

Sveprisutno računarstvo

Stamenković Teodora 1460
Profesor: Dragan Stojanović

Projekti

Projekat 1 - ThingsBoard

Projekat 2 i 3 - Smart Home aplikacija

Za sva 3 projekta su korišćeni Arduino Nano 33 BLE Sense Lite i Raspberry Pi 4



Arduino Nano 33 BLE Sense Lite

Senzori:

- APDS9960 senszor - **Proximity and Gesture Detection**
- LPS22HB senszor - **Barometric Pressure Sensor**
- LSM9DS1 senszor - **IMU for Motion Detection**



Projekat 1

Projekat 1

- Obrada i analiza podataka na open source platformi **ThingsBoard**
- Izvor podataka: Arduino Nano 33 BLE Sense Lite
- IoT aplikacija se izvršava na Edge-u -> Raspberry Pi 4

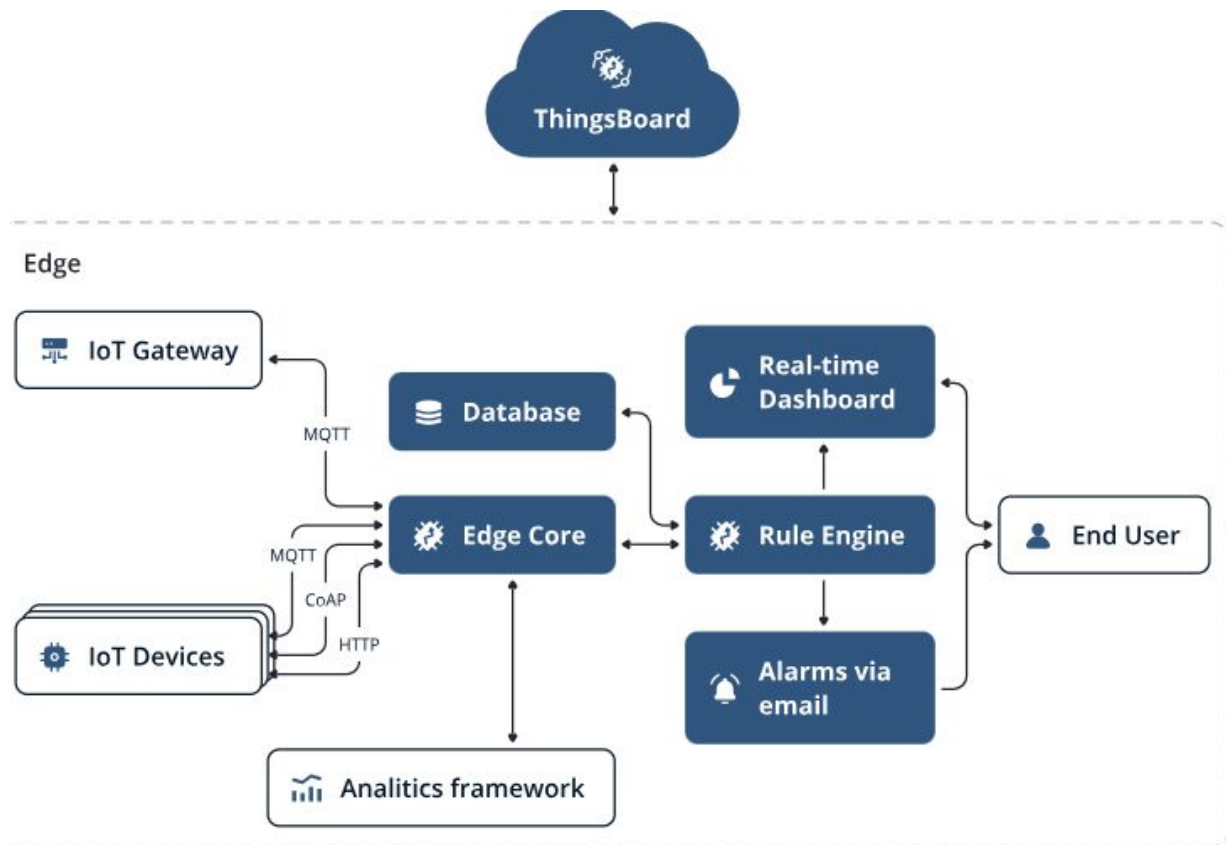
ThingsBoard

Open-source platforma za razvoj IoT aplikacija

Funkcionalnosti koje podržava:

- Povezivanje uređaja
- Prikupljanje i vizuelizacija podataka sa uređaja
- Analiziranje podataka i složeno procesiranje događaja
- Aktiviranje alarma i notifikacija
- Kontrola uređaja korišćenjem RPC-a

Arhitektura Thingsboard sistema



Entiteti

- **Tenant** - biznis entitet, individua ili organizacija koja poseduje *device* ili *asset*. Može da ima više administratora i milione potrošača, uređaja i asset-a.
- **Customer** - biznis entitet, individua ili organizacija koja koristi *devices*, *assets*. Može da ima više korisnika i milione uređaja i asset-a.
- **User** - upravljaju entitetima i komandnim tablama.
- **Device** - IoT entitet koji proizvodi telemetriju i obrađuje *RPC* komande. Npr. senzori, aktuatori, prekidači.
- **Asset** - apstraktni IoT entitet koji može biti povezan sa ostalim uređajima ili *asset*-ima. Npr. fabrika, polje, vozilo.
- **Entity views** - koriste se kada se sa potrošačima deli samo određeni deo uređaja ili asset-a.
- **Alarms** - događaji koji ukazuju na problem ili neuobičajenu situaciju
- **Dashboard** - vizuelizacija IoT podataka
- **Rule node** - čvorovi za obradu poruka i događaja
- **Rule chain** - definiše tok procesiranja

Rule engine

Procesiranje primljenih poruka korišćenjem logike i pravila definisanih od strane korisnika

Komponente:

- Poruka - bilo koji događaj (podaci sa uređaja, REST API događaj, RPC zahtev...)
- Rule Node - funkcija koja se izvršava i obrađuje poruku. Postoje različiti tipovi čvorova (filtriranje, transformisanje, slanje notifikacije)
- Rule Chain - lanac povezanih čvorova

Pokretanje

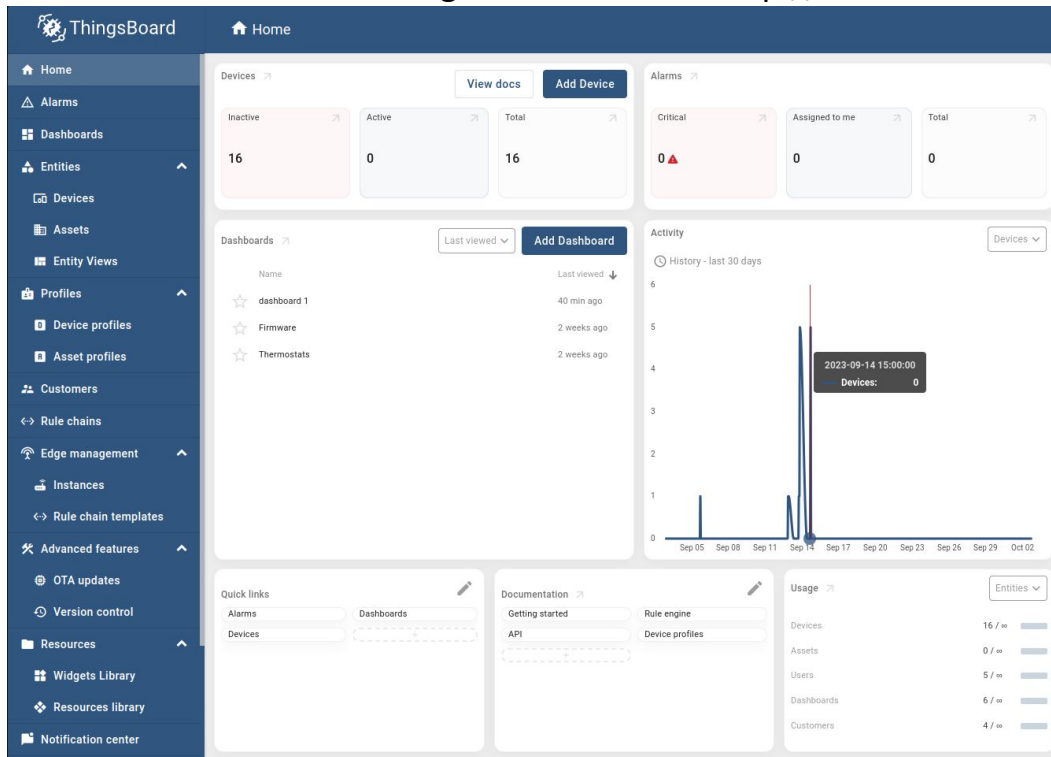
ThingsBoard Cloud server
(community edition)

- Baza: PostgreSQL
- Message service:
Kafka

File:

ThingsBoard/docker-compose.yml

ThingsBoard Cloud UI: <http://localhost:8080>



ThingsBoard Edge

1. Kreiranje edge instance na cloud-u

The screenshot displays the ThingsBoard Edge management interface. The left sidebar contains a navigation menu with the following items: Home, Alarms, Dashboards, Entities (expanded), Devices, Assets, Entity Views, Profiles (expanded), Device profiles, Asset profiles, Customers, Rule chains, Edge management (expanded), Instances (selected), Rule chain templates, Advanced features (expanded), OTA updates, Version control, Resources (expanded), Widgets Library, Resources library, and Notification center.

The main content area shows the 'Edge details' page for an instance named 'rpi'. The breadcrumb navigation at the top reads 'Edge management > Instances > rpi'. The page has a tabbed interface with 'Details' selected, and other tabs include 'Attributes', 'Latest telemetry', 'Alarms', 'Events', 'Downlinks', 'Relations', and 'Audit Logs'. Below the tabs, there are several action buttons: 'Make edge public', 'Assign to customer', 'Manage assets', 'Manage devices', 'Manage entity views', 'Manage dashboards', 'Manage rule chains', and 'Delete edge'. There are also buttons for 'Copy Edge Id', 'Copy Edge key', 'Copy Edge secret', and 'Sync Edge'. An 'Install & Connect Instructions' button is located below these.

The form fields for the edge instance are as follows:

- Name*: rpi
- Edge type*: default
- Edge key: 65c48890-ec3e-c4b3-e64f-3c4b3b649c05
- Edge secret: z234wh6j1b4kj85plq7h
- Label: (empty)
- Description: (empty)

ThingsBoard Edge

Docker kontejneri:

1. postgres
2. mytbedge

CLOUD_ROUTING_KEY - edge key
(generisan na cloud-u)

CLOUD_ROUTING_SECRET - edge
secret (generisan na cloud-u)

CLOUD_RPC_HOST - adresa mašine
na kojoj je pokrenut TB Cloud

```
version: '3.0'
services:
  mytbedge:
    restart: always
    image: "thingsboard/tb-edge:3.5.1.1EDGE"
    ports:
      - "18080:8080"
      - "11883:1883"
      - "15683-15688:5683-5688/udp"
    environment:
      SPRING_DATASOURCE_URL: jdbc:postgresql://postgres:5432/tb-edge
      CLOUD_ROUTING_KEY: ${EDGE_KEY}
      CLOUD_ROUTING_SECRET: ${EDGE_SECRET}
      CLOUD_RPC_HOST: ${CLOUD_IP}
    volumes:
      - ~/.mytb-edge-data:/data
      - ~/.mytb-edge-logs:/var/log/tb-edge
  postgres:
    restart: always
    image: "postgres:12"
    ports:
      - "5432"
    environment:
      POSTGRES_DB: tb-edge
      POSTGRES_PASSWORD: postgres
    volumes:
      - ~/.mytb-edge-data/db:/var/lib/postgresql/data
```

Uređaji na TB Edge-u

Devices Device Filter								+ ↺ 🔍	
<input type="checkbox"/>	Created time ↓	Name	Device profile	Label	State	Customer	Public	Is gateway	
<input type="checkbox"/>	2023-09-13 19:59:13	proximity-sensor	default		Inactive		<input type="checkbox"/>	<input type="checkbox"/>	🔗 👤 ⚙️ ↶️ 🛡️ 🗑️
<input type="checkbox"/>	2023-09-13 19:57:44	imu	default		Inactive		<input type="checkbox"/>	<input type="checkbox"/>	🔗 👤 ⚙️ ↶️ 🛡️ 🗑️
<input type="checkbox"/>	2023-09-13 19:50:28	pressure-sensor	pressure		Inactive		<input type="checkbox"/>	<input type="checkbox"/>	🔗 👤 ⚙️ ↶️ 🛡️ 🗑️
<input type="checkbox"/>	2023-09-13 19:48:40	thermostat	temperature		Inactive	Public	<input checked="" type="checkbox"/>	<input type="checkbox"/>	🔗 👤 ⚙️ ↶️ 🛡️ 🗑️

- Uređaji na edge-u su kreirani za odgovarajuće senzore na Arduinu

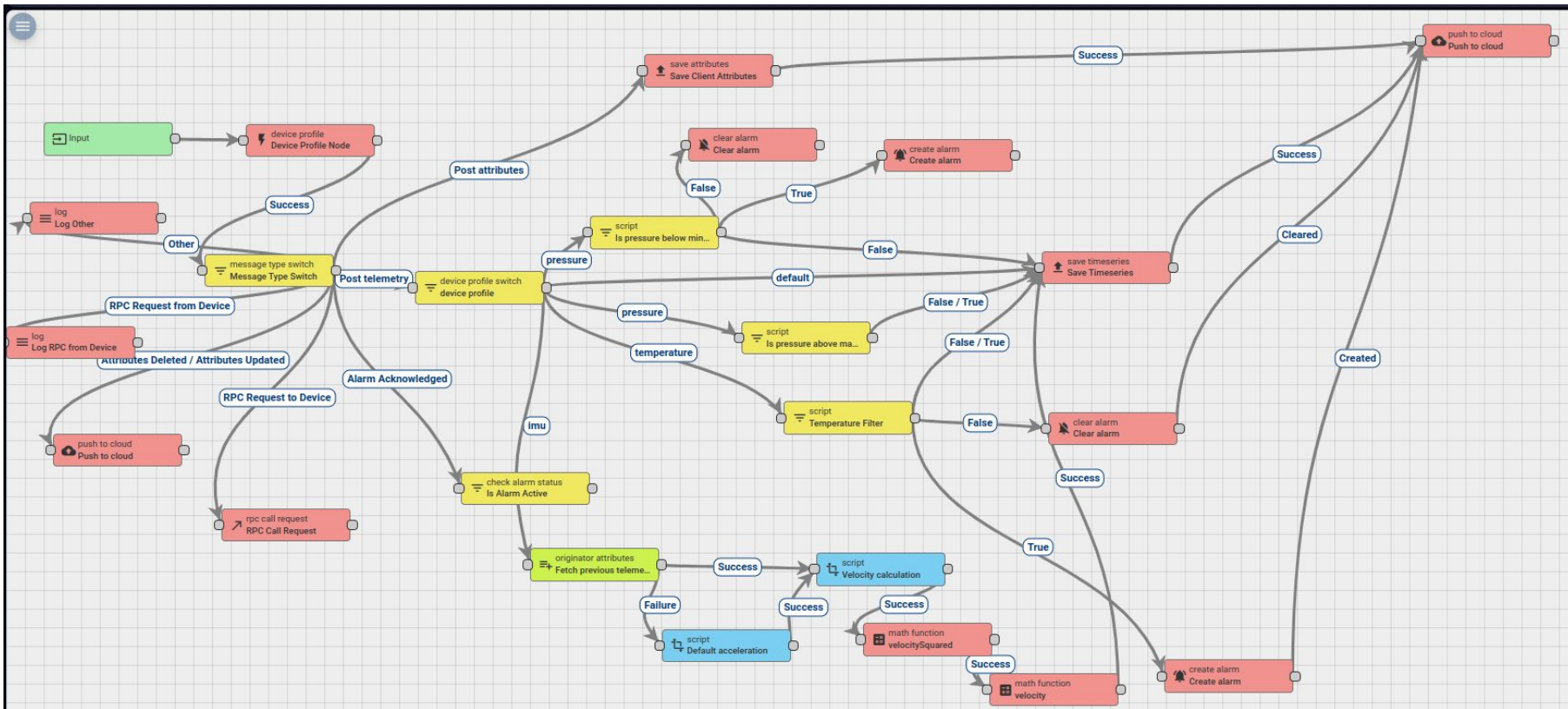
Slanje podataka sa senzora

- Sa Arduino Board-a se očitavaju podaci sa senzora koji se šalju na serijski port
- Python aplikacija čita podatke sa serijskog porta i šalje ih preko MQTT-a edge platformi
- Top-ic na koji se šalju podaci:

`v1/devices/me/telemetry`

- Token koji se generiše prilikom kreiranja uređaja na TB platformi se koristi kao username prilikom slanja poruke

Rule Chain



Filter čvorovi

Is pressure below minimum

Filter - script

Details

Events

Help

Name*

Is pressure below minimum

TBEL

Java Script

```
function Filter(msg, metadata, msgType) {
```

```
1 return msg.pressure < 90;
```

Temperature Filter

Filter - script

Details

Events

Help

Name*

Temperature Filter

TBEL

Java Script

```
function Filter(msg, metadata, msgType) {
```

```
1 return (msg.temperature != null && msg.temperature >= 30);
```


Čvor za uzimanje prethodnih vrednosti podataka iz baze

Fetch previous telemetry

Enrichment - originator attributes

[Details](#)[Events](#)[Help](#)

Name*

Fetch previous telemetry

☒ Tell Failure

If at least one selected key doesn't exist the outbound message will report "Failure".

Fetch into

☐ Data☒ Metadata

Client attribute keys

Hint: use `$(metadataKey)` for value from metadata, `$(messageKey)` for value from message body

Shared attribute keys

Hint: use `$(metadataKey)` for value from metadata, `$(messageKey)` for value from message body

Server attribute keys

Hint: use `$(metadataKey)` for value from metadata, `$(messageKey)` for value from message body

Latest time-series data keys

xAcc ×

yAcc ×

zAcc ×

timestamp ×

Hint: use `$(metadataKey)` for value from metadata, `$(messageKey)` for value from message body

☒ Fetch timestamp for the latest telemetry values

If selected, the latest telemetry values will also include timestamp, e.g: "temp": {"ts":1574329385897, "value":42}"

Description

Čvor koji izvršava transformaciju podataka

Velocity calculation

Transformation - script

Details Events Help

Name*
Velocity calculation

TBEL Java Script

```
function Transform(msg, metadata, msgType) {  
  1 var deltaTime = (msg.timestamp - metadata.timestamp) / 1000;  
  2 msg.velocityX = 0.5 * (msg.xAcc + parseInt(metadata.xAcc)) * deltaTime;  
  3 msg.velocityY = 0.5 * (msg.yAcc + parseInt(metadata.yAcc)) * deltaTime;  
  4 msg.velocityZ = 0.5 * (msg.zAcc + parseInt(metadata.zAcc)) * deltaTime;  
  5  
  6 return {  
  7     msg: msg,  
  8     metadata: metadata,  
  9     msgType: msgType  
 10 };  
}
```

Test transformer function

Description

Čvor koji izvršava matematičku funkciju

velocitySquared
Action - math function

DetailsEventsHelp

Name*
velocitySquared

Functions*
CUSTOM | Custom Function

Arguments

=

x.

Type*
Message body

Key*
velocityX

Default value
0

=

y.

Type*
Message body

Key*
velocityY

Default value
0

=

z.

Type*
Message body

Key*
velocityZ

Default value

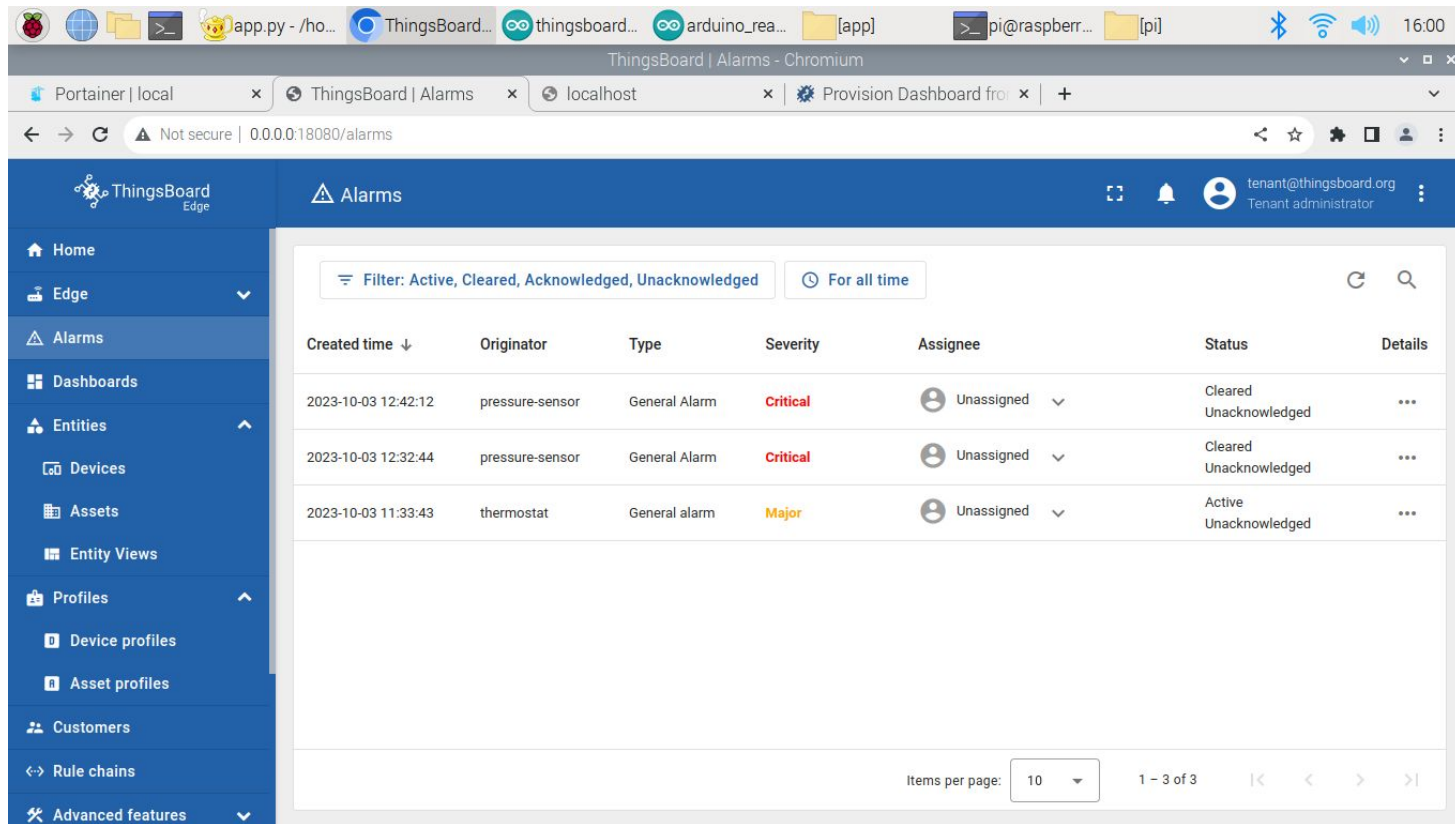
+ Add

Mathematical Expression *

$$x * x + y * y + z * z$$

Hint: specify a mathematical expression to evaluate. For example, transform Fahrenheit to Celsius using $(x - 32) / 1.8$

Prikaz alarma

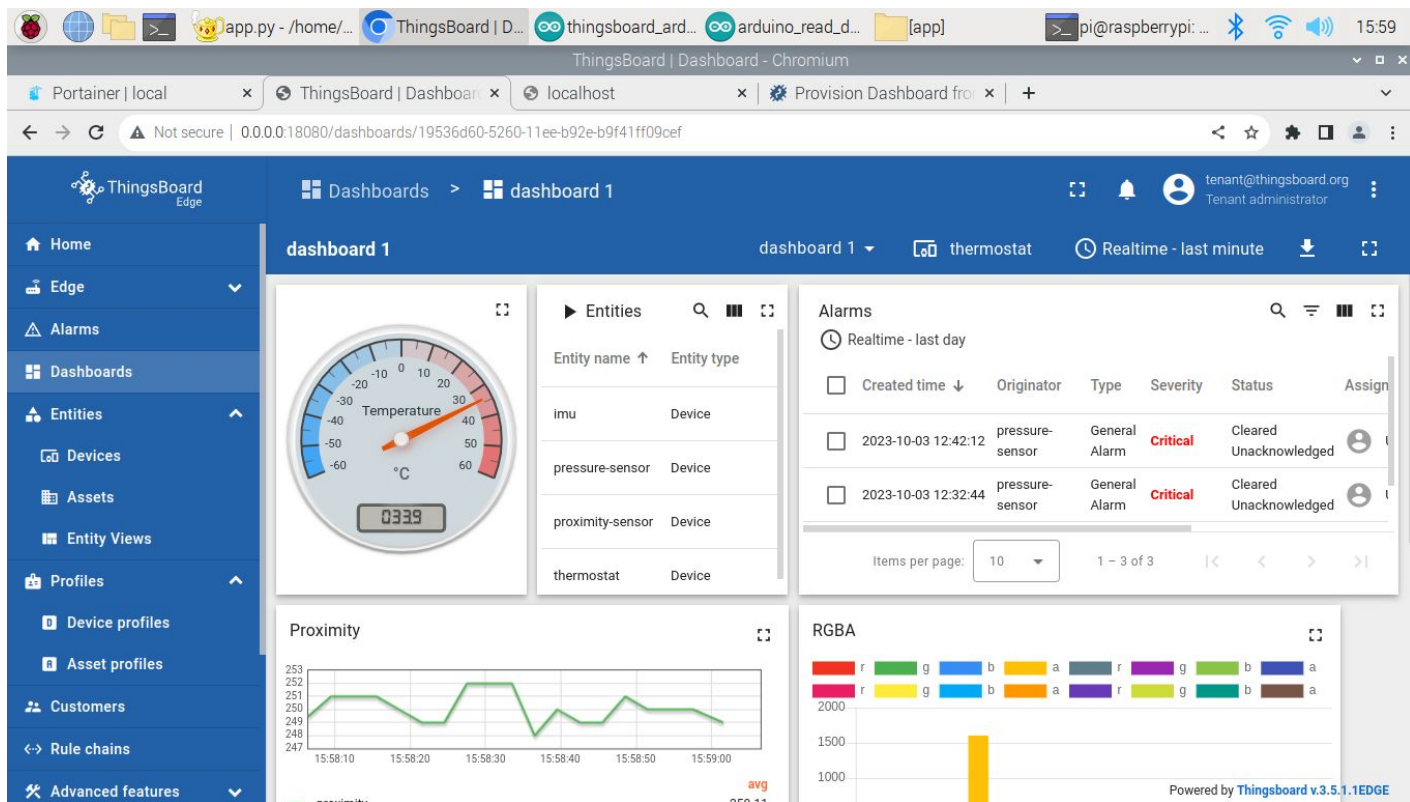


The screenshot shows the ThingsBoard Alarms page in a Chromium browser. The page title is "ThingsBoard | Alarms". The left sidebar contains navigation links: Home, Edge, Alarms (selected), Dashboards, Entities, Devices, Assets, Entity Views, Profiles, Device profiles, Asset profiles, Customers, Rule chains, and Advanced features. The main content area displays a table of alarms with the following columns: Created time, Originator, Type, Severity, Assignee, Status, and Details. The table contains three rows of data, all with a status of "Cleared Unacknowledged".

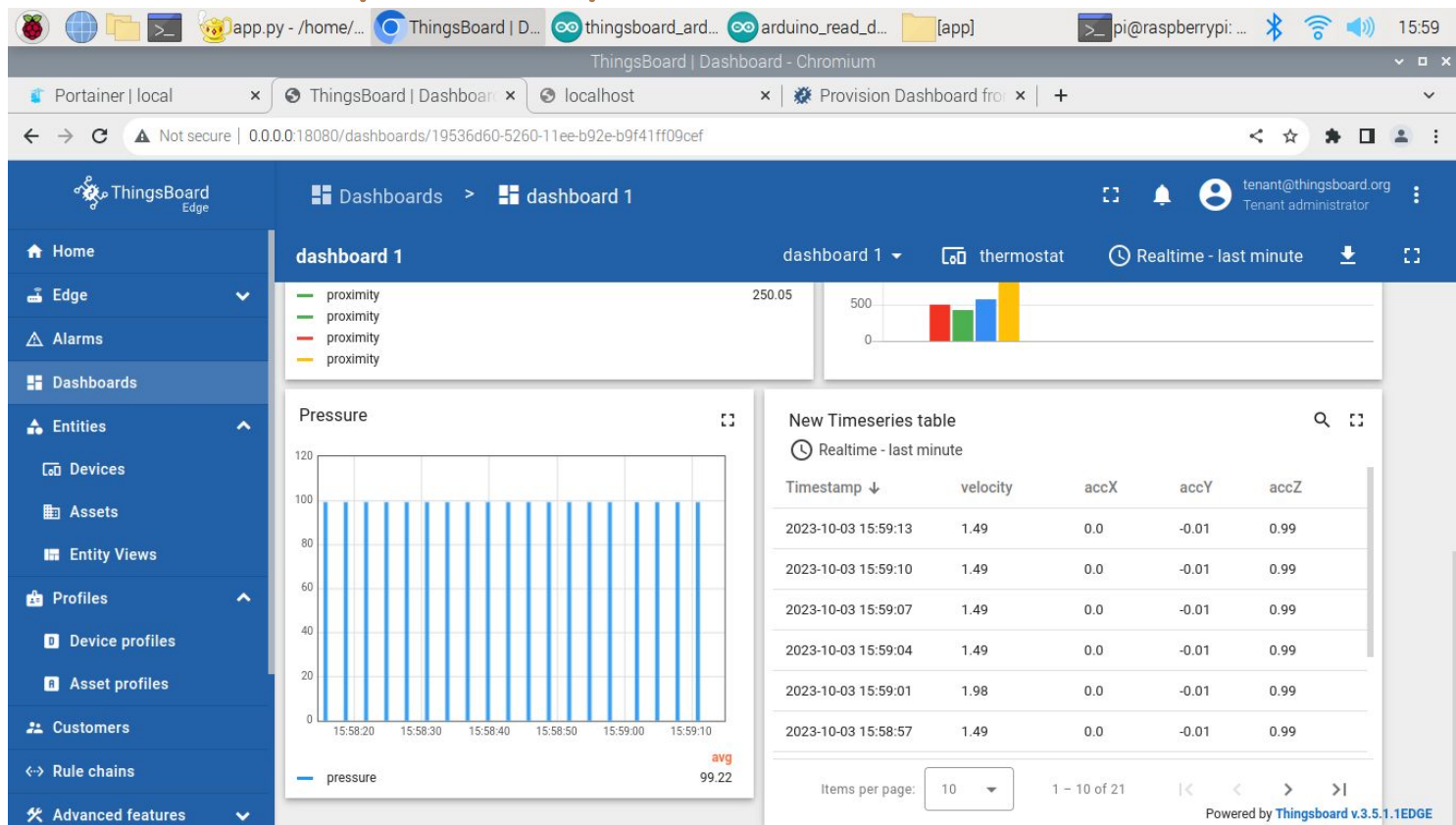
Created time ↓	Originator	Type	Severity	Assignee	Status	Details
2023-10-03 12:42:12	pressure-sensor	General Alarm	Critical	Unassigned	Cleared Unacknowledged	...
2023-10-03 12:32:44	pressure-sensor	General Alarm	Critical	Unassigned	Cleared Unacknowledged	...
2023-10-03 11:33:43	thermostat	General alarm	Major	Unassigned	Active Unacknowledged	...

At the bottom of the table, there is a pagination control showing "Items per page: 10" and "1 - 3 of 3".

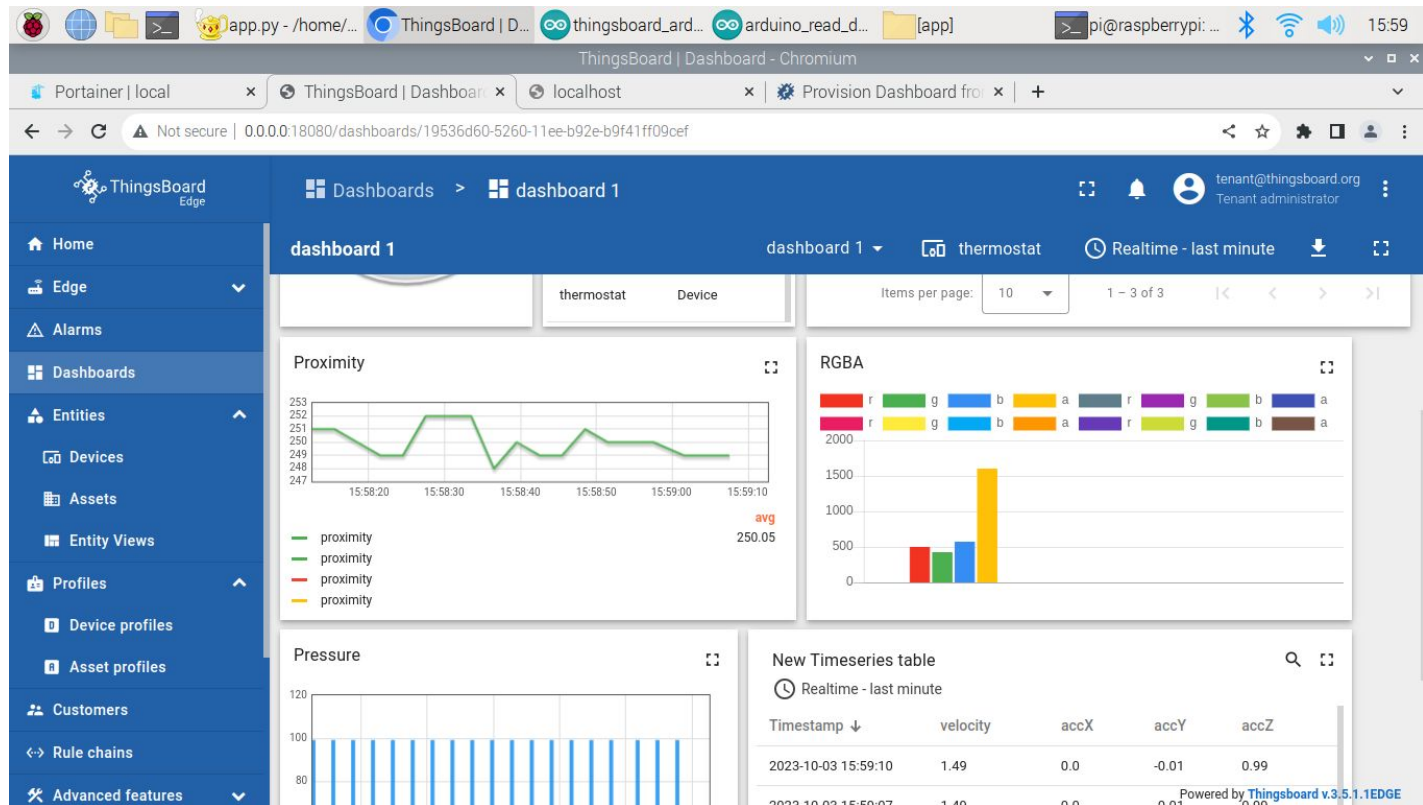
Dashboard - vizuelizacija podataka



Dashboard - prikaz pritiska i brzine



Dashboard - proximity i RGBA vrednosti





Projekti 2 i 3

IoT Sistem za praćenje i kontrolu pametne kuće

IoT sistem za praćenje i kontrolu HVAC sistema, osvetljenja i sigurnosnog sistema unutar pametne kuće.

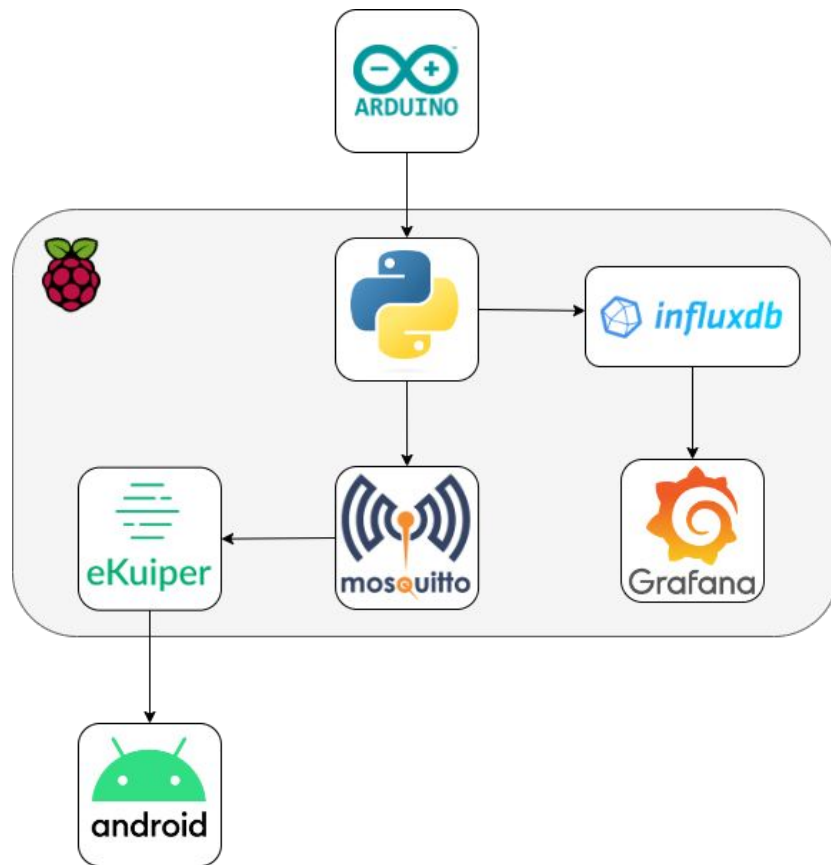
Izvor podataka: Arduino Nano 33 BLE Sense Lite

Raspberry Pi 4 se koristi za izvršavanje Docker mikroservisa.

Praćenje stanja sistema je dostupno preko Android mobilne aplikacije.

Arhitektura sistema

- **Mqtt** - Mosquitto message broker
- **InfluxDB** - baza za skladištenje očitanih podataka
- **Grafana** - vizuelizacija podataka
- **Ekuiper** - analiza i detekcija događaja
- **Ekuiper Manager** - softver za upravljanje Ekuiper-om
- **IoT app**



IoT app

- Python mikroservis
- Podaci se čitaju sa serijskog porta Arduino board-a
- Pročitani podaci se šalju na odgovarajući MQTT topic (paho mqtt client)
- Svi podaci se čuvaju u Influx bazi podataka (InfluxDB client)

eKuiper

Za svaki izvor podataka je kreiran stream na eKuiper-u.

Ukupno postoje 3 stream-a, za svaki senzor sa Arduina:

- `lps_data`
- `imu_data`
- `apds_data`

Na eKuiper-u su kreirana pravila za analizu primljenih podataka i detekciju događaja. Rezultati pravila se šalju na određeni MQTT topic.

Primer eKuiper pravila

- Detekcija niske temperature

Rules / View

Rule ID

low_temperature

Name

SQL

```
1 SELECT avg(temperature) FROM lps_data GROUP BY TUMBLINGWINDOW(ss, 10) HAVING avg(temperature) < 27.5;
2
```

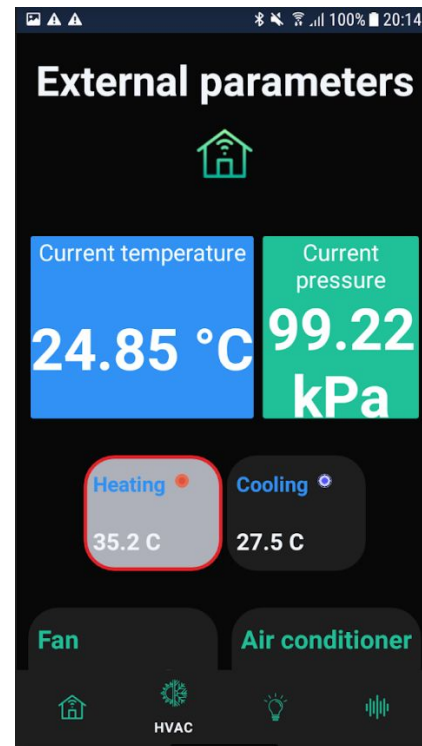
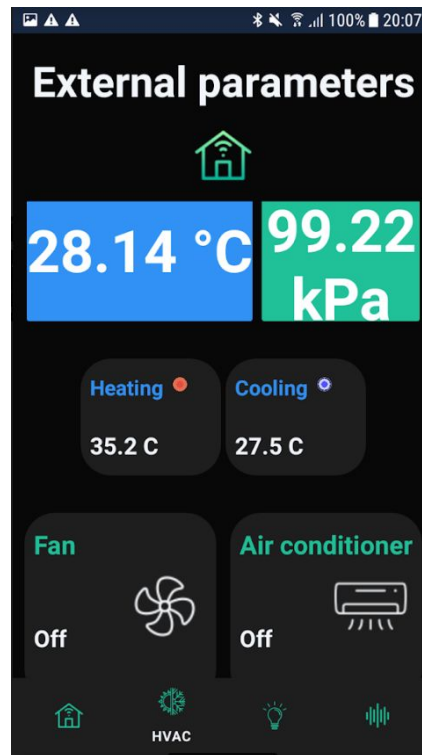
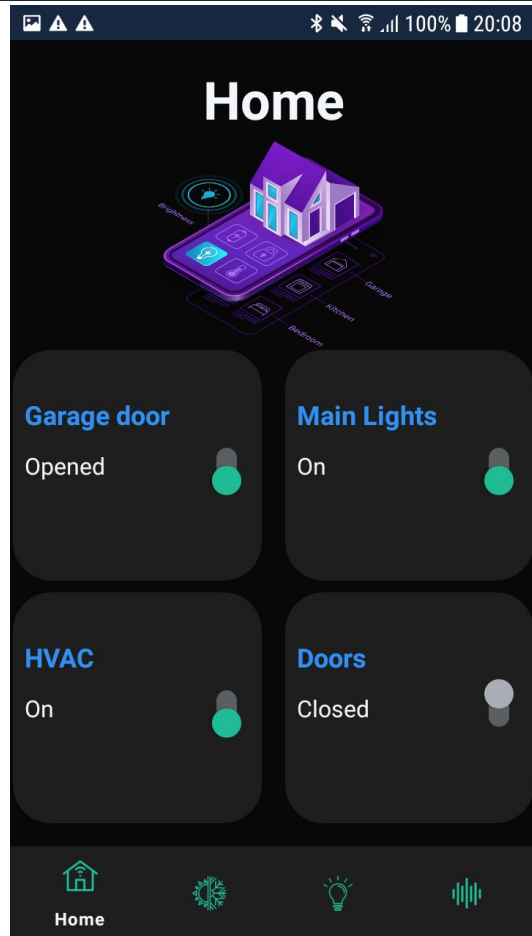
Actions

Sink	Operations
mqtt	View

Android aplikacija

- Za praćenje stanja sistema je implementirana Android aplikacija
- U okviru aplikacije je implementiran MQTT klijent, kako bi se čitali podaci sa topic-a
- Android uređaj se sa Arduino uređajem povezuje preko Bluetooth Low Energy
- Iz mobilne aplikacije se akcije šalju komande za aktiviranje akcija na Arduino

Android aplikacija



HVAC system control

BLE komunikacija

Android:

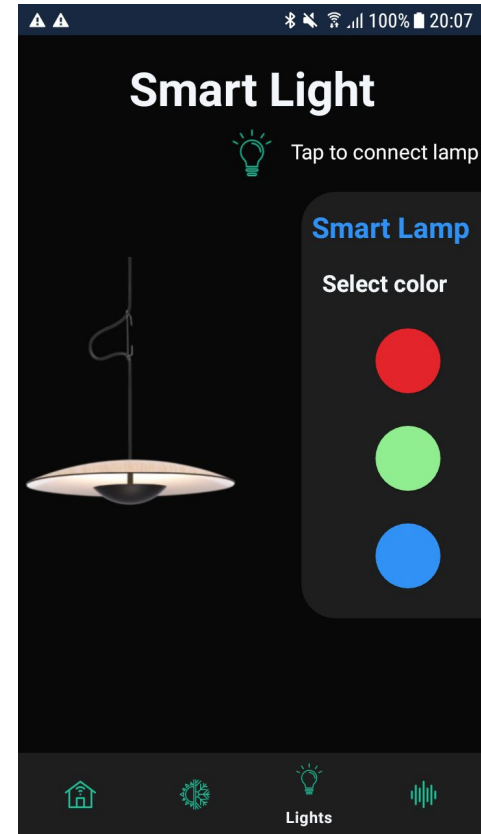
- BLE klijent za čitanje i upisivanje karakteristika
- Upisivanjem vrednosti karakteristike se šalje komanda Arduino uređaju

Arduino:

- Konekcija sa perifernim uređajima
- Dodavanje servisa i karakteristika
- Čitanjem karakteristike se aktivira RGB dioda

BLE

Nakon izbora boje, na Arduinou svetli RGB dioda odgovarajućom bojom



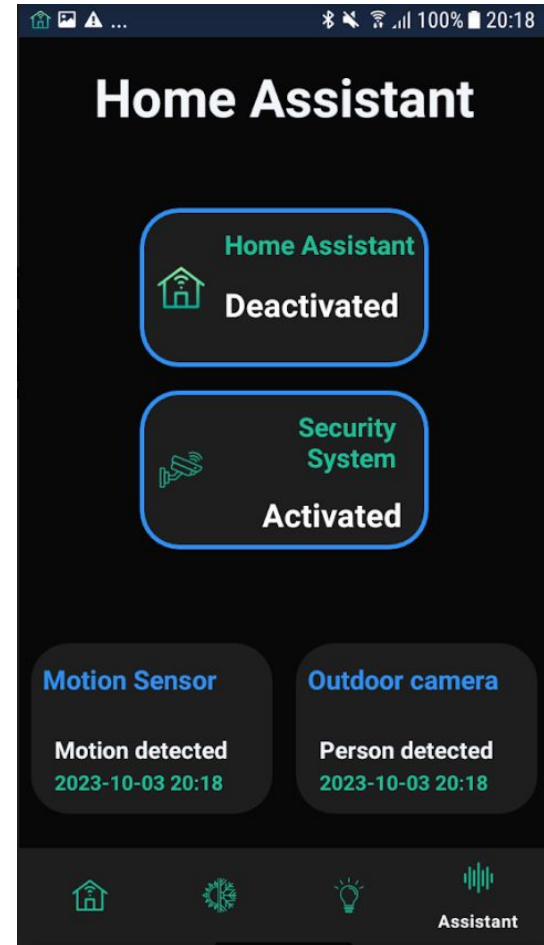
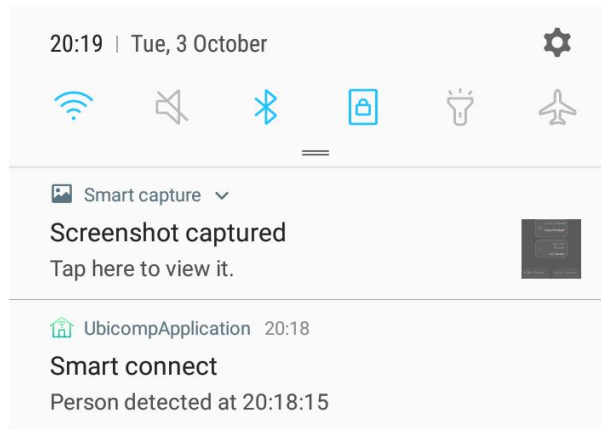
TensorFlow Lite

- Tensorflow model za klasifikaciju slika
- Klasifikacija u 3 kategorije: pas, mačka i osoba.
- Inception v3 model
- Model je konvertovan u *tflite* model kako bi se izvršavao na Raspberry Pi uređaju
- *run_detection.py* skripta klasifikuje sliku i ukoliko je detektovana osoba, poruka se šalje na MQTT topic

Pokretanje skripte:

```
python3 run_detection.py <image_file>
```

Obaveštenje o detekciji osobe u Android aplikaciji



Hvala na pažnji