Unige Crowd Detector

Tiziano Firpo - S 4379124
Andrea Canepa - S 4185248

Computer Science - IoT Final Project - A.Y. 2019/2020

Overview

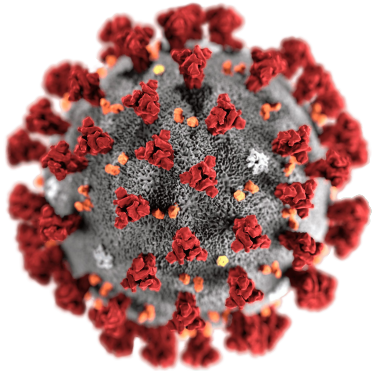
1. Introduction
2. Software and Hardware Architecture
3. API
4. Web App
5. Sensor
6. Bot Telegram
7. DEMO TIME
8. Future Development

Introduction

Introduction: problem

- ***COVID-19*** has imposed us some restrictions in our everyday life, i.e. **social distancing**
- "*Immuni*" app can trace our contacts with others but we can **prevent** in advance the gathering of people

Indoor crowd detection is the solution!

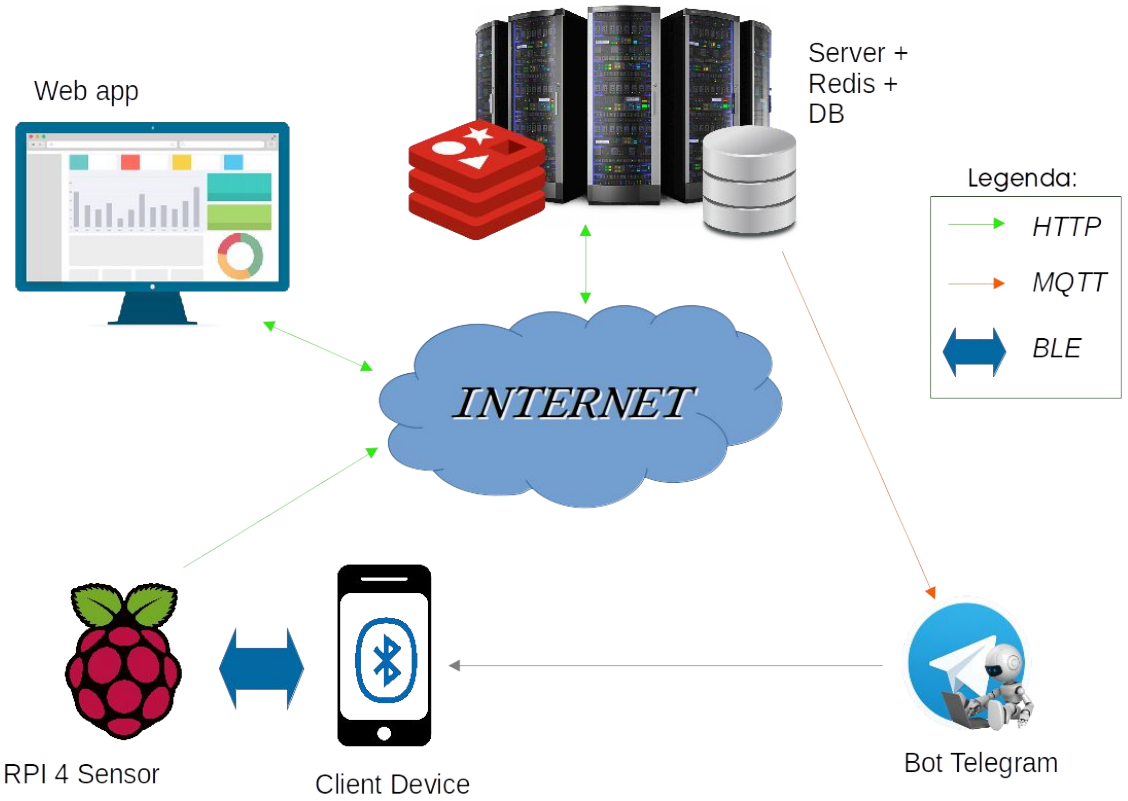


Introduction: Assumptions

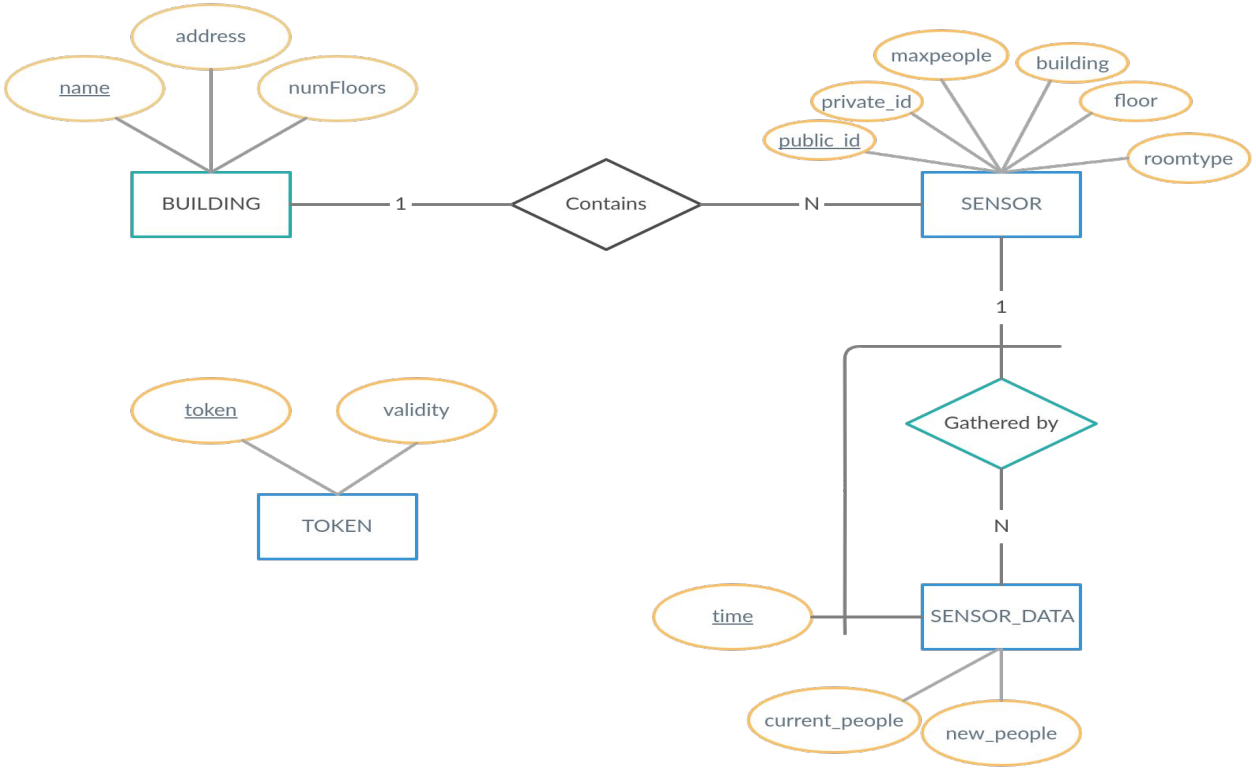
- Each individual has to turn on **bluetooth** on his/her own device
- Sensors should be well-distributed over the buildings, otherwise data can be duplicated
- There must be a **system administrator** that is able to configure and to maintain all the infrastructure
- People have to adhere to the **Telegram Bot** in order to receive real-time data from the infrastructure

Software and Hardware Architecture

System Architecture



Database ER Schema

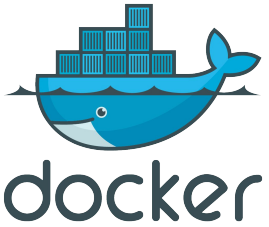


API

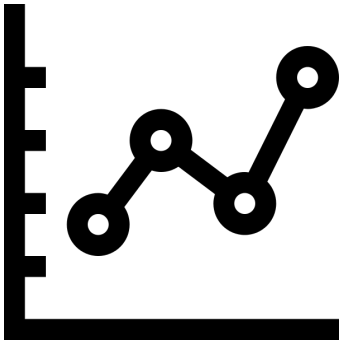
API pt. I

- The server main role is to **receive, store** and **process** sensors data
- Reachable through a **REST API**
- Endpoints are exposed using `Express.js`
- Rely on a *Postgres DB* and its connection is managed with `Knex.js`
- `Redis` is used to speed up some operations, avoiding to stress the DB

API pt. II



- Entirely ***Dockerized*** for easier deploys
- Offer *statistics* about the number of *new* and *current* people taken by a sensor during:
  - The same day
  - The day before
  - Last week
  - Last month

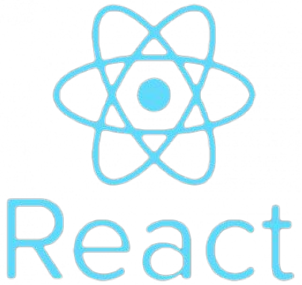


API pt. III

- Exposes an ***MQTT*** server over ***WebSocket***
- In case of alerts a message in the `ALERTS` topic will be sent, providing the following information:
  - *Building* where the sensor is placed
  - *Floor*
  - Room *type*
  - *Capacity* of the room
  - Number of *current* people
  - Other sensor's information

Web App

Web App pt. I: description

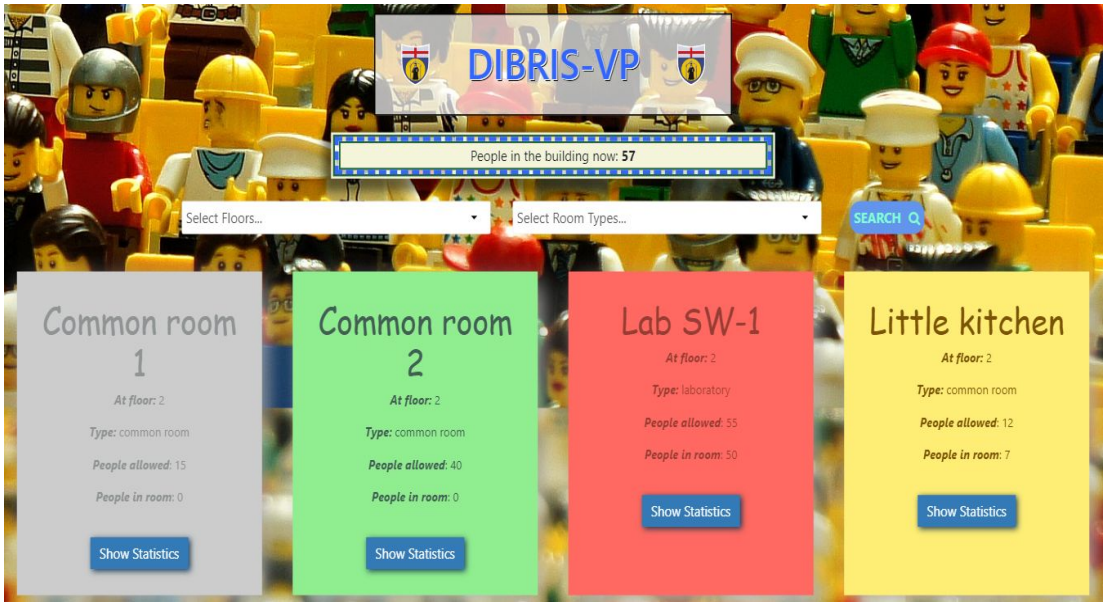


- Entirely developed with `React.js` framework
- ***Responsive*** web app to show well-formatted data also from mobile phone
- 3 pages: `index.js`, `buildings.js` and `stats.js`
- ***Sensor*** page updated every 5 *minutes*
- Easy-to-use site and intuitive front-end to drive the attention on ***data***
- *Stateless* and *userless* environment

Web App pt. II: `index.js`

- *Homepage* of the web app
- **Filter** *buildings* by *name*
- Click on “*Enter*” button to switch to the *building* page that shows sensors into it

Web App pt. III: `buildings.js`

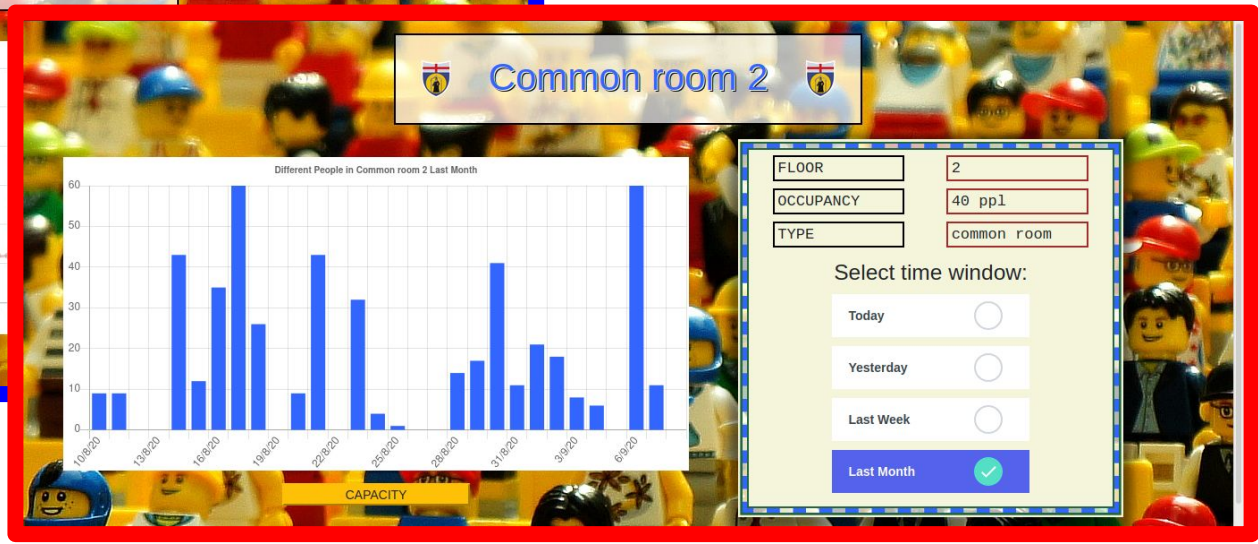


- Filter sensors by ***floor*** and ***type of room***
- Color-code to report ***crowding:***
  - 🟥: very crowded
  - 🟨: start gathering
  - 🟩: situation ok
  - ⬜: sensor idle for more than 30 mins.

Web App pt. IV.a: `stats.js`

- Show crowd statistics in different time ranges: **today**, ***yesterday***, ***last week*** and ***last month***
- Draw meaningful charts with `Charts.js`
- ***Tooltips*** on hover
- *Last** options offer also a ***DISTINCT*** button to show how many different people entered in the area
- *Real-time reload* of the ***chart*** when the user change *time range*

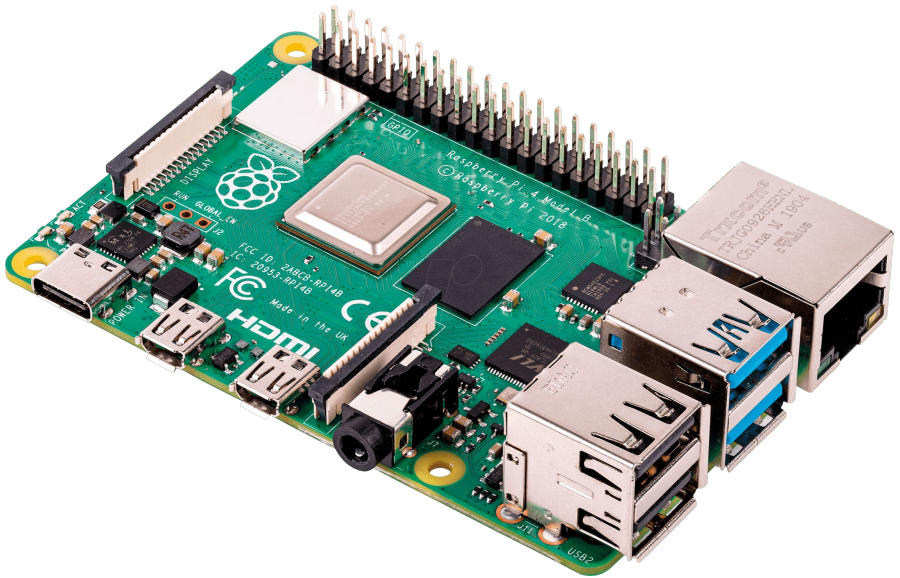
Web App pt. IV.b: `stats.js`



Sensor

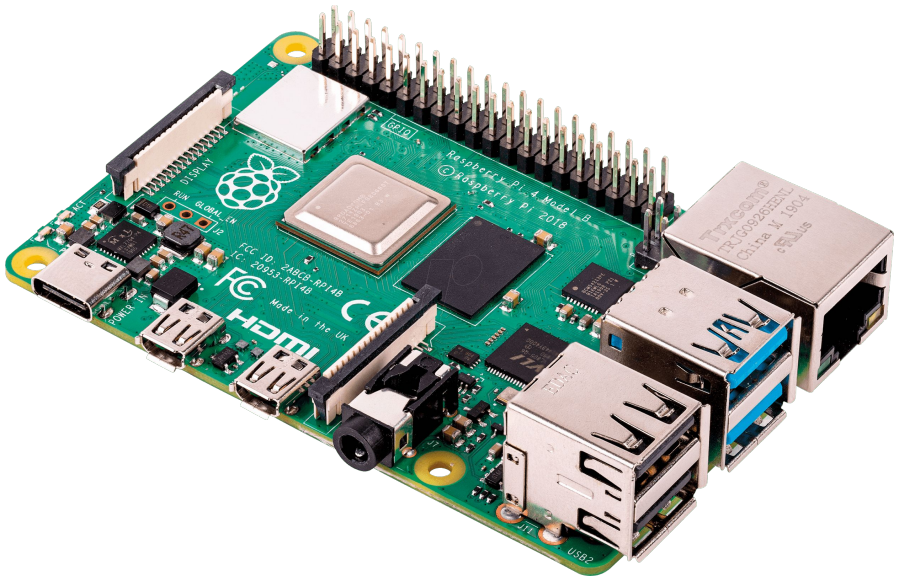
Sensor pt. I

- Code in *`sensor/sensor.js`*
- Configuration through script ***configure_rpi.sh***
- Detects people devices nearby and sends those data to the server
- Relies on **bluetooth**:
  - *`@abandonware/noble`*
  - *`node-bluetooth`*



Sensor pt. II

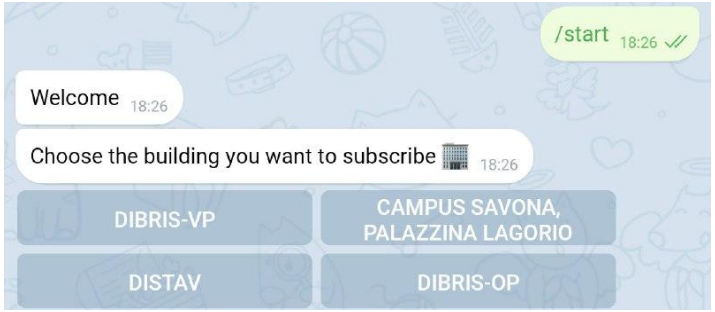
- Keep tracks of the *crowd* and also of the *newcomers* that arrives in the surveilled area
- ***CRON***: Everyday at 12:00 AM resets the data gathered during the day
- ***.env*** configuration file
- `DIBRIS_simulator.js`: fill the DB with sample data



Bot Telegram

Telegram Bot

- ***Real-time*** alerts in a user-friendly way
- It relies on the **MQTT** broker exposed by the server
- Each user can *subscribe* to a specific *building*
- Once subscribed, the user will receive alerts from it
- When an alert in the topic ***ALERTS*** is received, a message to all the subscribers will be sent

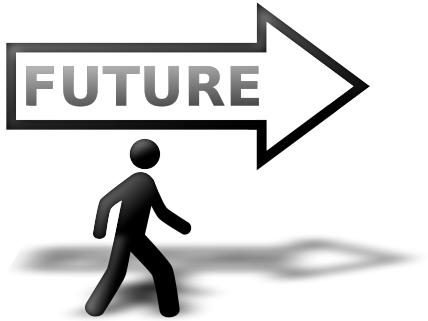


DEMO TIME

Code in a nutshell

Future Developments

And now ?



In order of importance, for a better use:

1. **Admin panel** to perform insertions and deletions of sensors and buildings
2. **User** identity and correspondent **login form**
3. **Favorites** for both sensors and buildings
4. **RFID** devices for the students
5. Mobile phone **APP** to ease the access to the data

Thank You for your attention !