VoCom – upravljanje govornim naredbama

Izvedbeno rješenje

Članovi tima:

- Antonio Bukovec
- Mario Kodba
- Dino Šarić
- Ivan Vrhoci

Sadržaj:

Postavljanje Google Home Mini uređaja

Nakon nabavke Google Home Mini uređaja, potrebno je nabaviti ili kompatibilan strujni adapter za europsko tržište ili koristiti bilo koji smartphone adapter koji je u mogućnosti dati struju jakosti 1.8 A ili više. Nakon priključenja Google Home Mini uređaja na napajanje, potrebno je na smartphone uređaju preuzeti Google Home aplikaciju, kreirati gmail račun ako ga već ne posjedujete, prijaviti se s tim računom unutar aplikacije te pritisnuti “Add” tipku te nakon toga “Set up device”. Uz praćenje daljnjih koraka na aplikaciji, kroz nekoliko minuta Google Home Mini uređaj trebao bi biti povezan na vašu wifi mrežu i spreman za korištenje.

Ovako konfiguriran Google Home Mini sposoban je izvršavati mnoštvo glasovnih naredbi kao što su: dodavanje podsjetnika, postavljanje alarma, dodavanje događaja u kalendar, davanje navigacijskih uputa i procjene potrebnog vremena za dolazak, pronalaženje vremenskih prognoza, pronalaženje “korak po korak” recepata, rezimiranje sportskih događaja, pričanje viceva, pronalaženje odgovara na postavljena pitanja itd.

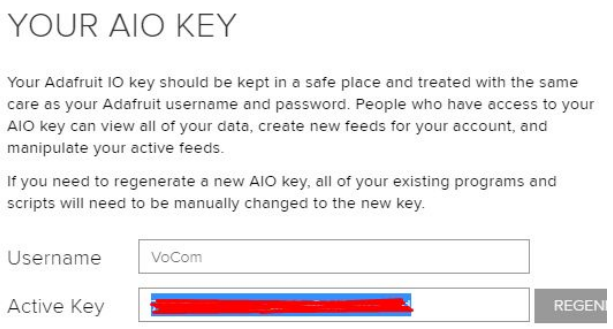
Osim navedenih naredbi, u sklopu ovog projekta bit će realizirani podsustavi koji će omogućiti realizaciju dodatnih naredbi. Realizirat će se podsustavi za:

- upravljanje intenzitetom svjetla
- slanje IC (infra crvenih) naredbi
- detekciju prisutnosti i “upravljanje” vratima*
- upravljanje kućanskim aparatima

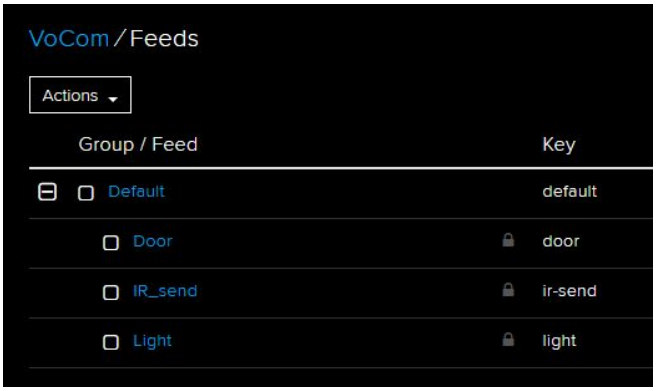
*upravljanje vrata će se samo simulirati paljenjem LED diode

Adafruit IO sučelje

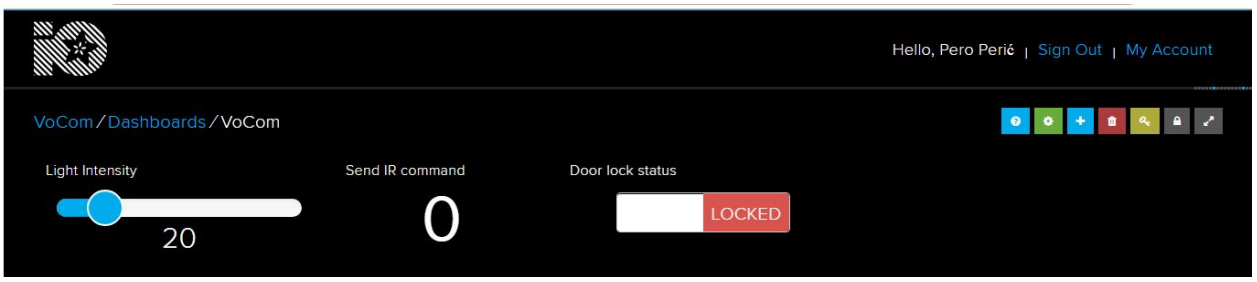
Nadalje, potrebno je kreirati Adafruit korisnički račun kako bi se koristile mogućnosti Adafruit IO servisa. Nakon prijave na https://io.adafruit.com/ potrebno je kreirati *Dashboard* u kojem će se zatim kreirati razni feed-ovi koji će biti korišteni u projektu. Pritiskom na oznaku '+' u gornjem desnom uglu, otvara se izbornik u kojem je moguće odabrati tip bloka koji će se koristiti. Mi smo odabrali slider za podsustav promjene intenziteta svjetla, text blok za IC podsustav i toggle za podsustav detekcije prisutnosti i upravljanje vratima. Nakon toga, potrebno je stvoriti feed za pojedini blok (npr. u našem slučaju: Light, IR_send i Door). Za upravljanje ovim feedovima, potrebno je znati korisničko ime računa i aktivacijski ključ. Njihova primjena bit će prikazana u pseudokodovima.



Slika 1. Aktivacijski ključ



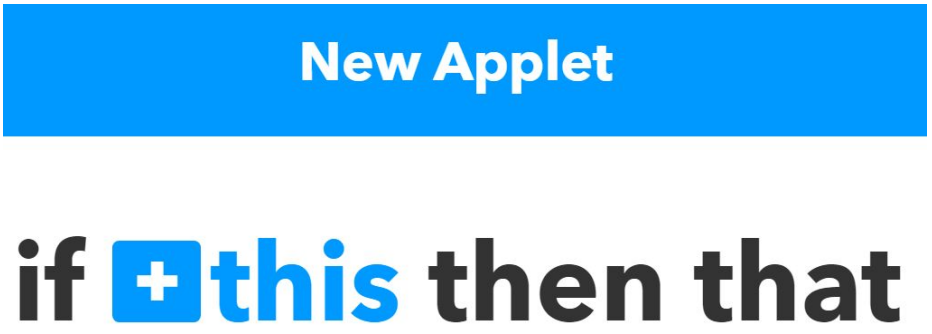
Slika 2. Pregled korištenih feedova



Slika 3. Kreirani dashboard

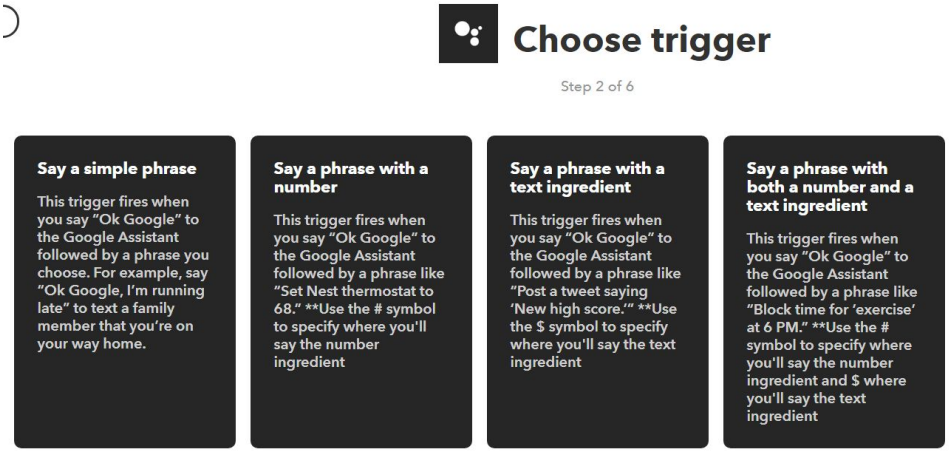
IFTTT aplikacije

Kako bi povezali Adafruit IO i naš Google Home uređaj, koristi ćemo IFTTT platformu. Potrebno je preuzeti IFTTT aplikaciju na smartphone uređaj i kreirati korisnički račun. Zatim je potrebno kreirati potrebne aplikacije za svaki od podsustava ili za svaku željenu naredbu.

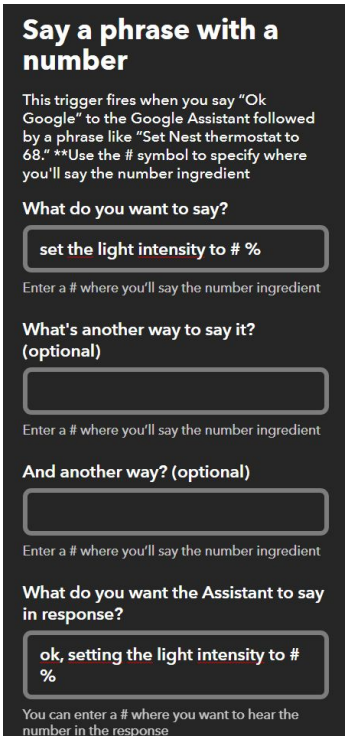


Slika 4. Kreiranje IFTTT korisničke aplikacije

U izborniku kreiranja nove aplikacije moguće je odrediti “this” i “that” dio. “This” dio predstavlja okidač za izvedbu “that” dijela. Kao okidač za naše aplikacije potrebno je odabrati Google Assistant. Moguće je odabrati nekoliko različitih okidača (jednostavna fraza, fraza koja sadrži broj (#), fraza koja sadrži određeni tekst ($) i fraza koja sadrži oboje). Za upravljanje intenzitetom svjetla odabrana je fraza koja sadrži broj kako bi se prenijela informacija o intenzitetu. Za slanje IR naredbi i upravljanje vratima koristit će se jednostavne fraze.

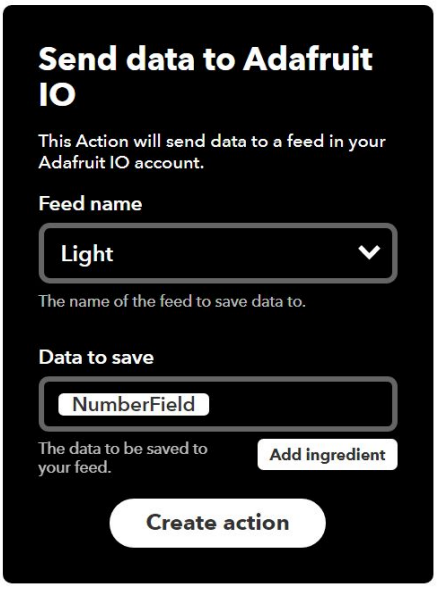


Slika 5. Google Assistant izbornik



Slika 6. Primjer okidača za upravljanje intenzitetom svjetla

Za "that" dio potrebno je odabrati Adafruit i povezati svoj Adafruit račun (koristeći korisničko ime i aktivacijski ključ) sa IFTTT platformom. Nakon toga potrebno je još odabrati feed na koji će okidač djelovati (Light) i što će se upisati u taj feed (broj koji smo izgovorili).



Slika 7. Adafruit dio aplikacije

Za podsustav IR upravljanja koristit će se fraze tipa “Turn on the TV”, “Increase the TV volume”... i svakoj od fraza bit će pridružen određen broj koji će se slati na Adafruit IO IR_send feed.

Za podsustav detekcije prisutnosti i upravljanja vratima, koristit će se fraze tipa “Unlock/Lock the door” i slat će se broj 0/1 na Adafruit IO Door feed.

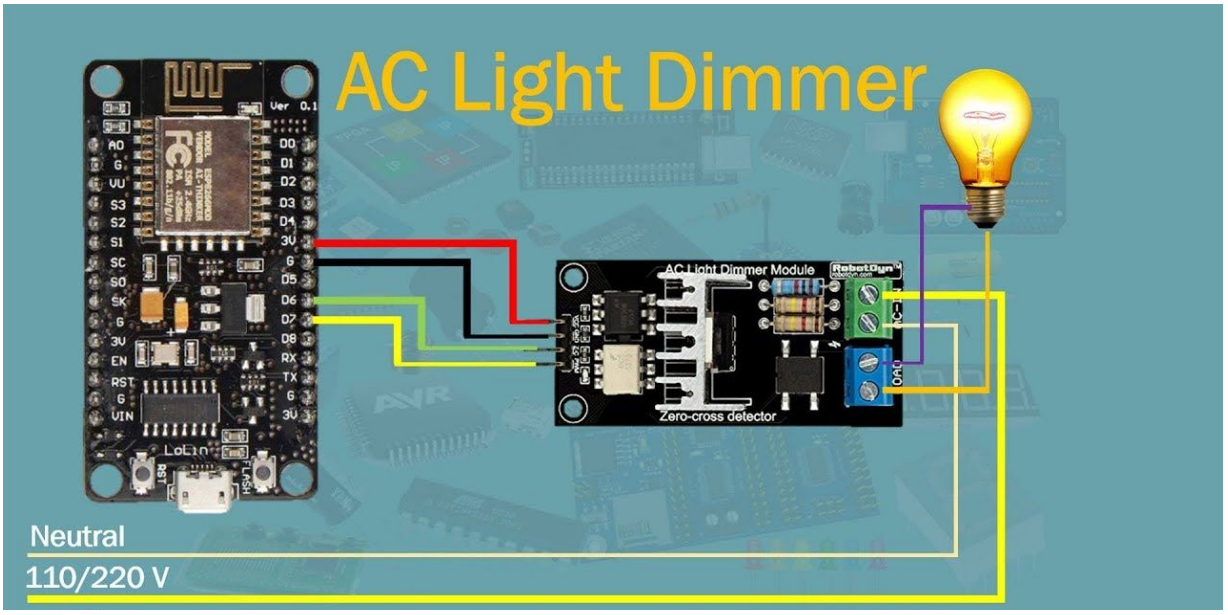
Za podsustav upravljanja kućanskim uređajima (uz korištenje sonoff wifi sklopki) kao okidač će se također koristiti Google Assistant, a “that” dio će biti “eWeLink Smart Home” za čije je korištenje potrebno preuzeti eWeLink aplikaciju, napraviti korisnički račun te sinkronizirati sonoff wifi sklopku s vašim računom. Zatim je potrebno odabrati “Turn 1 channel switch on or off”, odabrati željenu sklopku te njeno stanje (on/off). Za okidanje će se koristiti fraze tipa “Make me coffee”/”Turn off the coffee machine”.

Korištenjem drugih servisa na IFTTT platformi (poput Android Device, Android SMS, VoIP calls...) moguće je primjerice kreirati aplikacije za puštanje glazbe sa vašeg smartphone uređaja, slanje SMS poruka te glasovnih poruka u obliku poziva.

Sheme podsustava

- upravljanje intenzitetom svjetla

  Upravljanje intenzitetom svjetla izvodit će se pomoću NodeMCU v3 tiskane pločice koja se bazira na ESP8266 te AC light dimmer-a tvrtke RobotDyn. Primjer sheme prikazan je na slici 8. Kao output pin (PWM) koristi će se pin D7, a kao zero-cross pin odabran je pin D6. Za napajanje NodeMCU može se koristiti 5V adapter ili powerbank.

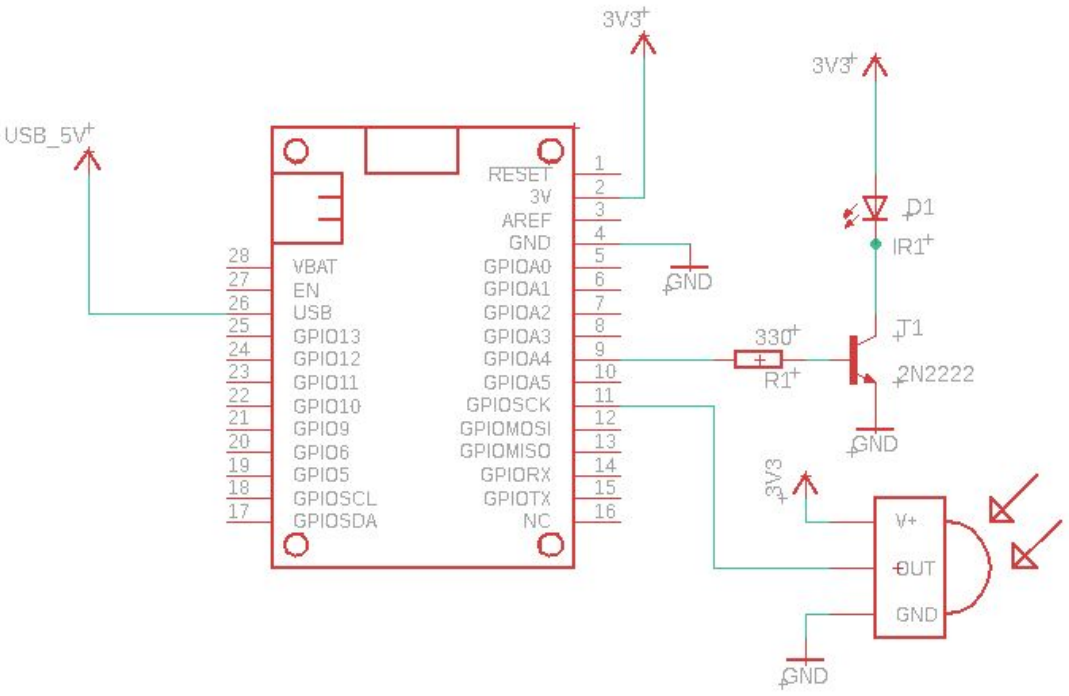


Slika 8. Shema podsustava za upravljanje intenzitetom svjetla

- slanje IC naredbi

Upravljanje TVom i klimom, tj. slanje IC signala izvodit će se pomoću NodeMCU v3 tiskane pločice koja se bazira na ESP8266 modulu uz pomoć IC modula za primanje IC signala i IC diode. Primjer cjelokupne sheme prikazan je na slici 9.

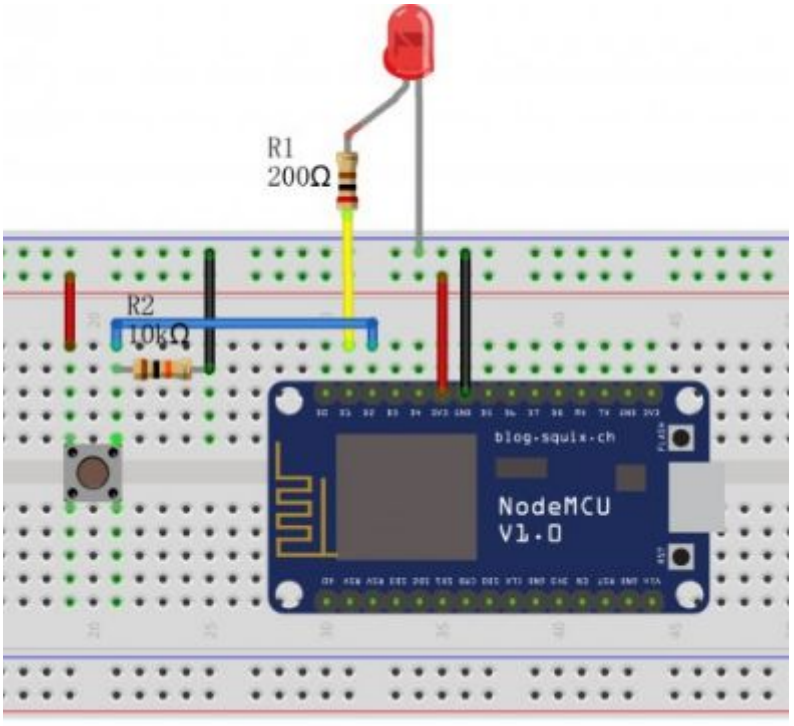
Prije slanja signala putem IC diode potrebno je odrediti koji signal poslati. Dekodiranje potrebnog signala se obavlja putem IC prijemnika i originalnog upravljača TVa/klima uređaja. Za napajanje NodeMCU može se koristiti 5V adapter ili powerbank.



Slika 9. Shema podsustava za primanje i slanje IC signala

- detekcija prisutnosti i “upravljanje” vratima

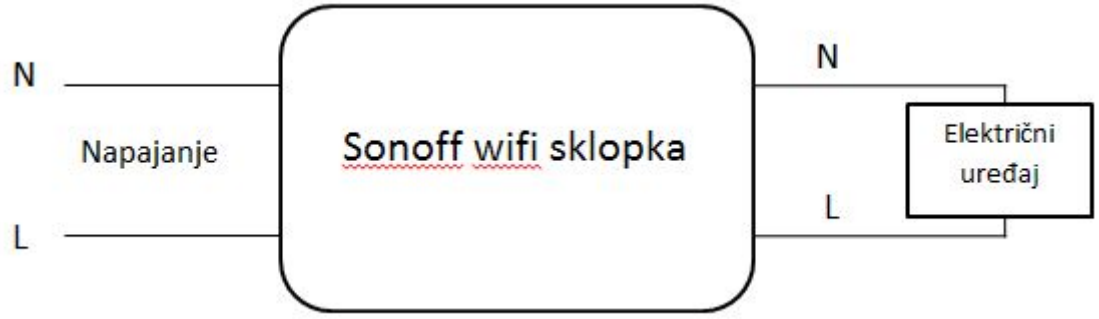
  Detekcija prisutnosti i “upravljanje” vratima izvodit će se pomoću NodeMCU v3 tiskane pločice koja se bazira na ESP8266. Za napajanje NodeMCU može se koristiti 5V adapter ili powerbank. Simulacija prisutnosti na vratima izvršit će se preko gumba dok će se otključavanje vrata simulirati preko paljenja/gašenja LED diode. Umjesto diode predlaže se uporaba modula za otključavanje/zaključavanje vrata.



Slika 11. Shema simulacije podsustava za detekciju prisutnosti i upravljanja vratima

- upravljanje kućanskim aparatima

Upravljanje kućanskim aparatima izvodit će se preko sonoff wifi sklopke. Sonoff sklopka za rad treba AC napajanje 90V-250 V, 50/60 Hz



Slika 12. Shema podsustava za upravljanje kućanskim aparatima

Pseudokodovi

- upravljanje intenzitetom svjetla

Korištene su Adafruit MQTT Library i RBDDimmer biblioteke. Potrebno je ispuniti WLAN_SSID i WLAN_PASS.

```
#include <ESP8266WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"
#include <RBDdimmer.h>
/************************* WiFi Access Point *********************************/
#define WLAN_SSID       "-----"
#define WLAN_PASS       "-----"
/************************* ESP8266 pin config *********************************/
#define outputPin  13 //D7 PWM
#define zerocross  12 //D6 ZC
/************************* Adafruit.io Setup *********************************/
#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883
#define AIO_USERNAME    "-----"
#define AIO_KEY         "-----"
/************ Global State ******************/
// Create an ESP8266 WiFiClient class to connect to the MQTT server.
WiFiClient client;
// Setup the MQTT client class by passing in the WiFi client and MQTT server and login details.
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT, AIO_USERNAME, AIO_KEY);
/****************************** Feeds ***************************************/
// Notice MQTT paths for AIO follow the form: <username>/feeds/<feedname>
// Setup feeds for subscribing to changes.
Adafruit_MQTT_Subscribe light_intensity = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME "/feeds/light");
/****************************** Dimmer Initialisation*************************/
dimmerLamp dimmer(outputPin, zerocross); //initialase port for dimmer for ESP8266, ESP32, Arduino due boards
int outVal = 0;
/*************************** Sketch Code ************************************/
void setup() {
  Serial.begin(115200);
  delay(10);

  Serial.println(F("Adafruit MQTT demo"));

  // Connect to WiFi access point.
  Serial.println(); Serial.println();
  Serial.print("Connecting to ");
  Serial.println(WLAN_SSID);
  WiFi.begin(WLAN_SSID, WLAN_PASS);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println();
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());

  // Setup MQTT subscription
  mqtt.subscribe(&light_intensity);

  // Dimmer initialisation
  dimmer.begin(NORMAL_MODE, ON); //dimmer initialisation: name.begin(MODE, STATE)
  Serial.println("Dimmer Program is starting...");
  Serial.println("Set value");
}
```

```
void loop() {
  MQTT_connect();

  Adafruit_MQTT_Subscribe *subscription;
  while ((subscription = mqtt.readSubscription(5000))) {

    if (subscription == &light_intensity){
    Serial.print(F("Got: "));
    uint16_t intensity=atoi((char *)light_intensity.lastread);
    Serial.println(intensity);

    int preVal = outVal;

      if (intensity < 101 && intensity > -1) outVal = intensity;
      delay(200);
      dimmer.setPower(outVal); // setPower(0-100%);
      delay(50);
   }

 }
  // ping the server to keep the mqtt connection alive
  if(! mqtt.ping(3)) {
    mqtt.disconnect();
  }
}

// Function to connect and reconnect as necessary to the MQTT server.
// Should be called in the loop function and it will take care if connecting.
void MQTT_connect() {
  int8_t ret;

  // Stop if already connected.
  if (mqtt.connected()) {
    return;
  }
  Serial.print("Connecting to MQTT... ");
  uint8_t retries = 3;
  while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected
       Serial.println(mqtt.connectErrorString(ret));
       Serial.println("Retrying MQTT connection in 5 seconds...");
       mqtt.disconnect();
       delay(5000);  // wait 5 seconds
       retries--;
       if (retries == 0) {
         // basically die and wait for WDT to reset me
         while (1);
       }
  }
  Serial.println("MQTT Connected!");
}
```

- slanje IC naredbi

Korištene su Adafruit MQTT Library i IRremoteESP8266 biblioteke. Potrebno je ispuniti WLAN_SSID i WLAN_PASS.

```
#include <ESP8266WiFi.h>
#include <Adafruit_MQTT.h>
#include <Adafruit_MQTT_Client.h>

#include <IRremoteESP8266.h>
/************************* WiFi Access Point *********************************/

#define WLAN_SSID       "____"
#define WLAN_PASS       "_____"
/************************* ESP8266 pin config *********************************/

#define LED_BUILTIN 2 //D4, internal led
#define IR_SEND_PIN D2
#define DELAY_BETWEEN_COMMANDS 1000

IRsend irsend(IR_SEND_PIN);
/************************* Adafruit.io Setup *********************************/

#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883
#define AIO_USERNAME    "VoCom"
#define AIO_KEY         "7ec7f6fc12f546e9962a6d48f6576f4d"

/************ Global State ******************/
WiFiClient client;
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT, AIO_USERNAME, AIO_KEY);
/****************************** Feeds ***************************************/
Adafruit_MQTT_Subscribe IR_send = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME "/feeds/IR_send");

/*************************** Sketch Code ************************************/

void setup() {
irsend.begin();
pinMode(led, OUTPUT);
digitalWrite(led,1);
Serial.begin(115200);
delay(10);

Serial.println(F("Adafruit MQTT demo")); //cemu taj F?
// Connect to WiFi access point.
Serial.println(); Serial.println();
Serial.print("Connecting to ");
Serial.println(WLAN_SSID);

WiFi.begin(WLAN_SSID, WLAN_PASS);
while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
}
Serial.println();
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());

// Setup MQTT subscription
mqtt.subscribe(&IR_send);


Serial.println("Program is starting...");
Serial.println("Set value");
```

```
}


void loop() {


 MQTT_connect();

 Adafruit_MQTT_Subscribe *subscription;
 while ((subscription = mqtt.readSubscription(5000))) {

        if (subscription == &IR_send){
        Serial.print(F("Got: "));

        uint16_t IR_command=atoi((char *)IR_send.lastread);
        Serial.println(IR_command);

 switch (IR_command) {
        case 1:
        irsend.sendcommand(...);
        break;
        case 2:
        irsend.sendcommand(...);
        break;
        case 3:
        irsend.sendcommand(...);
        break;
        case 4:
        irsend.sendcommand(...);
        break;
        default:
        irsend.sendcommand(...);
        break;
 }
 // ping the server to keep the mqtt connection alive
 // NOT required if you are publishing once every KEEPALIVE seconds

 if(! mqtt.ping(3)) {
        mqtt.disconnect();
 }

 }

 // Stop if already connected.
 if (mqtt.connected()) {
        return;
 }

 Serial.print("Connecting to MQTT... ");

 uint8_t retries = 3;
 while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected
        Serial.println(mqtt.connectErrorString(ret));
        Serial.println("Retrying MQTT connection in 5 seconds...");
        mqtt.disconnect();
        delay(5000);  // wait 5 seconds
        retries--;
        if (retries == 0) {
        // basically die and wait for WDT to reset me
        while (1);
        }
 }
 Serial.println("MQTT Connected!");
 }
 }
```

- detekcija prisutnosti i “upravljanje” vratima

Korištene su Adafruit MQTT Library i Bounce2 biblioteke. Potrebno je ispuniti WLAN_SSID i WLAN_PASS.

```
#include <ESP8266WiFi.h>
#include <Bounce2.h>
#include <esp8266-google-home-notifier.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"


#define BUTTON_PIN 4   // D2/GPIO4
int led = D7;

WiFiClient client;
Adafruit_MQTT_Client mqtt(&client, MQTT_SERV, MQTT_PORT, MQTT_NAME, MQTT_PASS);

Adafruit_MQTT_Subscribe onoff = Adafruit_MQTT_Subscribe(&mqtt, MQTT_NAME "/f/onoff");
Adafruit_MQTT_Publish LightsStatus = Adafruit_MQTT_Publish(&mqtt, MQTT_NAME "/f/LightsStatus");


const char* ssid     = "<REPLASE_YOUR_WIFI_SSID>";
const char* password = "<REPLASE_YOUR_WIFI_PASSWORD>";

Bounce debouncer = Bounce();
GoogleHomeNotifier ghn;

void setup() {

 // Setup the button with an internal pull-up :
 pinMode(BUTTON_PIN,INPUT_PULLUP);

 // After setting up the button, setup the Bounce instance :
 debouncer.attach(BUTTON_PIN);
 debouncer.interval(5); // interval in ms

 Serial.begin(115200);
 Serial.println("");
 Serial.print("connecting to Wi-Fi");
 WiFi.mode(WIFI_STA);
 WiFi.begin(ssid, password);

 while (WiFi.status() != WL_CONNECTED) {
  delay(250);
  Serial.print(".");
 }
 Serial.println("");
 Serial.println("connected.");
 Serial.print("IP address: ");
 Serial.println(WiFi.localIP());  //Print the local IP

 const char displayName[] = "Family Room";  /// must match the name of Google Home

 Serial.println("connecting to Google Home...");
 if (ghn.device(displayName, "en") != true) {
  Serial.println(ghn.getLastError());
  return;
 }
 Serial.print("found Google Home(");
 Serial.print(ghn.getIPAddress());
 Serial.print(":");
 Serial.print(ghn.getPort());
 Serial.println(")");

 pinMode(LED_BUILTIN, OUTPUT);
 digitalWrite(LED_BUILTIN, LOW);
```

```
  mqtt.subscribe(&onoff);

  pinMode(led, OUTPUT);
  digitalWrite(LED_BUILTIN, HIGH);
  digitalWrite(led, LOW);


}

void loop() {
  MQTT_connect();
  debouncer.update();
  if ( debouncer.fell() ) {
    if (ghn.notify("Someone is at the door!") != true) {
      Serial.println(ghn.getLastError());
      return;
    }
  //Read from our subscription queue until we run out, or
  //wait up to 5 seconds for subscription to update
  Adafruit_MQTT_Subscribe * subscription;
  while ((subscription = mqtt.readSubscription(5000)))
  {
    //If we're in here, a subscription updated...
    if (subscription == &onoff)
    {
      //Print the new value to the serial monitor
      Serial.print("onoff: ");
      Serial.println((char*) onoff.lastread);

      //If the new value is  "ON", turn the light on.
      //Otherwise, turn it off.
      if (!strcmp((char*) onoff.lastread, "ON"))
      {
        //active low logic
        digitalWrite(led, HIGH);
        LightsStatus.publish("ON");
      }
      else if (!strcmp((char*) onoff.lastread, "OFF"))
      {
        digitalWrite(led, LOW);
        LightsStatus.publish("OFF");

      }
      else
      {
        LightsStatus.publish("ERROR");
      }
    }
    else
    {
      //LightsStatus.publish("ERROR");
    }
  }
  //  if (!mqtt.ping())
  //  {
  //    mqtt.disconnect();
  //  }

  }
}



void MQTT_connect()
{

  //  // Stop if already connected
  if (mqtt.connected() && mqtt.ping())
  {
    //    mqtt.disconnect();
    return;
```

```
  }

  int8_t ret;

  mqtt.disconnect();

  Serial.print("Connecting to MQTT... ");
  uint8_t retries = 3;
  while ((ret = mqtt.connect()) != 0) // connect will return 0 for connected
  {
    Serial.println(mqtt.connectErrorString(ret));
    Serial.println("Retrying MQTT connection in 5 seconds...");
    mqtt.disconnect();
    delay(5000);  // wait 5 seconds
    retries--;
    if (retries == 0)
    {
      ESP.reset();
    }
  }
  Serial.println("MQTT Connected!");
}
```

Detaljni timeline

18. travanj	Izrada projektnog zadatka
2. svibanj	Izrada idejnog rješenja
16. svibanj	Izrada izvedbenog rješenja - dizajn cijelog sustava
26. svibanj	Dovršetak svih podsustava
27. svibanj	Testiranje cjelokupnog sustava na fakultetu
30. svibanj	Izrada sustava s potrebnim komponentama - provjera s korisnikom
7. lipanj	Izmjena sustava u skladu s korisnikovim željama
11. lipanj	Izrada video materijala
12. lipanj	Izrada prezentacije
13. lipanj	Cjelokupna dokumentacija, video i prezentacija rada sustava

Primopredajni protokol

Nakon izrađenog sustava, izvršit će se završna provjera s korisnikom. Napravit će se izmjene, ukoliko će one biti potrebne, do krajnjeg definiranog roka.

Funkcionalnost sustava bit će demonstrirana pred korisnikom na fakultetu za uređaje koje je moguće donijeti i povezati na fakultetu, dok će funkcionalnost ostalih uređaja, radi jednostavnosti, biti prikazana putem video prezentacije.

Nakon zadane govorne naredbe, Google Home Mini daje potvrdu o primljenoj naredbi putem integriranog zvučnika, a stanje i izlaze pojedinih senzora i uređaja moguće je pratiti putem Adafruit IO besplatnog online servisa. Ako je naredba ispravno procesuirana, na određenom uređaju bi trebala biti vidljiva promjena (npr. postavljanje svjetlosti na 70% jačine). Cjelokupna dokumentacija, video i prezentacija bit će dostavljeni do 13. lipnja 2019.