond to client only after last line received
            if (c == '\n' && currentLineIsBlank) {
               // send a standard http response header
               client.println("HTTP/1.1 200 OK");
               client.println("Content-Type: text/html");
               client.println("Connnection: close");
               client.println();

               // open requested web page file

               if (StrContains(HTTP_req, "GET /odgovor")) {
                  if (inicijalizacija == 1) {
                     client.print("255#/#255");
                  } else {
                     digitalWrite(Q1_0, HIGH);
                     for (int i = 0; i < BROJ_TROSILA_SENZORA; i++) {
                        client.print(trosila[i].id);
                        client.print("#");
                        client.print(trosila[i].boolean);
                        client.print("#");
                        client.print(trosila[i].podatak);
                        client.print("#");
                        client.print(trosila[i].podatak_set);
                        client.print("#");
                        client.print(trosila[i].mod);
                        if (i < (BROJ_TROSILA_SENZORA - 1)) {
                           client.print("#/#");
                        }
                     }
                     digitalWrite(Q1_0, LOW);
                  }
               } else if (StrContains(HTTP_req, "GET /postavke")) {
                  digitalWrite(Q1_1, HIGH);
                  for (int i = 0; i < BROJ_TROSILA_SENZORA; i++) {
                     client.print(trosila[i].id);
                     client.print("#");
                     client.print(trosila[i].tip);
                     client.print("#");
                     client.print(trosila[i].boolean);
                     client.print("#");
                     client.print(trosila[i].podatak);
                     client.print("#");
                     client.print(trosila[i].podatak_set);
                     client.print("#");
```

```
                client.print(trosila[i].mod);
                client.print("#");
                client.print(trosila[i].kat);
                client.print("#");
                client.print(trosila[i].prostor);
                client.print("#");
                client.print(trosila[i].ime);
                client.print("#");
                client.print(trosila[i].kod);
                if (i < (BROJ_TROSILA_SENZORA - 1)) {
                   client.print("#/#");
                }
             }
             inicijalizacija = 0;
             digitalWrite(Q1_1, LOW);
          } else if (StrContains(HTTP_req, "GET /naredbe")) {
             //Obrada naredbi
             /* Odgovor - u slučaju da treba - u JSONP formatu
                 client.print("callbackMoivet(");
                 client.print("'");
                 client.print(random(10, 20));
                 client.print("'");
                 client.print(");");
             */
             ObradaNaredbi(HTTP_req);
          } else {
             client.println("HTTP/1.1 404 Not Found");
             client.println("<br/>Stranica nije pronadena!!!");
          }

          // reset buffer index and all buffer elements to 0
          req_index = 0;
          StrClear(HTTP_req, REQ_BUF_SZ);
          break;
       }
       // every line of text received from the client ends with \r\n
       if (c == '\n') {
          // last character on line of received text
          // starting new line with next character read
          currentLineIsBlank = true;
       } else if (c != '\r') {
          // a text character was received from client
          currentLineIsBlank = false;
       }
```

```
         } // end if (client.available())
      } // end while (client.connected())
      delay(1); // give the web browser time to receive the data
      client.stop(); // close the connection
   } // end if (client)
}
// sets every element of str to 0 (clears array)
void StrClear(char * str, char length) {
   for (int i = 0; i < length; i++) {
      str[i] = 0;
   }
}
```

5.4 Aktiviranje trošila i ažuriranje senzora

Funkcija za aktiviranje trošila i ažuriranje senzora koristi podatke koji su spremljeni u strukture trošila i senzora za fizičko aktiviranje digitalnih ili relejnih izlaza, ili obradu podataka sa senzora.

Funkcija prolazi kroz sva inicijalizirana trošila i senzore, i čita trenutno stanje istih u ovisnosti da li imaju diskretnu vrijednost ili podatak. Trošila i senzori, kao što su rasvjeta, ventilatori, alarmi, vatrodojava, imaju zapisanu diskretnu vrijednost. Funkcija čita kod, u kojem je naveden index I/O priključka, na kojem se nalazi trošilo, pretvara ga iz String objekta u int vrijednost. Int vrijednost služi kao index za pronalazak I/O pina u polju pinova. Te ga aktivira ili deaktivira u ovisnosti o stanju koje je zapisano u boolean polju strukture.

Za ostale tipove trošila ili senzora potrebno je napisati dodatni kod za obradu podataka. Ja sam, kao primjer, napisao obradu podataka za termostat. Gdje, program pronalazi sve I/O pinove iz polja kod termostata. Zatim čita u kojem je modu rada termostat. Čita podatak i podatak_set, koji predstavljaju temperaturu na senzoru termostata, odnosno podešenu temperaturu na termostatu. U primjeru, ventilokonvektor, kojim upravlja termostat, ima tri brzine i ventil. Funkcija računa koju brzinu ventilatora treba u određenom modu rada upaliti, te da li je potrebno upaliti ventil. I na taj način se kontrolira temperatura u prostoriji u kojoj je definiran termostat.

```
void PaljenjeAktuatoraSenzora() {
   char nar, nar1, nar2, nar3, nar4;
   char pod, pod1, pod2, pod3;
   uint8_t buffer_trosila; //releji lampe, ventilatori, alarmi, vatrodojave
   uint8_t buffer_s1, buffer_s2, buffer_s3, buffer_v; //releji klima

   for (int i = 0; i < BROJ_TROSILA_SENZORA; i++) {

      if ((trosila[i].tip <= 2) || (trosila[i].tip == 5) || (trosila[i].tip ==
6)) {
         buffer_trosila = trosila[i].kod.toInt();
         if (trosila[i].boolean == 1) {
            digitalWrite(digitalni_izlazi[buffer_trosila], HIGH);
         } else {
            digitalWrite(digitalni_izlazi[buffer_trosila], LOW);
```

```
        }

    } else if (trosila[i].tip == 3) {

        nar = trosila[i].kod.indexOf(".");
        nar1 = trosila[i].kod.indexOf(".", nar + 1);
        nar2 = trosila[i].kod.indexOf(".", nar1 + 1);
        buffer_s1 = trosila[i].kod.substring(0, nar).toInt();
        buffer_s2 = trosila[i].kod.substring(nar + 1, nar1).toInt();
        buffer_s3 = trosila[i].kod.substring(nar1 + 1, nar2).toInt();
        buffer_v = trosila[i].kod.substring(nar2 + 1).toInt();

        //Hlađenje
        if (trosila[i].mod == "cool") {

            if (trosila[i].podatak_set >= trosila[i].podatak) {
                digitalWrite(digitalni_izlazi[buffer_v], LOW);
                digitalWrite(digitalni_izlazi[buffer_s1], LOW);
                digitalWrite(digitalni_izlazi[buffer_s2], LOW);
                digitalWrite(digitalni_izlazi[buffer_s3], LOW);
            }
            if (trosila[i].podatak_set < trosila[i].podatak) {
                if (trosila[i].podatak_set >= (trosila[i].podatak - 10)) {
                    digitalWrite(digitalni_izlazi[buffer_v], HIGH);
                    digitalWrite(digitalni_izlazi[buffer_s2], LOW);
                    digitalWrite(digitalni_izlazi[buffer_s3], LOW);
                    digitalWrite(digitalni_izlazi[buffer_s1], HIGH);
                }
            }
            if (trosila[i].podatak_set < (trosila[i].podatak - 10)) {
                if (trosila[i].podatak_set >= (trosila[i].podatak - 20)) {
                    digitalWrite(digitalni_izlazi[buffer_v], HIGH);
                    digitalWrite(digitalni_izlazi[buffer_s1], LOW);
                    digitalWrite(digitalni_izlazi[buffer_s3], LOW);
                    digitalWrite(digitalni_izlazi[buffer_s2], HIGH);
                }
            }
            if (trosila[i].podatak_set < (trosila[i].podatak - 20)) {
                digitalWrite(digitalni_izlazi[buffer_v], HIGH);
                digitalWrite(digitalni_izlazi[buffer_s1], LOW);
                digitalWrite(digitalni_izlazi[buffer_s2], LOW);
                digitalWrite(digitalni_izlazi[buffer_s3], HIGH);
            }
        }
```

```
        //Grijanje
        if (trosila[i].mod == "heat") {

           if (trosila[i].podatak_set <= trosila[i].podatak) {
              digitalWrite(digitalni_izlazi[buffer_v], LOW);
              digitalWrite(digitalni_izlazi[buffer_s1], LOW);
              digitalWrite(digitalni_izlazi[buffer_s2], LOW);
              digitalWrite(digitalni_izlazi[buffer_s3], LOW);
           }
           if (trosila[i].podatak_set > trosila[i].podatak) {
              if (trosila[i].podatak_set <= (trosila[i].podatak + 10)) {
                 digitalWrite(digitalni_izlazi[buffer_v], HIGH);
                 digitalWrite(digitalni_izlazi[buffer_s2], LOW);
                 digitalWrite(digitalni_izlazi[buffer_s3], LOW);
                 digitalWrite(digitalni_izlazi[buffer_s1], HIGH);
              }
           }
           if (trosila[i].podatak_set > (trosila[i].podatak + 10)) {
              if (trosila[i].podatak_set <= (trosila[i].podatak + 20)) {
                 digitalWrite(digitalni_izlazi[buffer_v], HIGH);
                 digitalWrite(digitalni_izlazi[buffer_s1], LOW);
                 digitalWrite(digitalni_izlazi[buffer_s3], LOW);
                 digitalWrite(digitalni_izlazi[buffer_s2], HIGH);
              }
           }
           if (trosila[i].podatak_set > (trosila[i].podatak + 20)) {
              digitalWrite(digitalni_izlazi[buffer_v], HIGH);
              digitalWrite(digitalni_izlazi[buffer_s1], LOW);
              digitalWrite(digitalni_izlazi[buffer_s2], LOW);
              digitalWrite(digitalni_izlazi[buffer_s3], HIGH);
           }
        }
        //Gašenje
        if (trosila[i].mod == "off") {
           digitalWrite(digitalni_izlazi[buffer_v], LOW);
           digitalWrite(digitalni_izlazi[buffer_s1], LOW);
           digitalWrite(digitalni_izlazi[buffer_s2], LOW);
           digitalWrite(digitalni_izlazi[buffer_s3], LOW);
        } } } }
```

5.5 Provjera stanja tipkala trošila ili alarma

Sva trošila se osim kontrole putem aplikacije, mogu kontrolirati i putem tipkala. To je jako bitna funkcija kod sustava pametnih kuća, jer sustav može zakazati i korisnik mora imati opciju kako kontrolirati barem osnovna trošila u kućanstvu. Ali, također i iz praktičnosti.

Funkcija ProvjeraTipkala() čita stanje na svim inicijaliziranim ulaznim I/O priključcima. Zatim funkcija, za svaki ulaz koji je pročitala, prolazi sva trošila koja koriste taj ulaz i mijenja stanje trošila.

Funkcija ProvjeraTipkala() se konstantno poziva unutar main() funkcije, te je potrebno programirati zaštitu da se promjena stanja trošila ne bi bespotrebno izvršavala. Zato je implementirano da se promjena stanja trošila odvija samo na padajućem dijelu impulsa ulaza. Prijelaz iz 1 u 0.

```
void ProvjeraTipkala() {
   String buffer;

   for (int i = 0; i < sizeof(digitalni_ulazi); i++) { //digitalni ulaz može sl
užiti za više trošila/senzora

      buffer = String(i);
      stanje[i] = digitalRead(digitalni_ulazi[i]);

      if (stanje[i] != zadnje_stanje_tipkala[i]) {

         zadnje_stanje_tipkala[i] = stanje[i];

         if (stanje[i] == 0) {
            for (int j = 0; j < BROJ_TROSILA_SENZORA; j++) {

               if (trosila[j].kod_s == buffer) {

                  Serial.print(trosila[j].boolean);
                  if (trosila[j].boolean == 1) {
                     trosila[j].boolean = 0;
                  } else {
                     trosila[j].boolean = 1;
```

```
                    }

                }
            }
        }
    }
}
```

5.6 Obrada naredbi poslanih sa korisničkog sučelja

Korisnička sučelja direktno šalju M-Duino PLC-u naredbe za aktiviranje trošila ili podešavanje senzora. Kada funkcija ethernet_poziv() otkrije zahtjev sa korisničkog sučelja, ona poziva funkciju ObradaNaredbi().

Ova funkcija je specifična zbog toga što može obraditi jedan ili više zahtjeva, koji su spojeni u URL zahtjev. Sustav pametne kuće je napravljen na način da korisnik ima najbrži mogući odziv na korisničkom sučelju. Tako ako korisnik npr. brzo aktivira uređaje, sustav neće slati svaku naredbu zasebno. Nego će naredbe spojiti u jedan zahtjev, te će tako spojeni zahtjevi funkcija ObradaNaredbi() obraditi i zapisati podatke u strukture trošila. Ukupno je predviđeno da korisnik ne može poslati više od tri naredbe odjednom, zbog ugrađenog vremenskog okvira na korisničkom sučelju.

Funkcija razlaže tekst, koji je izdvojen iz URL-a naredbe. Pronalazi specijalne znakove, koji obavijaju korisne podatke. To je napravljeno na način da je pronađen *index* od svakog specijalnog zanka. Ti *index-i* tada služe za određivanje granica unutar teksta gdje se nalaze korisni podaci. Podaci se tada zapisuju u strukture trošila i senzora.

```
void ObradaNaredbi(char * str_req) {
   String str(str_req);
   uint8_t id, boolean, index_strukture;
   int podatak_set;
   String mod, naredbe[3];

   char nar, nar1, nar2, nar3, nar4;
   char pod, pod1, pod2, pod3;

   nar2 = -1, nar3 = -1;
   nar4 = -1;

   nar = str.indexOf("?", 0);
   nar1 = str.indexOf("?", nar + 1);
   if (str.indexOf("?", nar1 + 1) != -1) {
      nar2 = str.indexOf("?", nar1 + 1);
      if (str.indexOf("?", nar2 + 1)) {
```

```
         nar3 = str.indexOf("?", nar2 + 1);
      }
   }

   naredbe[0] = str.substring(nar + 1, nar1);
   if (nar2 != -1) {
      naredbe[1] = str.substring(nar1 + 1, nar2);
   } else {
      naredbe[1] = "0";
   }
   if (nar3 != -1) {
      naredbe[2] = str.substring(nar2 + 1, nar3);
   } else {
      naredbe[2] = "0";
   }

   for (int i = 0; i < 3; i++) {
      if (naredbe[i] != "0")
         for (int j = 0; j < 4; j++) {
            pod = naredbe[i].indexOf("!");
            pod1 = naredbe[i].indexOf("!", pod + 1);
            pod2 = naredbe[i].indexOf("!", pod1 + 1);

            index_strukture = naredbe[i].substring(0, pod).toInt() - 1;
            trosila[index_strukture].boolean = naredbe[i].substring(pod + 1, po
d1).toInt();
            trosila[index_strukture].podatak_set = naredbe[i].substring(pod1 +
1, pod2).toInt();
            trosila[index_strukture].mod = naredbe[i].substring(pod2 + 1);
         }
   }

}
```

6. Raspberry PI

6.1 CRON JOB

Raspberry PI kao računalo ima instaliran operacijski sustav Raspbian, koji je inačica Linux operativnog sustava Debian. Linux operativni sustavi imaju opciju servisa CRON JOB. To je servis koji se pokreće sa operativnim sustavom. Njegova zadaća je da periodički obavlja zadane radnje na Linux platformi. Ja sam koristio ovaj servis da mogu periodički pozvati PHP skriptu, koja dobavlja podatke sa M-Duino PLC-a o statusu svih trošila i senzora. I prikupljene podatke sprema u lokalnu bazu podataka. Ovo je bitan proces, jer se tako rasterećuje PLC od dodatnih zahtjeva od strane korisničkih sučelja. Ako bi više korisničkih sučelja slalo zahtjeve za obnovu stanja trošila i senzora i usput slalo naredbe, M-Duino PLC ne bi mogao vremenski obraditi toliko HTTP zahtjeva. A i HTTP zahtjevi bi dolazili u nepredviđeno vrijeme, što bi povečalo zagušenje. Stoga lokalna baza podataka služi za osvježavanje stanja trošila i senzora na korisničkim sučeljima, dok se same naredbe šalju direktno na PLC.

Osvježavanje stanja trošila i senzora ne treba bit tako učestalo. I kod modernih sustava pametnih kuća, naredba koja se ne obradi, će tek nakon nekoliko sekundi vratiti prvobitno stanje trošila na sučelju. Soga je CRON JOB podešen na slanje zahtjeva prema M-DUINO PLC-u svakih 1,5 sekundi.

CRON JOB se u Linux operativnom sustavu podešava na način da se pozove terminal i upiše naredba: *sudo crontab -e*

U tako otvoreni file za podešavanje cron job servisa se upisuje vrijeme kada će se izvršiti skripta i lokacija skripte. Linux CRON JOB podržava maksimalni razmak između pozivanja skripti od jedne minute. Tako da treba 40 puta pozvati skriptu *cron_job_status_update.php* na način da se prilikom svakog poziva upiše sleep opcija za svaki poziv. Tako se dobije željena duljina poziva od 1,5 sekunde:

```
* * * * * (php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 1.5; php /var/www/html/cron_job_status_update.php)
```

```
* * * * * (sleep 3; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 4.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 6; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 7.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 9; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 10.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 12; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 13.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 15; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 16.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 18; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 19.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 21; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 22.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 24; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 25.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 27; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 28.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 30; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 31.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 33; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 34.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 36; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 37.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 39; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 40.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 42; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 43.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 45; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 46.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 48; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 49.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 51; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 52.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 54; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 55.5; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 57; php /var/www/html/cron_job_status_update.php)
* * * * * (sleep 58.5; php /var/www/html/cron_job_status_update.php)
```

PHP skripta *cron_job_status_update.php* koja se periodički poziva, koristi mogućnost PHP jezika da komunicira sa lokalnom bazom podataka. Skripta je podijeljena u dva dijela. Prvi dio se poziva kada M-Duino PLC odgovori ovoj skripti da se tek pokrenuo, odnosno da ova skripta

osvježi konstrukciju (katovi, prostorije, trošila, senzori) pametne kuće u bazi podataka na Raspberry PI računalu. Kada M-Duino vrati poruku „255#/#255“, skripta će znati da treba obnoviti MySQL bazu podataka na Raspberry PI računalu, te će ponovno poslati zahtjev M-Duino PLC-u da vrati konstrukciju. Primljeni podaci se obrađuju i spremaju u bazu podataka. Korisnička sučelja, prilikom učitavanja, pribave podatke o kontrukciji.

U slučaju da PLC zaprimi zahtjev „GET /odgovor“, PLC vrati podatke o stanju svih trošila i senzora, a skripta *cron_job_status_update.php* osvježi te podatke u bazi podataka.

```
<?php
   $citanje_podataka = file_get_contents("http://192.168.0.121/odgovor");

   $servername = "localhost";
   $username = "admin";
   $password = "root";
   $dbname = "pametnakuca";
   $conn = new mysqli($servername, $username, $password, $dbname);

   if ($citanje_podataka == "255#/#255") {

       $citanje_podataka = file_get_contents("http://192.168.0.121/postavke");

       $GET_podaci = explode('#/#', $citanje_podataka);

       $DELETE_sql = "DELETE FROM `trosila`";
       $result = $conn->query($DELETE_sql);

       $INSERT_sql_start = "INSERT INTO `trosila` (`id`,`tip`, `bool`, `podatak`,
`podatak_set`, `mod`, `kat`, `prostor`, `ime`, `kod`) VALUES
   ";
       $INSERT_sql = "";

       for ($i = 0; $i < count($GET_podaci); $i++) {

           $unos = explode('#', $GET_podaci[$i]);

           if ($i != (count($GET_podaci) - 1)) {
               $INSERT_sql .= "('" . $unos[0] . "', '" . $unos[1] . "', '" . $unos[2] . "',
'" . $unos[3] . "', '" . $unos[4] . "', '" . $unos[5] . "', '" . $unos[6] . "', '" .
$unos[7] . "', '" . $unos[8] . "', '" . $unos[9] . "'),
```

```
                ";
            } else {
                $INSERT_sql .= "('" . $unos[0] . "', '" . $unos[1] . "', '" . $unos[2] . "',
'" . $unos[3] . "', '" . $unos[4] . "', '" . $unos[5] . "', '" . $unos[6] . "', '" .
$unos[7] . "', '" . $unos[8] . "', '" . $unos[9] . "');";
            }
        }
        $result = $conn->query($INSERT_sql_start . $INSERT_sql);
    } else {
        if ($citanje_podataka[0] == '') {
            exit();
        }

        $GET_podaci = explode('#/#', $citanje_podataka);

        for ($i = 0; $i < count($GET_podaci); $i++) {

            $unos = explode('#', $GET_podaci[$i]);

            $result = $conn->query("UPDATE `trosila` SET `bool`='" . $unos[1] . "',
`podatak`='" . $unos[2] . "', `podatak_set`='" . $unos[3] . "', `mod`='" . $unos[4] . "'
WHERE `id`='" . $unos[0] . "';");
        }
    }
    $conn->close();
    ?>
```

6.2 MariaDB

MariaDB je poslužitelj MySQL baze podataka, koju koristi sustav pametne kuće. Baza podataka je izrađena prilikom dizajniranja projekta, te se u toj istoj bazi podataka izradi tablica sa određenom strukturom, u koju će se upisivati podaci primljeni da M-Duino PLC-a.

U ovom slučaju baza podataka se zove pametnakuca, a tablica trosila. Izrađena tablica je specifična jer ima *heap (memory) engine*. To znači da se podaci u tablici zapravo nalaze u RAM memoriji. Odnosno, da se podaci ne zapisuju na MMC karticu operativnog sustava. Kada se Raspberry PI računalo izgasi, svi podaci iz te tablice se izgube. Osvježavanje podataka se događa često, te bi prilikom čestog zapisivanja podataka na MMC karticu, moglo doći do njenog oštećenja nakon nekog vremena. Stoga je bolje zapisivati podatke u brzu RAM memoriju, jer podaci nisu potrebni nakon gašenja „kontrolera“. MySQL kod za izradu baze podataka sa pripadajućom tablicom se nalazi u nastavku.

```
/* Izrada baze podataka pametnakuca */
CREATE DATABASE `pametnakuca`;

/* Naredba za izrada tablice trosila u bazi podataka pametna kuca */
DROP TABLE IF EXISTS `trosila`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `trosila` (
  `id` tinyint(255) NOT NULL AUTO_INCREMENT,
  `tip` tinyint(255) DEFAULT NULL,
  `bool` tinyint(255) DEFAULT NULL,
  `podatak` int(11) DEFAULT NULL,
  `podatak_set` int(11) DEFAULT NULL,
  `mod` varchar(255) DEFAULT NULL,
  `kat` varchar(255) DEFAULT NULL,
  `prostor` varchar(255) DEFAULT NULL,
  `ime` varchar(255) DEFAULT NULL,
  `kod` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MEMORY AUTO_INCREMENT=11 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

7. Korisničko sučelje

7.1 Uvod

Korisničko sučelje je zapravo WEB stranica koja se nalazi na Apache 2 WEB poslužitelju. Izgled stranice ovisi o korištenim alatima za izradu stranice. Za ovu priliku sam koristio Bootstrap[18] razvojni okvir, koji koristi HTML i CSS[19] (*Cascading Style Sheets*) prezentacijske jezike za vizualno uređivanje html stranica i javascipt odnosno jQuery jezike. Pomoću Bootstrap razvojnog okvira sam izradio stranicu koja ima značajke da se bez većih problema može prikazivati na uređajima sa različitim veličinama ekrana. Da bi stranica bila što jednostavnija za korištenje, ona ima prazan centralni dio koji služi za prikaz ikona trošila i senzora, dok je iznad centralnog dijela prikazana navigacija sa padajućim izbornikom, pomoću kojeg se označava prostorija kojom se želi upravljati. Pozadinu zauzima slika velikog formata koja je prilagođena za prikazivanje na ekranima različite veličine.

Slika 7.1 - Korisničko sučelje

7.2 Navigacija

Navigacija se, kako je spomenuto, nalazi pri vrhu stranice. Prilikom inicijalnog učitavanja glavne stranice, PHP skripta koja se nalazi u datoteci index.php (glavna stranica), učitava iz baze podataka, konstrukciju navigacije unutar <ul> i <li> tag-ova html-a. Ti tagovi su također unaprijeđeni CSS-om i javascript skriptom da dobiju funkciju padajućeg izbornika.

```
<!-- ############### NAVIGACIJA -->
<!-- ############### NAVIGACIJA -->
<?PHP
   $servername = "localhost";
   $username = "admin";
   $password = "root";
   $dbname = "pametnakuca";
   $conn = new mysqli($servername, $username, $password, $dbname);

   $P_konstrukcija = [];

   $query_katovi = "SELECT DISTINCT kat FROM `trosila`";
   $query_prostorije = "SELECT DISTINCT `prostor` FROM `trosila`";
   $query_trosila = "SELECT `id`, `tip`, `bool`, `podatak`, `podatak_set`, `mod`, `ime`,
`kod` FROM `trosila`";

   $result = $conn->query($query_katovi);
   if (!$result) {
       echo "GRESKA";
       exit();
   } else {
       $z = 0;
       while ($row = mysqli_fetch_row($result)) {

           echo '<li class="cn-dropdown-item has-down">
           <a href="#">' . $row[0] . '</a>
           <ul class="dropdown">
           '; //style="display: block;"

           $result_prostorije = $conn->query($query_prostorije . " WHERE `kat`='" . $row[0]
. "'");
           while ($row_prostorije = mysqli_fetch_row($result_prostorije)) {
               if ($row_prostorije[0] != 'xxx') {
                   echo '<li><a onclick="prostor(' . $z . ')">' . $row_prostorije[0] .
'</a></li>';
```

```
                }
                $result_trosila = $conn->query($query_trosila . " WHERE `prostor`='" .
$row_prostorije[0] . "' AND `kat`='" . $row[0] . "' ORDER BY `tip`");
                $i = 0;
                while ($row_trosila = mysqli_fetch_row($result_trosila)) {
                    $P_konstrukcija[$row[0]][$row_prostorije[0]][$i][0] = $row_trosila[0];
                    $P_konstrukcija[$row[0]][$row_prostorije[0]][$i][1] = $row_trosila[1];
                    $P_konstrukcija[$row[0]][$row_prostorije[0]][$i][2] = $row_trosila[2];
                    $P_konstrukcija[$row[0]][$row_prostorije[0]][$i][3] = $row_trosila[3];
                    $P_konstrukcija[$row[0]][$row_prostorije[0]][$i][4] = $row_trosila[4];
                    $P_konstrukcija[$row[0]][$row_prostorije[0]][$i][5] = $row_trosila[5];
                    $P_konstrukcija[$row[0]][$row_prostorije[0]][$i][6] = $row_trosila[6];
                    $P_konstrukcija[$row[0]][$row_prostorije[0]][$i][7] = $row_trosila[7];
                    $i++;
                }
                if ($row_prostorije[0] != 'xxx') {
                    $z++;
                }
            }
            echo '
                </ul>
                </li>
                ';
        }
    }
    echo '<script> var stara_konstrukcija = ' . json_encode($P_konstrukcija) . '; </script>';
    ?>
<!-- ############### NAVIGACIJA -->
<!-- ############### NAVIGACIJA -->
```

7.3 Učitavanje prostorija, trošila i senzora

Centralni dio stranice sadržava sve prostorije koje se učitavaju prilikom inicijalnog učitavanja stranice. Stranice se ispisuju pomoću PHP skripte, te su automatski sakrivene u pozadini stranice (style="display:none"). Javascript, nakon završetka učitavanja stranice, prikaže prostoriju koja je navedena u javascript skripti pomoću ID-ja prostorije (Određuje programer). Ako želimo prikazati neku drugu stranicu, idemo na navigaciju i označimo prostoriju koju želimo prikazati. Link poziva javascript skriptu na stranici da sakrije sve prostorije osim označene.

Unutar okvira prostorija se učitaju trošila i senzori složeni po svojim ID-ovima. Također se unutar okvira trošila dodaju trošila i senzori koji imaju naziv prostorije „xxx“. To su uređaji koji se prikazuju u svim prostorijama. Svako trošilo ili senzor ima svoju ikonu, koja posjeduje link na javascript skriptu, koja šalje naredbe M-Duino PLC-u.

```
<!-- ############################ SADRŽAJ/STRANICE ################################ -->
<!-- ############################ SADRŽAJ/STRANICE ################################ -->
<?php
  $prostor_H1 = '<div class="container NV_tekst" id="';
  $prostor_H2 = '" style="display:none"><div style="padding:3%; font-size:3vw
!important;">';
  $prostor_H3 = '</div><div class="row justify-content-start">';
  $prostor_F1 = '</div></div>
  ';

  $trosilo_H1 = '<div class="col-2 p-3"><figure><img class="img-fluid ';
  $trosilo_H1_prem = '" onClick="akcija(';
  $trosilo_H2 = ')" src="';
  $trosilo_H2_sen = '" src="';
  $trosilo_H_TER = '<div class="col-2 p-3">';
  $trosilo_B1 = '" id="';
  $trosilo_B2 = '"><figcaption>'; //iza ovoga dođe naziv trošila, ili kompozit naziva i
podatka senzora
  $trosilo_F1 = '</figcaption></figure></div>
  ';
  $trosilo_F_TER = '</div>
  ';
  $xxx_trosila = '';
  $termostat_html = '';
  $termostat_prostorija = '';

  $result_trosila = $conn->query($query_trosila . " WHERE `prostor`='xxx' ORDER BY `tip`");
  $buffer;
  while ($row_trosila = mysqli_fetch_row($result_trosila)) {

      if ($row_trosila[2] == 0) {
          $buffer = 1;
      } else {
          $buffer = 0;
      }
      if ($row_trosila[1] == 3) {
          //$xxx_trosila .= $trosilo_H_TER . 'termostat' . $trosilo_F_TER;
```

```
      }
      if ($row_trosila[1] == 1) {
          $xxx_trosila .= $trosilo_H1 . 's' . $row_trosila[0] . $trosilo_H1_prem .
$row_trosila[0] . ', ' . $buffer . ', ' . $row_trosila[4] . ', \'' . $row_trosila[5] . '\',
' . $row_trosila[1] . $trosilo_H2;
          if ($row_trosila[2] == 0) {
              $xxx_trosila .= 'img/slike_tranzicija/SW_OFF.png';
          } else {
              $xxx_trosila .= 'img/slike_tranzicija/SW_ON.png';
          }
          $xxx_trosila .= $trosilo_B1 . 's' . $row_trosila[0] . $trosilo_B2 .
$row_trosila[6] . $trosilo_F1;
      }
      if ($row_trosila[1] == 2) {
          $xxx_trosila .= $trosilo_H1 . 's' . $row_trosila[0] . $trosilo_H1_prem .
$row_trosila[0] . ', ' . $buffer . ', ' . $row_trosila[4] . ', \'' . $row_trosila[5] . '\',
' . $row_trosila[1] . $trosilo_H2;
          if ($row_trosila[2] == 0) {
              $xxx_trosila .= 'img/slike_tranzicija/FAN_OFF.png';
          } else {
              $xxx_trosila .= 'img/slike_tranzicija/FAN_ON.png';
          }
          $xxx_trosila .= $trosilo_B1 . 's' . $row_trosila[0] . $trosilo_B2 .
$row_trosila[6] . $trosilo_F1;
      }
      if ($row_trosila[1] == 4) {
          $xxx_trosila .= $trosilo_H1 . 's' . $row_trosila[0] . $trosilo_H2_sen .
'img/slike_tranzicija/SENZOR.png' . $trosilo_B1 . 's' . $row_trosila[0] . $trosilo_B2 .
$row_trosila[6] . ': ' . (intval($row_trosila[3]) / 10) . $row_trosila[5] . $trosilo_F1;
      }
      if ($row_trosila[1] == 5) {
          $xxx_trosila .= $trosilo_H1 . 's' . $row_trosila[0] . $trosilo_H2_sen;
          if ($row_trosila[2] == 0) {
              $xxx_trosila .= 'img/slike_tranzicija/ALARM_OFF.png';
          } else {
              $xxx_trosila .= 'img/slike_tranzicija/ALARM_ON.png';
          }
          $xxx_trosila .= $trosilo_B1 . 's' . $row_trosila[0] . $trosilo_B2 .
$row_trosila[6] . $trosilo_F1;
      }
      if ($row_trosila[1] == 6) {
          $xxx_trosila .= $trosilo_H1 . 's' . $row_trosila[0] . $trosilo_H2_sen;
          if ($row_trosila[2] == 0) {
              $xxx_trosila .= 'img/slike_tranzicija/FIRE_OFF.png';
```

```
          } else {
              $xxx_trosila .= 'img/slike_tranzicija/FIRE_ON.png';
          }
          $xxx_trosila .= $trosilo_B1 . 's' . $row_trosila[0] . $trosilo_B2 .
$row_trosila[6] . $trosilo_F1;
      }
  }

  $i = 0;
  $result = $conn->query($query_katovi);

  while ($row_katovi = mysqli_fetch_row($result)) {

      $result_prostorije = $conn->query($query_prostorije . " WHERE `kat`='" .
$row_katovi[0] . "'");

      while ($row_prostorije = mysqli_fetch_row($result_prostorije)) {

          if ($row_prostorije[0] != 'xxx') {
              echo $prostor_H1 . 'p' . $i . $prostor_H2 . $row_katovi[0] . ' - ' .
$row_prostorije[0] . $prostor_H3;

              $result_trosila = $conn->query($query_trosila . " WHERE `prostor`='" .
$row_prostorije[0] . "' AND `kat`='" . $row_katovi[0] . "' AND `prostor` != 'xxx' ORDER BY
`tip`");
              $buffer;
              $termostat_prostorija = '';
              $termostat_html = '';
              while ($row_trosila = mysqli_fetch_row($result_trosila)) {

                  if ($row_trosila[2] == 0) {
                      $buffer = 1;
                  } else {
                      $buffer = 0;
                  }
                  if ($row_trosila[1] == 3) {
                      $termostat_prostorija = $row_prostorije[0];
                      $termostat_html = '
                      <div class="col-xs-5 col-sm-5 col-md-5 col-lg-4" style="color:#555
!important;">
                          <div class="card" style="background-color:rgba(255,255,255,0.7);">
                              <div class="card-header ttttt' . $row_trosila[0] . '">
                                  Termostat je ugašen
                              </div>
```

```
                            <div class="card-block">
                                <div class="row p-2">
                                    <div class="col-xs-4 col-sm-6 col-md-6 col-lg-6 tt' .
$row_trosila[0] . '"
                                        style="font-size:4.5vw !important;">
                                        25
                                    </div>
                                    <div id="t' . $row_trosila[0] . '" class="col-xs-4
col-sm-6 col-md-6 col-lg-6 t' . $row_trosila[0] . '"
                                        style="font-size:4.5vw !important;">
                                        35
                                    </div>
                                </div>
                                <div class="row">
                                    <div class="col-sm-10 col-md-10 col-lg-10">
                                        <input id="ttt' . $row_trosila[0] . '" class="ttt'
. $row_trosila[0] . '" style="width:90%" type="range"
                                            value="250" min="180" max="320">
                                    </div>
                                </div>
                                <div class="row p-2">
                                    <div class="col-sm-2 col-md-2 col-lg-2">
                                        <div class="btn-group btn-group-sm" role="group"
                                            aria-label="">
                                            <button class="btn btn-primary  btn-block"
                                                onClick="termostat_mod(\'off\', ' .
$row_trosila[0] . ')">OFF</button>
                                            <button class="btn btn-primary  "
                                                onClick="termostat_mod(\'heat\', ' .
$row_trosila[0] . ')">HEAT</button>
                                            <button class="btn btn-primary  btn-block"
                                                onClick="termostat_mod(\'cool\', ' .
$row_trosila[0] . ')">COOL</button>
                                        </div>
                                    </div><!-- /col -->
                                </div><!-- /row -->
                            </div>
                        </div>
                    </div>
                    <div class="tttt' . $row_trosila[0] . '"
style="display:none;">off</div>
                    <script>
                        var termostat_selektor' . $row_trosila[0] . ' =
document.getElementById("ttt' . $row_trosila[0] . '");
```

```
                        var postavljena_selektor' . $row_trosila[0] . ' =
document.getElementById("t' . $row_trosila[0] . '");
                        termostat_selektor' . $row_trosila[0] .
'.addEventListener(\'input\', (evt' . $row_trosila[0] . ') => {
                            zabrana_update = true;
                            postavljena_selektor' . $row_trosila[0] . '.innerHTML =
termostat_selektor' . $row_trosila[0] . '.value / 10;
                        });
                        termostat_selektor' . $row_trosila[0] .
'.addEventListener(\'touchend\', (evt' . $row_trosila[0] . ') =>
{
                            slanje_podataka_termostata(' . $row_trosila[0] . '); // ID
                            zabrana_update = false;
                        });
                        termostat_selektor' . $row_trosila[0] .
'.addEventListener(\'mouseup\', (evt' . $row_trosila[0] . ') => {
                            slanje_podataka_termostata(' . $row_trosila[0] . '); // ID
                            zabrana_update = false;
                        });
                    </script>';
                }
                if ($row_trosila[1] == 1) {
                    echo $trosilo_H1 . 's' . $row_trosila[0] . $trosilo_H1_prem .
$row_trosila[0] . ', ' . $buffer . ', ' . $row_trosila[4] . ', \'' . $row_trosila[5] . '\',
' . $row_trosila[1] . $trosilo_H2;
                    if ($row_trosila[2] == 0) {
                        echo 'img/slike_tranzicija/SW_OFF.png';
                    } else {
                        echo 'img/slike_tranzicija/SW_ON.png';
                    }
                    echo $trosilo_B1 . 's' . $row_trosila[0] . $trosilo_B2 .
$row_trosila[6] . $trosilo_F1;
                }
                if ($row_trosila[1] == 2) {
                    echo $trosilo_H1 . 's' . $row_trosila[0] . $trosilo_H1_prem .
$row_trosila[0] . ', ' . $buffer . ', ' . $row_trosila[4] . ', \'' . $row_trosila[5] . '\',
' . $row_trosila[1] . $trosilo_H2;
                    if ($row_trosila[2] == 0) {
                        echo 'img/slike_tranzicija/FAN_OFF.png';
                    } else {
                        echo 'img/slike_tranzicija/FAN_ON.png';
                    }
                    echo $trosilo_B1 . 's' . $row_trosila[0] . $trosilo_B2 .
$row_trosila[6] . $trosilo_F1;
```

```
                }
                if ($row_trosila[1] == 4) {
                    echo $trosilo_H1_sen . 's' . $row_trosila[0] . $trosilo_H1_prem .
$trosilo_H2_sen . 'img/slike_tranzicija/SENZOR.png' . $trosilo_B1 . 's' . $row_trosila[0] .
$trosilo_B2 . $row_trosila[6] . ': ' . (intval($row_trosila[3]) / 10) . $row_trosila[5] .
$trosilo_F1;
                }
                if ($row_trosila[1] == 5) {
                    echo $trosilo_H1 . 's' . $row_trosila[0] . $trosilo_H2_sen;
                    if ($row_trosila[2] == 0) {
                        echo 'img/slike_tranzicija/ALARM_OFF.png';
                    } else {
                        echo 'img/slike_tranzicija/ALARM_ON.png';
                    }
                    echo $trosilo_B1 . 's' . $row_trosila[0] . $trosilo_B2 .
$row_trosila[6] . $trosilo_F1;
                }
                if ($row_trosila[1] == 6) {
                    echo $trosilo_H1 . 's' . $row_trosila[0] . $trosilo_H2_sen;
                    if ($row_trosila[2] == 0) {
                        echo 'img/slike_tranzicija/FIRE_OFF.png';
                    } else {
                        echo 'img/slike_tranzicija/FIRE_ON.png';
                    }
                    echo $trosilo_B1 . 's' . $row_trosila[0] . $trosilo_B2 .
$row_trosila[6] . $trosilo_F1;
                }
            }
            echo $xxx_trosila;
            if ($termostat_prostorija == $row_prostorije[0]) {
                echo $termostat_html;
            }
            echo $prostor_F1;
            $i++;
        }
    }
  }
  echo "<script>var broj_prostorija = " . $i . ";</script>";
  ?>
<!-- ############################## SADRŽAJ/STRANICE ################################## -->
<!-- ############################## SADRŽAJ/STRANICE ################################## -->
```

7.4 Slanje naredbe na M-Duino PLC

Korisnik na stranici, klikom na ikonu npr. nekog trošila, pokreće akciju slanja naredbe na M-Duino PLC. Naredba sa ikone trošila se obrađuje na način da se podaci, redom id, boolean, podatak_set i mod spajaju u tekstualni oblik. Primjer poruke: „?1!1!254!cool“. Stranica svakih 500ms ispituje da li postoji naredba ili naredbe spremne za slanje. I ako postoje, javascript šalje HTTP JSONP zahtjev M-Duino PLC-u. Primjer potpunog URL-a za slanje PLC-u: „/naredbe?1!1!254!cool“ (postaviti će termostat pod ID-om 1 u cool mode i postaviti zadanu temperaturu na 25.4°C)

```
 //Funkcija koja šalje kodove na M-Duino PLC - Arduino
$.fn.naredbe = function() {
    //ajax zahtjev prema poslužitelju na relejnom modulu - Arduinu
    $.ajax({
        url: Rel_modul_server + '/naredbe' + URL_str + '?',

        //Naziv callback parametra
        jsonpCallback: "callbackMoivet",

        //Odgovor relejnog modula - Arduina će biti u JSONP formatu
        dataType: "jsonp",

        //Format je tekst
        data: {
            format: "text"
        },

        /*
        Odgovor poslužitelja. Poslužitelj zapravo ne treba slati direktan odgovor za set
poslanih naredbi. Samo ih mora obraditi
        WEB poslužitelj stranice pametne kuće će redovito (CRON job) slati zahtjeve za
provjeru stanja svih trošila/senzora
        */
        success: function(response) {}
    });
    if ((Math.floor(Date.now() / 1000) - timespamp) >= timestamp_offset) {
        timespamp = Math.floor(Date.now() / 1000);
    }
}
```

```
/*
Funkcija sa vremenskim okidačem, koja svakih 500ms provjerava da li ima naredbi za slanje.
Ako ima, iste šalje na M-Duino PLC - Arduino
*/
setInterval(function() {
    if (URL_str != "") {
        $.fn.naredbe();
    }
    URL_str = "";
}, 500);
```

7.5 Osvježavanje stanja trošila i senzora iz baze podataka

Stranica svakih 500ms ispituje da li su zadovoljeni svi uvjeti za osvježavanje korisničkog sučelja. Da li su sve naredbe u redu čekanja poslane. Ako su svi uvjeti zadovoljeni, stranica će poslati zahtjev za osvježavanje stanja trošila i senzora na stranici. Podaci koje dobije, javascript skripta obrađuje, te mijenja ikone stanja svih trošila ili senzora. Čak i onih koji su skriveni unutar skrivenih dijelova stranice, odnosno prostorija. Tako ako korisnik označi neku novu prostoriju, stanje trošila i senzora će već bit osvježeno.

```
//Funkcija čita podatke o statusu trošila/senzora iz baze podataka
var oscilacije = false;

$.fn.obnova = function() {
    //ajax zahtjev prema poslužitelju web stranice pametne kuće
    var podaci;

    $.getJSON('provjera_konstrukcija.php?interno=trosila', function(podaci) {
        $.each(podaci, function(key, val) {

            var buff = [];

            $.each(val, function(kljuc, podatak) {
                buff.push(podatak);
            });

            var bool_buff = 0;
```

```
            if (buff[2] == 0) {
                bool_buff = 1;
            } else {
                bool_buff = 0;
            }
            if (buff[1] == 1) { //rasvjeta
                $(".s" + buff[0]).attr("onclick", "akcija(" + buff[0] + ", " + bool_buff +
", " + buff[4] + ", '" + buff[5] + "', " + buff[1] + ")");
                if (buff[2] == 0) {
                    $(".s" + buff[0]).attr("src", "img/slike_tranzicija/SW_OFF.png");
                } else {
                    $(".s" + buff[0]).attr("src", "img/slike_tranzicija/SW_ON.png");
                }
            }
            if (buff[1] == 2) { //ventilator
                $(".s" + buff[0]).attr("onclick", "akcija(" + buff[0] + ", " + bool_buff +
", " + buff[4] + ", '" + buff[5] + "', " + buff[1] + ")");
                if (buff[2] == 0) {
                    $(".s" + buff[0]).attr("src", "img/slike_tranzicija/FAN_OFF.png");
                } else {
                    $(".s" + buff[0]).attr("src", "img/slike_tranzicija/FAN_ON.png");
                }
            }
            if (buff[1] == 5) { //alarm
                if (buff[2] == 0) {
                    $(".s" + buff[0]).attr("src", "img/slike_tranzicija/ALARM_OFF.png");
                } else {
                    if (oscilacije == 1) {
                        $(".s" + buff[0]).attr("src", "img/slike_tranzicija/ALARM_ON.png");
                        //window.alert("ON");
                    } else if (oscilacije == 5) {
                        $(".s" + buff[0]).attr("src", "img/slike_tranzicija/ALARM_OFF.png");
                        //window.alert("OFF");
                    }
                }
            }
            if (buff[1] == 6) { //vatrodojava
                if (buff[2] == 0) {
                    $(".s" + buff[0]).attr("src", "img/slike_tranzicija/FIRE_OFF.png");
                } else {
                    if (oscilacije == 1) {
                        $(".s" + buff[0]).attr("src", "img/slike_tranzicija/FIRE_ON.png");
                    } else if (oscilacije == 6) {
                        $(".s" + buff[0]).attr("src", "img/slike_tranzicija/FIRE_OFF.png");
```

```
                    }
                }
            }
            if (buff[1] == 3) { //termostat
                if (zabrana_update == false) {
                    $(".tttt" + buff[0]).text(buff[5]); //MOD rada
                    $(".tt" + buff[0]).text(buff[3] / 10); //Temperatura
                    $(".t" + buff[0]).text(buff[4] / 10); //Postavljeno
                    $(".ttt" + buff[0]).val(buff[4]); //Temperatura

                    if (buff[5] == 'off') {
                        $('.ttttt' + buff[0]).text('Termostat je ugašen');
                    } else if (buff[5] == 'heat') {
                        $('.ttttt' + buff[0]).text('Grijanje - Termostat');
                    } else if (buff[5] == 'cool') {
                        $('.ttttt' + buff[0]).text('Hlađenje - Termostat');
                    }
                }
            }
            if (buff[1] == 4) { //senzor (figcaption)
                $(".s" + buff[0]).next().text(buff[6] + ': ' + Math.round(buff[3] / 10) +
buff[5]);
            }

        });
    });

    URL_privremena_zabrana = [];
    if (oscilacije == 8) { //Oscilacija promjene ikona za Alarme
        oscilacije = 0;
    } else {
        oscilacije++;
    }

}
```

7.6 PHP skripa za komunikaciju sa bazom podataka

PHP skripta *provjera_konstrukcija.php* je skripta koja obrađuje zahtjeve sa glavne stranice index.php. Svi zahtjevi, osim zahtjeva sa naredbom prema M-Duino PLC-u, pozivaju ovu skriptu, koja obrađuje zahtjev, spaja se na bazu podataka, prikuplja podatke, te ih u JSON formatiranom obliku prosljeđuje glavnoj stranici. Glavna stranica pomoću ove skripte pristupa bazi podataka na način da javascript šalje HTTP GET zahtjev ovoj skripti sa dodatnim parametrima. Ovisno što stranici treba. Ako stranica želi podatke o konstrukciji poslati će zahtjev: „GET /provjera_konstrukcija.php?interno=konstrukcija“. A, ako stranica želi podatke o stanju trošila i senzora, poslati će HTTP GET zahtjev: „GET /provjera_konstrukcija.php?interno=trosila“. Ova skripta može služiti i za provjeru da li je došlo do promijena u konstrukciji pametne kuće. Te na taj način signalizirati glavnoj stranici da izvrši auto osvježavanje sadržaja.

```
<?php
  $servername = "localhost";
  $username = "admin";
  $password = "root";
  $dbname = "pametnakuca";
  $conn = new mysqli($servername, $username, $password, $dbname);

  $P_konstrukcija = [];
  $P_trosila = [];
  $P_tro = [];

  $query_katovi = "SELECT DISTINCT kat FROM `trosila`";
  $query_prostorije = "SELECT DISTINCT `prostor` FROM `trosila`";
  $query_trosila = "SELECT `id`, `tip`, `bool`, `podatak`, `podatak_set`, `mod`, `ime`,
`kod` FROM `trosila`";
  $query_trosila_kratki = "SELECT `id`, `tip`, `bool`, `podatak`, `podatak_set`, `mod`,
`ime` FROM `trosila`";

  if ($_SERVER["REQUEST_METHOD"] == "GET") {
      if ($_GET['interno'] == "konstrukcija") {
          $result = $conn->query($query_katovi);
          if (!$result) {
              //echo "GRESKA";
              exit();
          } else {
              while ($row = mysqli_fetch_row($result)) {
```

```
                $result_prostorije = $conn->query($query_prostorije . " WHERE `kat`='" .
$row[0] . "'");
                while ($row_prostorije = mysqli_fetch_row($result_prostorije)) {
                    //$P_konstrukcija[$row[0]]
                    $result_trosila = $conn->query($query_trosila . " WHERE `prostor`='" .
$row_prostorije[0] . "' AND `kat`='" . $row[0] . "' ORDER BY `tip`");
                    $i = 0;
                    while ($row_trosila = mysqli_fetch_row($result_trosila)) {
                        $P_konstrukcija[$row[0]][$row_prostorije[0]][$i][0] =
$row_trosila[0];
                        $P_konstrukcija[$row[0]][$row_prostorije[0]][$i][1] =
$row_trosila[1];
                        $P_konstrukcija[$row[0]][$row_prostorije[0]][$i][2] =
$row_trosila[2];
                        $P_konstrukcija[$row[0]][$row_prostorije[0]][$i][3] =
$row_trosila[3];
                        $P_konstrukcija[$row[0]][$row_prostorije[0]][$i][4] =
$row_trosila[4];
                        $P_konstrukcija[$row[0]][$row_prostorije[0]][$i][5] =
$row_trosila[5];
                        $P_konstrukcija[$row[0]][$row_prostorije[0]][$i][6] =
$row_trosila[6];
                        $P_konstrukcija[$row[0]][$row_prostorije[0]][$i][7] =
$row_trosila[7];
                        $i++;
                    }
                }
            }
        }
        echo json_encode($P_konstrukcija);
    } else if ($_GET['interno'] == "trosila") {
        $result = $conn->query($query_trosila_kratki);
        if (!$result) {
            //echo "GRESKA";
            exit();
        } else {
            $h = 0;
            while ($row = mysqli_fetch_row($result)) {
                $P_tro[0] = $row[0];
                $P_tro[1] = $row[1];
                $P_tro[2] = $row[2];
                $P_tro[3] = $row[3];
                $P_tro[4] = $row[4];
                $P_tro[5] = $row[5];
```

```
                $P_tro[6] = $row[6];
                $P_trosila[$h] = $P_tro;
                $h++;
            }
        }
        echo json_encode($P_trosila);
    }
}
$conn->close();
?>
```

8. Zaključak

Opisani sustav internetske kontrole pametne kuće je osmišljen kao pokazno predstavljanje načina na koji sustavi kontrole pametne kuće rade. Iako, upotrijebljene tehnologije nisu primarni izbor za kvalitetan konačni sustav. HTTP komunikacija je sasvim dovoljna za pokazno predstavljanje sustava.

Ovo rješenje ipak ima određene mane. Prvenstveno sigurnost, kao jednu od osnovnih funkcija pametne kuće. Također, ovo rješenje je izvedeno po principu konstantnog pozivanja poslužitelja, bilo M-Duina, bilo Raspberry PI računala. Profesionalni sustavi rade po principu okidača. Što smanjuje korištenje komunikacijskih kanala i fragmentaciju komunikacije. Arduino uređaj, koliko je atraktivan po jednostavnosti korištenja, ipak je jednostavan mikrokontroler, ograničen frekvencijom glavnog oscilatora i količinom memorije. Odnosno, ne može zadovoljiti brzinom obrade podataka i nema napredna programska rješenja.

Ipak, ovaj primjer upravljanja se može koristiti u svrhu kontrole jednostavnih pametnih kuća. Korisničko sučelje je osnova sustava, dok se dodatnim programiranjem M-Duino PLC-a može proširiti funkcionalnost istog. Mogu se dodavati specijalne scene za paljenje trošila i podešavanje senzora. Npr. „Dobrodošli“ tipkalo, koje može upaliti termostat na željeni mod rada i temperaturu, te upaliti rasvjetu na prilazu. Također, na M-Duino PLC-u se mogu programirati funkcije za komunikaciju sa dodatnim modulima za proširenje sustava.

Projektiranje i izrada kompleksnih sustava, kao što je internetska aplikacija za kontrolu pametne kuće, zahtjeva mnogo vremena. Potrebno je pomno konstruirati komunikacijski protokol, korisničko sučelje, obradu podataka... Stoga je povratna informacija korisnika krucijalna u otklanjanu nedostataka i u unaprijeđenju sustava. I stvaranju konačne aplikacije.

9. Literatura

[1] https://en.wikipedia.org/wiki/Transmission_Control_Protocol

[2] https://en.wikipedia.org/wiki/HTTP

[3] https://www. https://www.raspberrypi.org/

[4] https://www.linux.org/

[5] https://www.industrialshields.com/

[6] https://www.arduino.cc/

[7] https://httpd.apache.org/

[8] https://www.microsoft.com/en-us/windows?r=1

[9] https://www.raspbian.org/

[10] https://www.debian.org/

[11] https://www.php.net/

[12] https://www.javascript.com/

[13] https://jquery.com/

[14] https://en.wikipedia.org/wiki/HTML

[15] https://mariadb.org/

[16] https://www.mysql.com/

[17] https://www.arduino.cc/en/software

[18] https://getbootstrap.com/

[19] https://en.wikipedia.org/wiki/CSS

10.Popis slika

Slika 2.1 - Mogućnost spajanja senzora u sustavu pametne kuće..... 7
Slika 2.2 - Nacrt pokazne ploče upravljanja pametnom kućom 8
Slika 4.1 - Grafički prikaz komunikacijskog algoritma pametne kuće..... 13
Slika 7.1 - Korisničko sučelje 38

11.Popis tablica

Tablica 2.1 - Raspbarry PI karakteristike 4
Tablica 2.2 – M-Duino 38AR+..... 5