PSOT - Public Space Occupancy Tracking

IoT Project Final Presentation

APPLICATION

PSOT is a system that monitors the occupancy of certain public spaces through the collection of unique MAC addresses using Raspberry Pis spread out in those spaces.

The information would then be available through a web dashboard and sent through a message queue. This way people can quickly choose places to study, have lunch, etc.

ABOUT THE PROJECT

Nowadays, it is common for a person to carry a device with connectivity capabilities. These devices usually have associated with them a MAC address that is supposed to be unique.

If we run on the assumption mentioned above, we can measure the amount of people in a space by counting the amount of devices in the vicinity using their MAC addresses.

PLANNED PROOF OF CONCEPT

- Single Raspberry Pi scanning and collecting MAC Addresses around itself, through bluetooth and Wi-Fi
- A database collecting the scan data
- A periodic algorithm that gathers the scan data from the database and calculates the occupancy of each space
- A very simple dashboard that displays the current occupancy of each space, updating it in real time

REALISED PROOF OF CONCEPT

The realised proof of concept contains all elements predicted in the plan, however:

- The Raspberry Pi only collects MAC addresses through Wi-Fi
- As it is more uncommon for devices to communicate through bluetooth making it uncommon to catch their MAC address
- So, for a proof of concept it wasn't worth the effort

ARCHITECTURE

The system architecture is composed of the following components:

A Raspberry Pi fleet **01**

An ingest service **02**

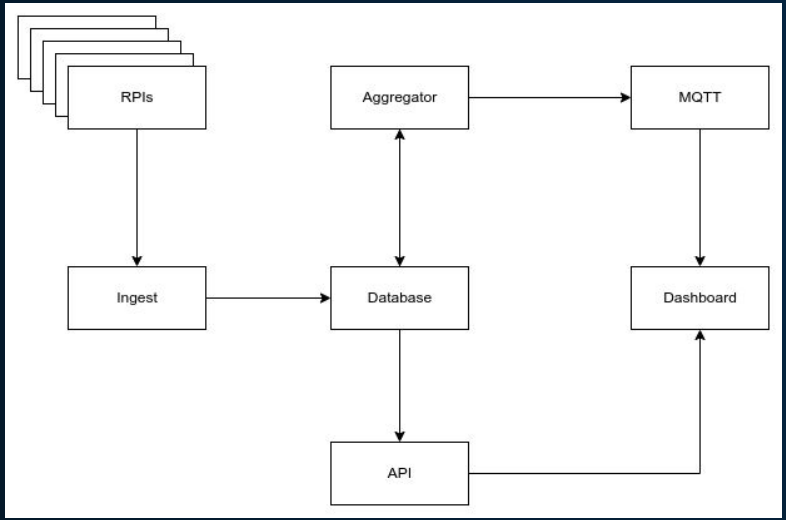
A Database **03**

04 An aggregator service

05 An API

06 A dashboard

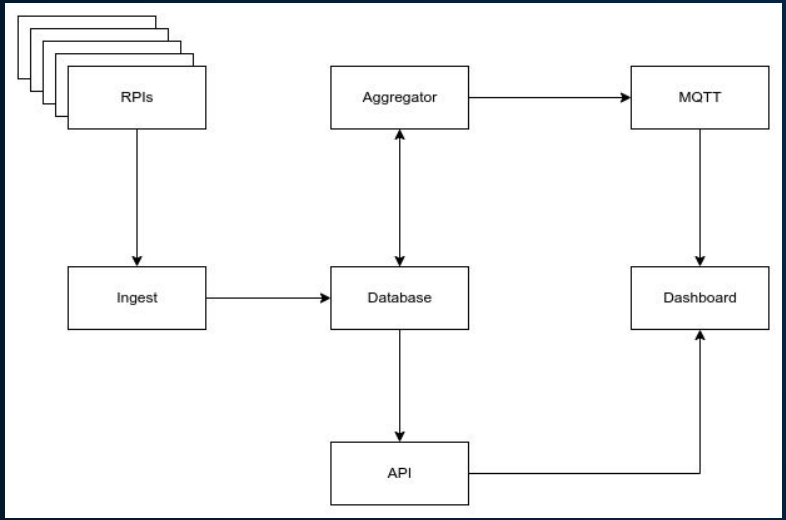
07 An MQTT broker



ARCHITECTURE

The Raspberry Pis/Scanners scan surrounding MAC addresses and send them to the Ingest service through HTTP over IP. The data comes structure in JSON objects.

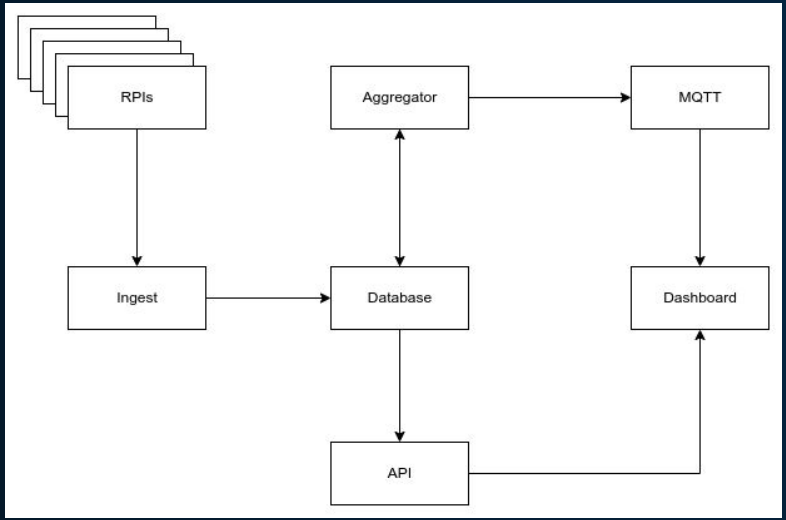
The Ingest service's function is to receive the scan data from the Scanners and insert it into the database. It exists in the middle to ensure good decoupling of the Scanners and database. This way we can swap the database implementation without having to change the software on the scanners as well.



ARCHITECTURE

The Aggregator's function is to periodically collect the most recent scan data and using it to estimate the occupancy level of the spaces.

Once those are calculated it publishes the values to the database as well as send them through a message queue to notify any interested parties. It also clears all old scan data for privacy reasons.



ARCHITECTURE

Finally, the Dashboard displays the occupancy of each space through a simple list containing badges informing if the space is free, occupied or full. For each place it also provides a chart with the readings from the last 6 hours.

The data is provided by an API, as well as a message queue to ensure there is no need to reload the page.

AGGREGATOR ALGORITHM

One of the main challenges of the aggregator is that it needs to use data obtained from different RPIs and determine which of the people are actually in the space, as some MAC addresses are only of people passing by. For this, the aggregator applies the following algorithm:

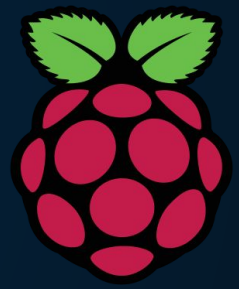
- Pick the scans done more recently;
- Organize the scans by the space they survey;
- Sort the scans by the time they were done;
- Arrange the scans into time frames;
- Enumerate the MAC addresses detected in each of the time frames;
- Count the amount of times each MAC address appears in a different time frame;
- If this number reaches a certain threshold, the MAC address is assumed to be from a device that is occupying space at the scanned location.

How recent the scans are, as well as the size of the timeframes and threshold can be changed.

DECISIONS

Raspberry PI:

- The RPI was chosen as it was relatively simple to work with.
- Since it is essentially a small computer, it is easy to write software for it.
- Also, since it can run docker, the setup needed to get the scanner code running was minimal.



DECISIONS

HTTP over IP and JSON between RPIs and Ingest:

- Ingest is a web server running on the cloud, so HTTP made a lot of sense from the beginning.
- Since the Raspberry Pi is basically a small computer, it also had all the capabilities needed to access the Internet, making this choice the simpler course of action.
- JSON is also a simple format that is used for communication over the web, and many languages already implement a parser and encoder for it.

DECISIONS

Central processing of the scan data:

- The data is processed centrally by a single service.
- This is done because each space can have more than one Raspberry PI scanning for MAC addresses, which would greatly increase the complexity of the data processing as we would need to have the raspberry pis sharing information with each other.
- It also make changes to the estimation algorithm easier since changing software running on the cloud is simpler than changing software running on machines distributed geographically.

DECISIONS

Use of MQTT for occupancy notifications:

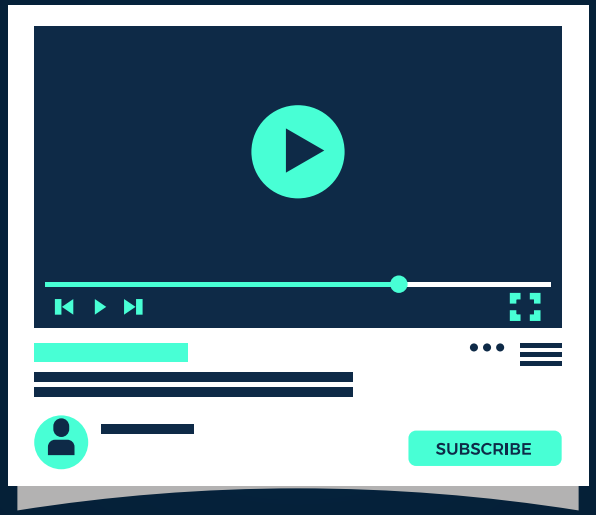
- In a more complete and production ready version of the system, there could devices physically deployed around spaces that display the occupancy level of the spaces, as well as users who choose to be notified when a certain space becomes available.

- Using an MQTT broker to distribute these notifications makes it super simple for future consumers to plug into the data.

CHALLENGES

- We can't force devices to give us their MAC address, this means we have to scan for long periods of time. This won't guarantee that the scanners find every MAC address, however, so the aggregator has to consider multiple scans over a period of time.
- Another point is since the scanners are only scanning for Wi-Fi packets, we can only detect devices if they are connected the a Wi-Fi access point, making this system only suitable for spaces with Wi-Fi networks that many devices are connected to.
- Detecting only MAC addresses isn't enough, since one person can have more than one device, like a laptop and phone, and there are devices that don't belong to anyone, like printers.
- To counteract this, there must be many calibration parameters, like scaling constants to adjust the number of detected mac addresses or MAC address blacklists to filter out devices we know are not associated with people.

VIDEO DEMO



https://drive.google.com/file/d/1SfzJ2hdXoxZzysMr1iBNw1JLm-vEdrLt/view?usp=sharing

WHAT WE LEARNED

- Working with the Raspberry PI, mostly on setting up it's configurations so it needs the least possible setting up to get the scanner running. We learned about editing the network files on the OS and enable startup services like ssh to make it easier to connect to the device to work on it.

- We also learned about creating a data processing system with many services that collect data and apply algorithms over it.

- Finally, we learned of the challenges that come with this subject of measuring the occupancy of spaces through the scanning and detection of devices.

SELF-EVALUATION

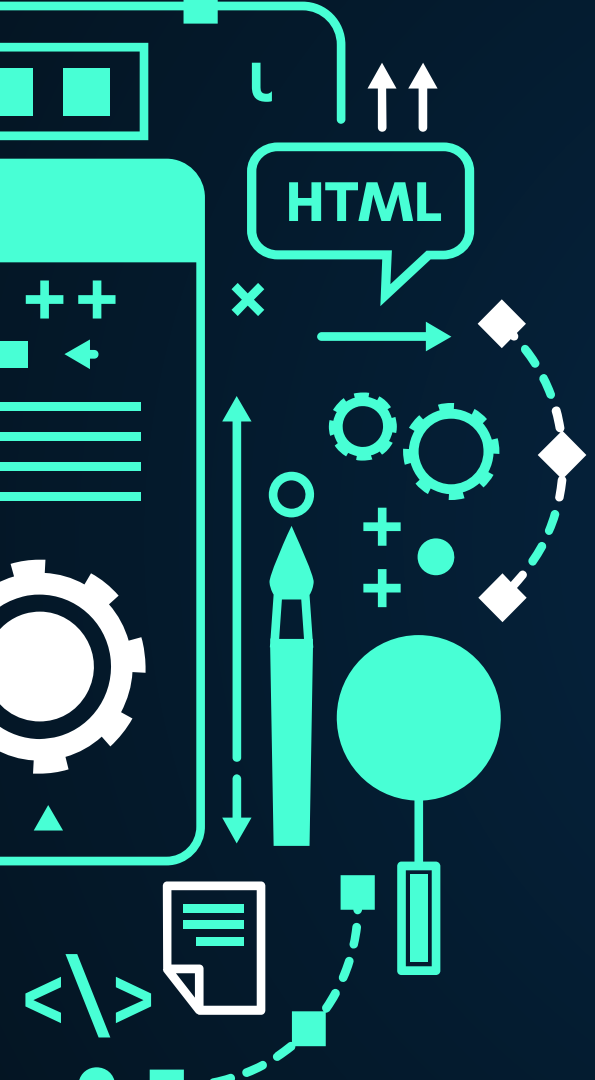
GRADE PROPOSAL: 15

DIOGO PEREIRA
20%

JOÃO PINTO
20%

MIGUEL PIRES
20%

MOISÉS ROCHA
40%



THANK YOU!

Does anyone have any question?

Group 8:
Diogo Pereira - 201505318
João Pinto - 201705547
Miguel Pires - 201406989
Moisés Rocha - 201707329

CREDITS

This is where you give credit to the ones who are part of this project.

- Presentation template by Slidesgo
- Icons by Flaticon
- Infographics by Freepik
- Images created by Freepik
- Author introduction slide photo created by Freepik
- Text & Image slide photo created by Freepik