What's That Smell?

Detecting Air Quality with Python, Raspberry Pi, and Redis

Justin Castilla

Developer Advocate @ Redis

justin@redis.com

(AP Photo/ Eric Risberg)

Thanks to the sponsors of Codemash 2024!

OIT, LLC

Okta

CAS

Cloud Security Partners

Flexjet

Oracle

Manifest Solutions

CGI

AccumTech

JPMorgan Chase

CNWR

Fleet Device Management

Qase

Text Control, LLC

Daugherty Business Solutions

Callibrity

OverDrive Inc.

JumpMind

Camunda, Inc.

DOmedia

AccumTech

Improving

Code Squid, LLC

Coffee and Code

Making Software Greener

Covered today:

- Motivation for this project
- How air quality is determined
- How to measure airborne particulate matter
- Assembling the hardware sensor device
- Parsing the raw data
- Visualizing the data
- Extensibility and utility of data

Sad Introductory Stats for 2023 Wildfires in the United States West Coast:

- **2,540,000** acres of land burned
- **48,681** individual fires
- **176** acres average per fire
- **13,887** buildings destroyed
- Western US wildfires destroyed 246% more homes and buildings over the past decade

Sad Introductory Stats for 2020s Wildfires in the United States West Coast:

- **We learned about fire tornadoes**

Wildfire Smoke - How does it affect us?

- Eye and respiratory tract irritation
- Reduced lung function
- Bronchitis
- Exacerbation of Asthma and Heart Failure
- Overall Premature death

Wildfire Smoke - How we measure it

- PM 2.5: Particulate Matter 2.5 micrometers and smaller
- Small enough to pass through to the deepest part of the lungs and into the bloodstream
- AQI (Air Quality Index): a computed value based on PM 2.5 to convey health risks

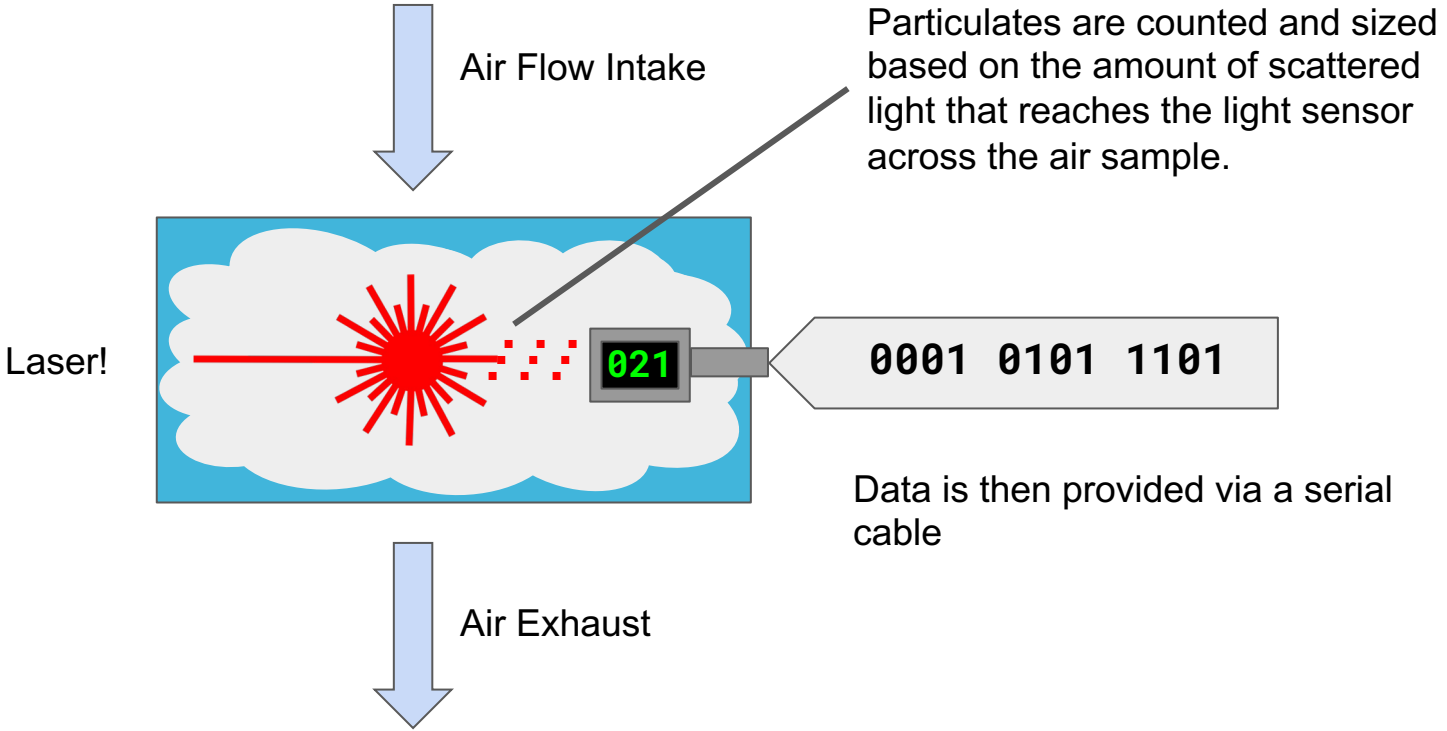
Wildfire Smoke - How we measure it

0 - 50	Good	Air quality is considered satisfactory, and air pollution poses little or no risk
51 - 100	Moderate	Air quality is acceptable; however, for some pollutants there may be a moderate health concern for a very small number of people who are unusually sensitive to air pollution.
101-150	Unhealthy for Sensitive Groups	Members of sensitive groups may experience health effects. The general public is not likely to be affected.
151-200	Unhealthy	Everyone may begin to experience health effects; members of sensitive groups may experience more serious health effects
201-300	Very Unhealthy	Health warnings of emergency conditions. The entire population is more likely to be affected.
300+	Hazardous	Health alert: everyone may experience more serious health effects

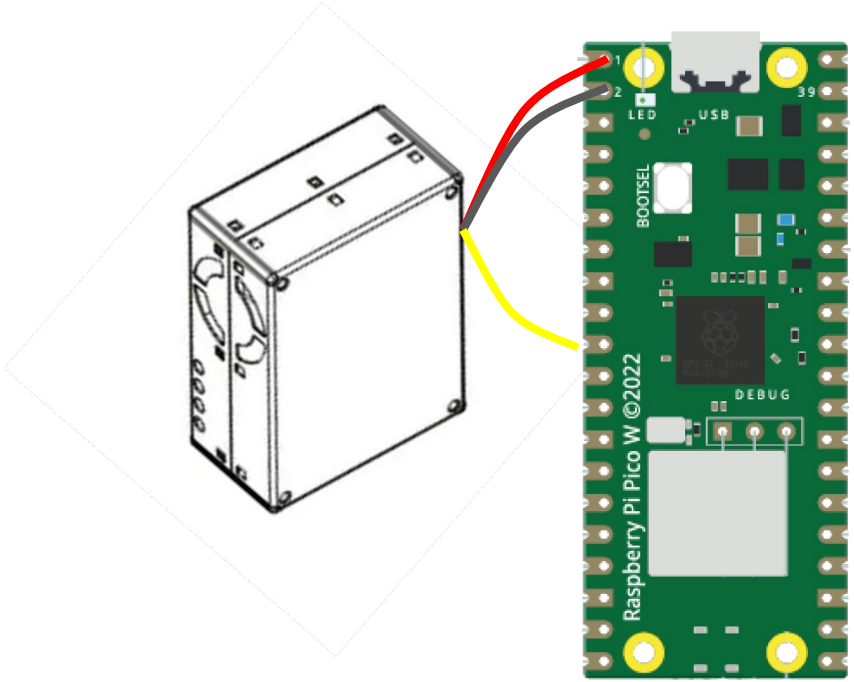
Wildfire Smoke - How do we measure it

Plantower PMS 5003 Particulate Matter Sensor

Wildfire Smoke - Plantower PMS5003 breakdown



The Raspberry Pi Pico



- Capable of running Micropython or Circuitpython
- Wireless capabilities
- Dual-core ARM processor,
- 264 kB of SRAM
- 2MB of on-board flash memory
- Runs on 5v, 1a USB power
- 21x51mm (4/5 x 2 inches)
- Only $6.00 (USD)

Redis

Store key and value pairs

- **Strings/Numbers**
- Lists/Sets/Sorted Sets
- Hashes
- **Streams**
- **TimeSeries**
- **JSON**
- Query

Pi Pico W Code - Tasks

- Send a liveliness pulse every five minutes
- Sample the air every five seconds
  - Convert PM2.5 to AQI
- Send PM2.5, AQI, and temperature to a Redis Stream

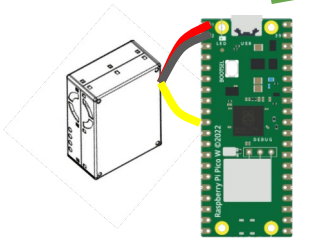
Pi Pico W Code - Send a liveliness pulse

```
import secrets
import picoredis as client

SENSOR_LOCATION = 'unit:2’
TTL_TIMER = 60 * 5

# Connect to RedisCloud database
redis = client.Redis(
host = secrets.REDIS_HOST,
port = secrets.REDIS_PORT)
redis.auth(secrets.REDIS_PASS)

# Initial announcement that we exist
redis.SET(f'ttl:{SENSOR_LOCATION}', 'active', 'EX', TTL_TIMER)
```

Pi Pico W Code – Sample the air

```
import PMS5003
import utility

sensor = PMS5003.PMS5003(
  uart=UART_connection,
  pin_enable=machine.Pin(3),
  pin_reset=machine.Pin(2),
  mode="active")

raw_reading = sensor.read()
pm25 = raw_reading['pm25’]
aqi = utility.convert(raw_reading)
temperature_reading = utility.read_onboard_temp()
```

Pi Pico W Code – But what's a Stream?



"office"

```
target:'office'
PM2.5: 4
AQI: 16
temp: 67.75
```

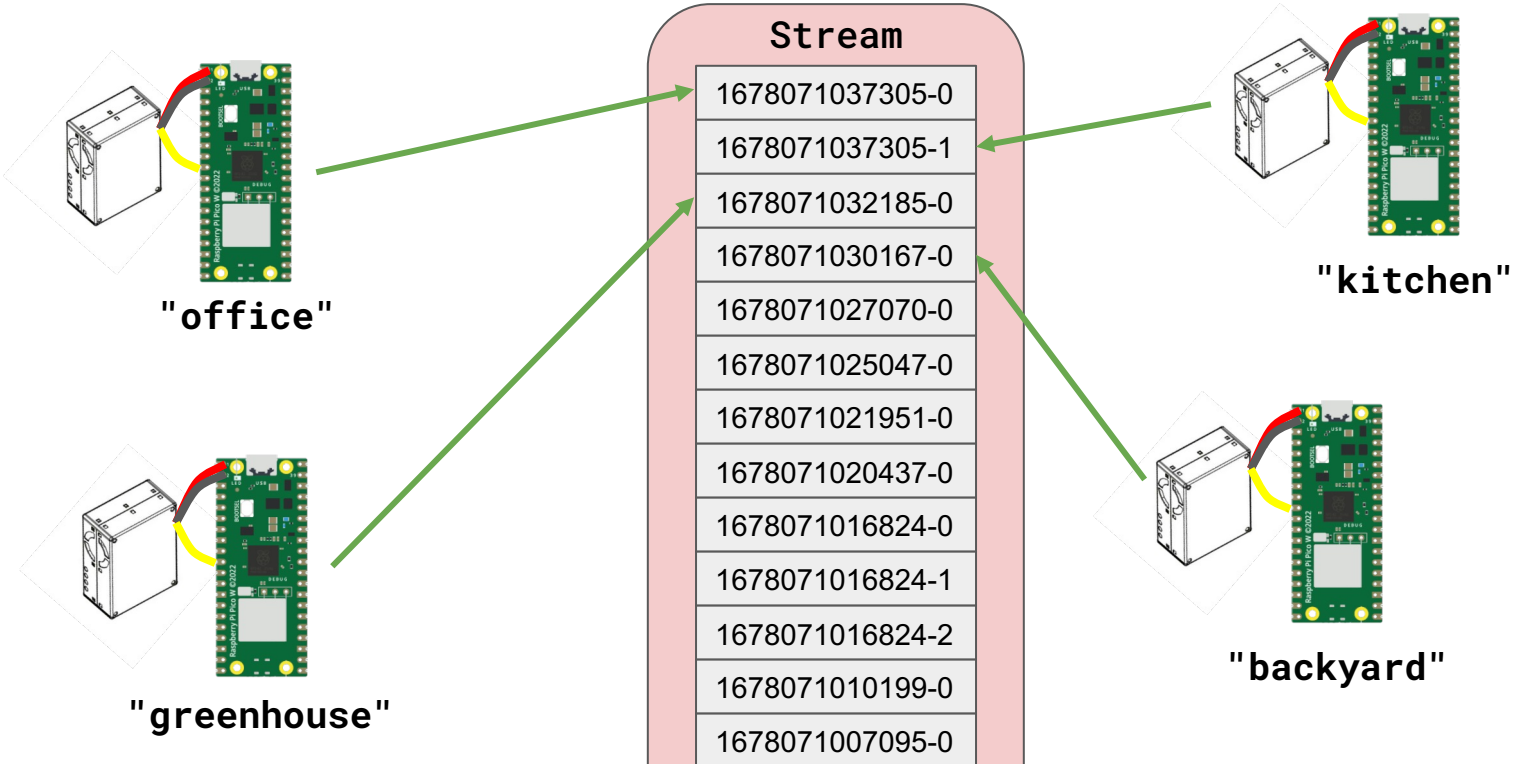
Stream

- 1678071037305-0
- 1678071035287-0
- 1678071032185-0
- 1678071030167-0
- 1678071027070-0
- 1678071025047-0
- 1678071020437-0
- 1678071016824-0
- 1678071016824-1
- 1678071016824-2
- 1678071010199-0
- 1678071007095-0
- 1678071005081-0

Newest

Oldest

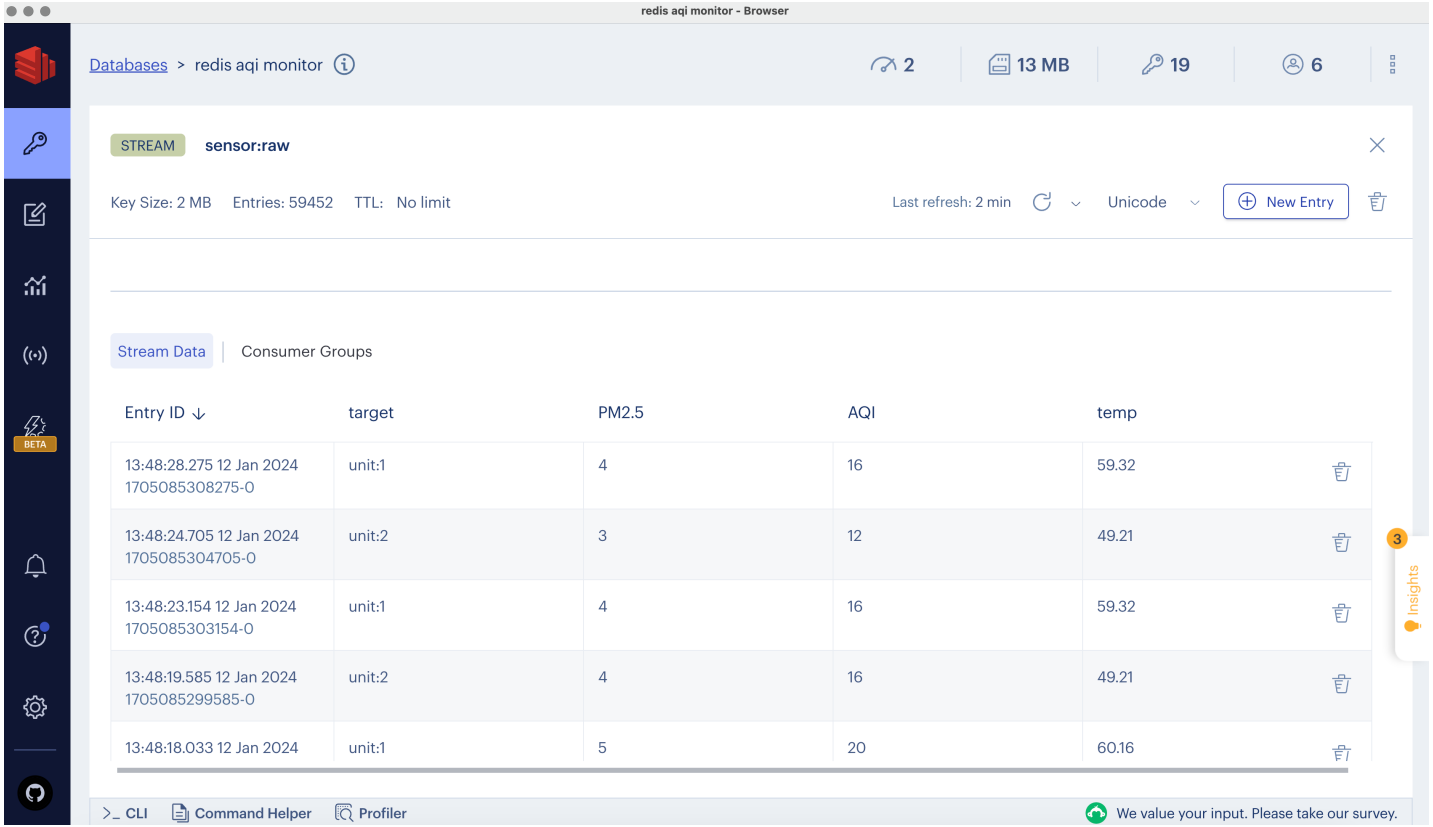
Pi Pico W Code – Scaling out our sensors



Pi Pico W Code – Send data to Redis Stream

```
SENSOR_LOCATION = 'unit:2'
STREAM_KEY = 'sensor:raw'

# send readings to Redis via a stream add command
redis.XADD(STREAM_KEY,
  '*',
  'target', SENSOR_LOCATION,
  'PM2.5', raw_reading,
  'AQI', aqi,
  'temp', temperature_reading)
```

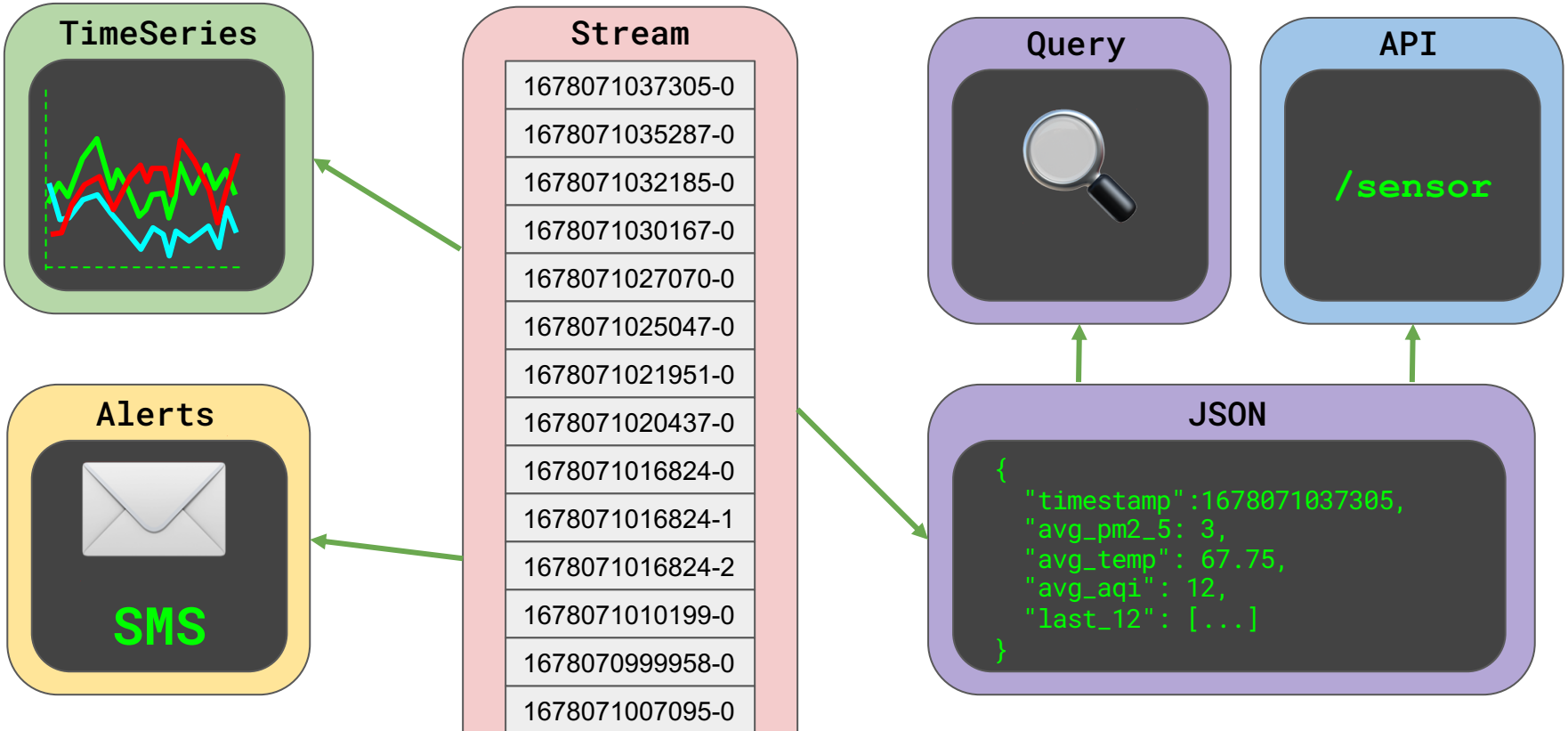
Pi Pico W Code – Send data to Redis Stream



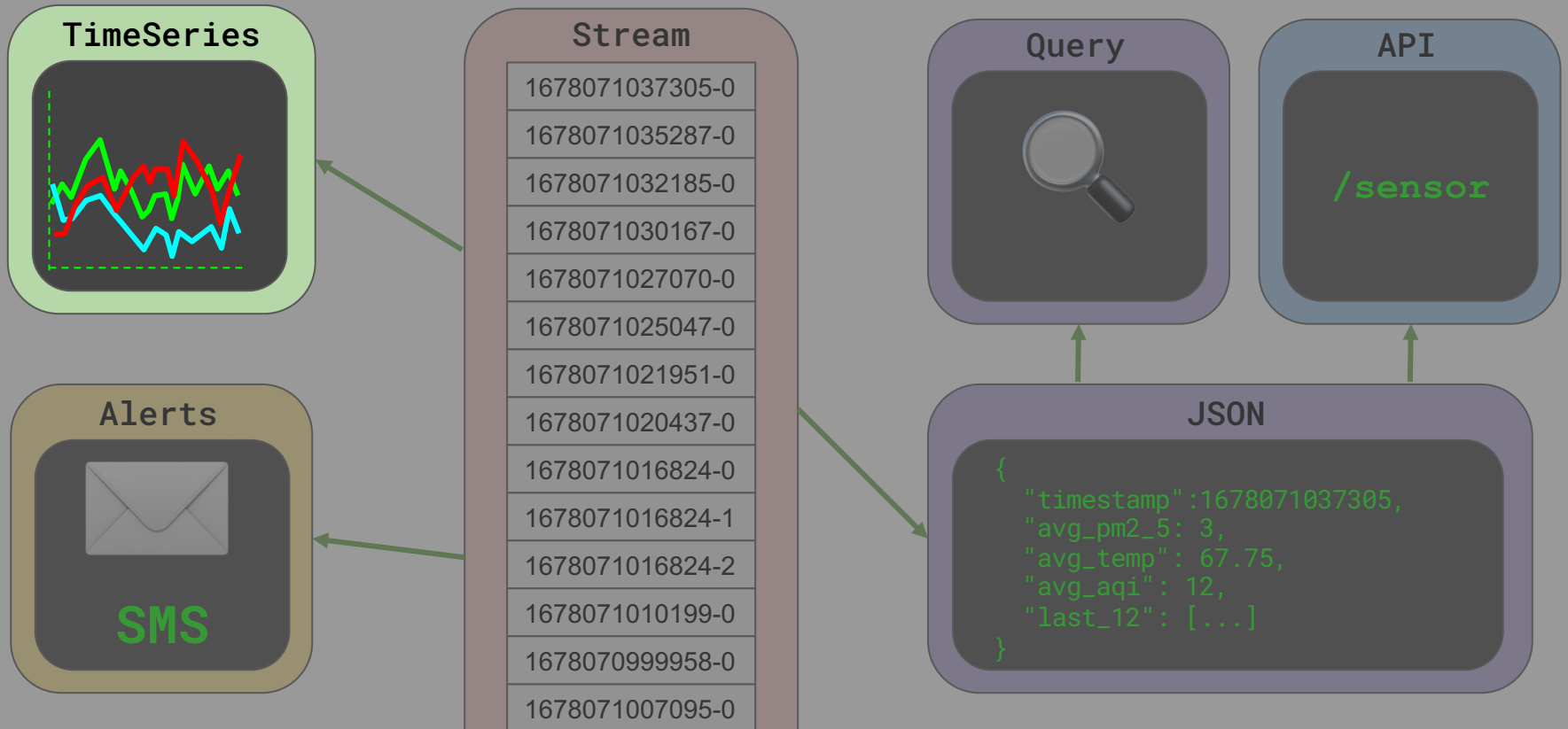
Python Microservices - Tasks

- Insert PM2.5, AQI, and temperature into individual timeseries per sensor unit
- Trigger SMS when air quality threshold is met
- Provide an API to Grafana for visualization

Producers and Consumers - Making the data work



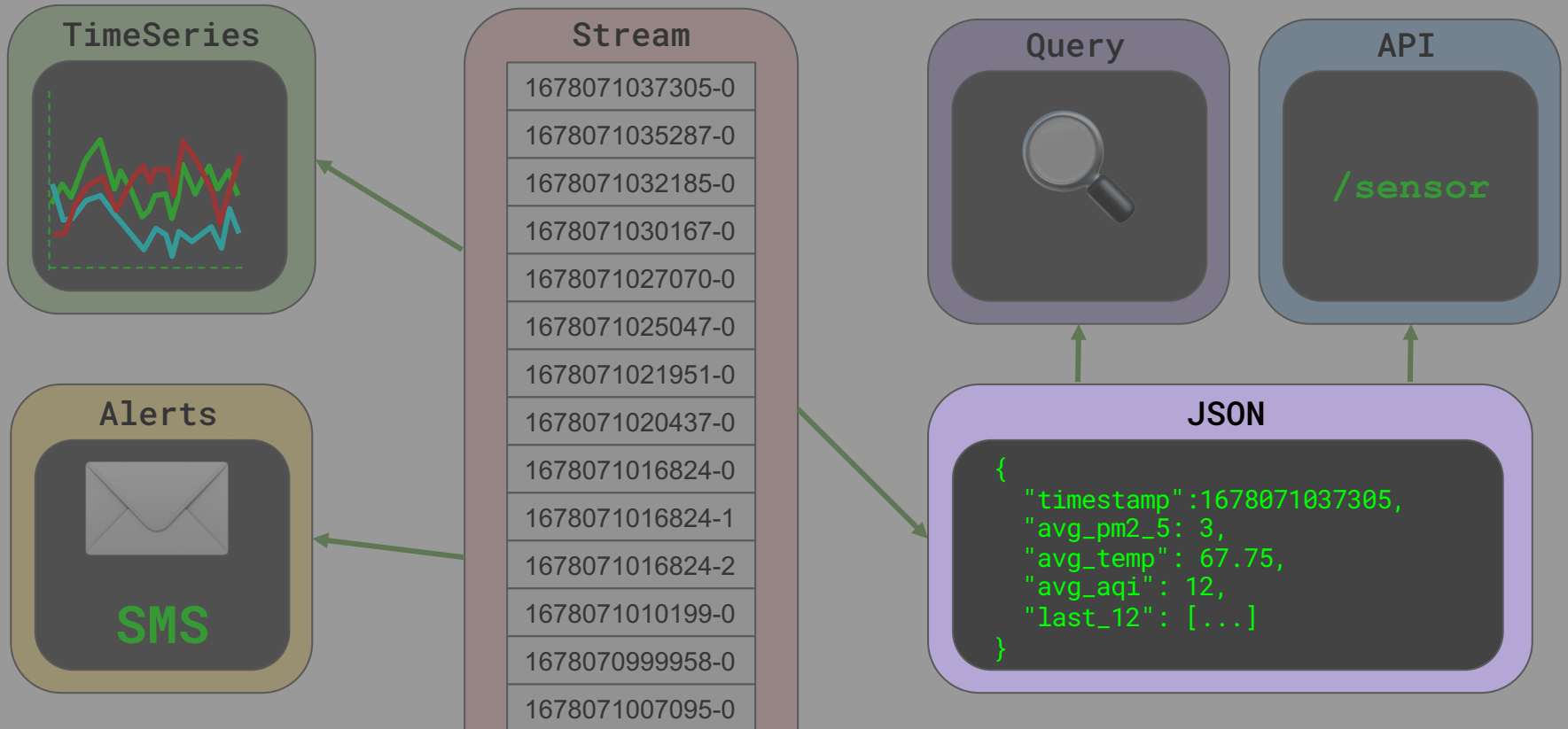
Python Microservices – Insert data into Timeseries



Python Microservices – Insert data into Timeseries

```
STREAM_KEY = 'sensor:raw'
stream_entry_id = '$'

while(True):
  # wait to get most recent stream entry
  result = redis.XREAD(streams={STREAM_KEY: stream_entry_id} ,
    count=1,
    block=50000)

  ts_key_prefix = f'ts:{result["target"]}'

  # create three separate timeseries entries from each stream entry
  redis.TS().ADD(f'{ts_key_prefix}:aqi',
    result['timestamp'],
    result['AQI'])

  stream_entry_id = result[‘entry_id']
```

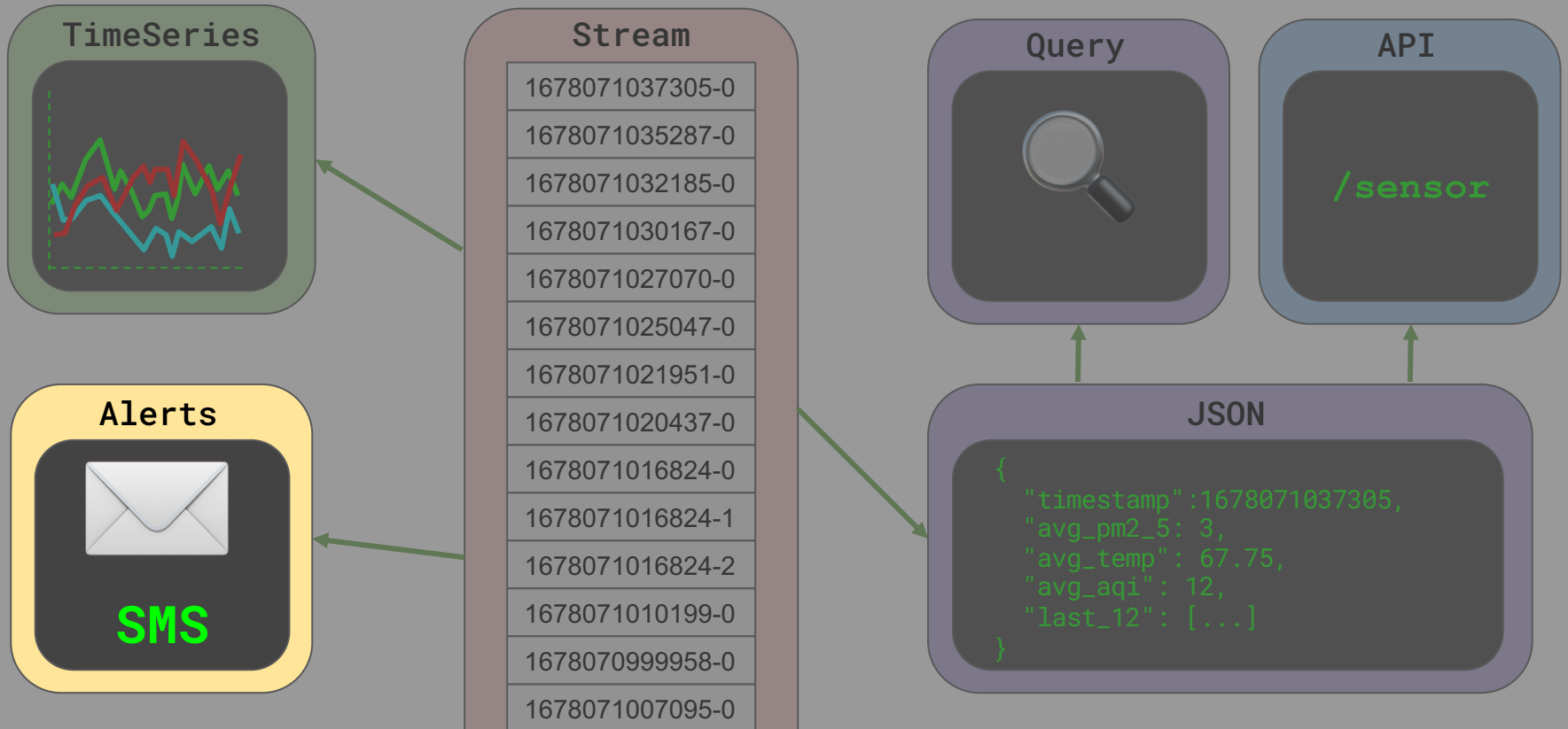
Python Microservices – Convert Data to JSON



Python Microservices – Convert Data to JSON

```
STREAM_KEY = 'sensor:raw'
stream_entry_id = '$'

while(True):
  result = r.xread(streams={STREAM_KEY: stream_entry_id},
    count=1,
    block=50000)

  redis.JSON().SET(json_key, '.',
    { 'timestamp': result['timestamp']
      'current_pm2_5': result['pm2_5'],
      'current_temp': result['temp'],
      'current_aqi': result['aqi'],
    })

  stream_entry_id = result['entry_id']
```

Python Microservices – SMS Alerts!



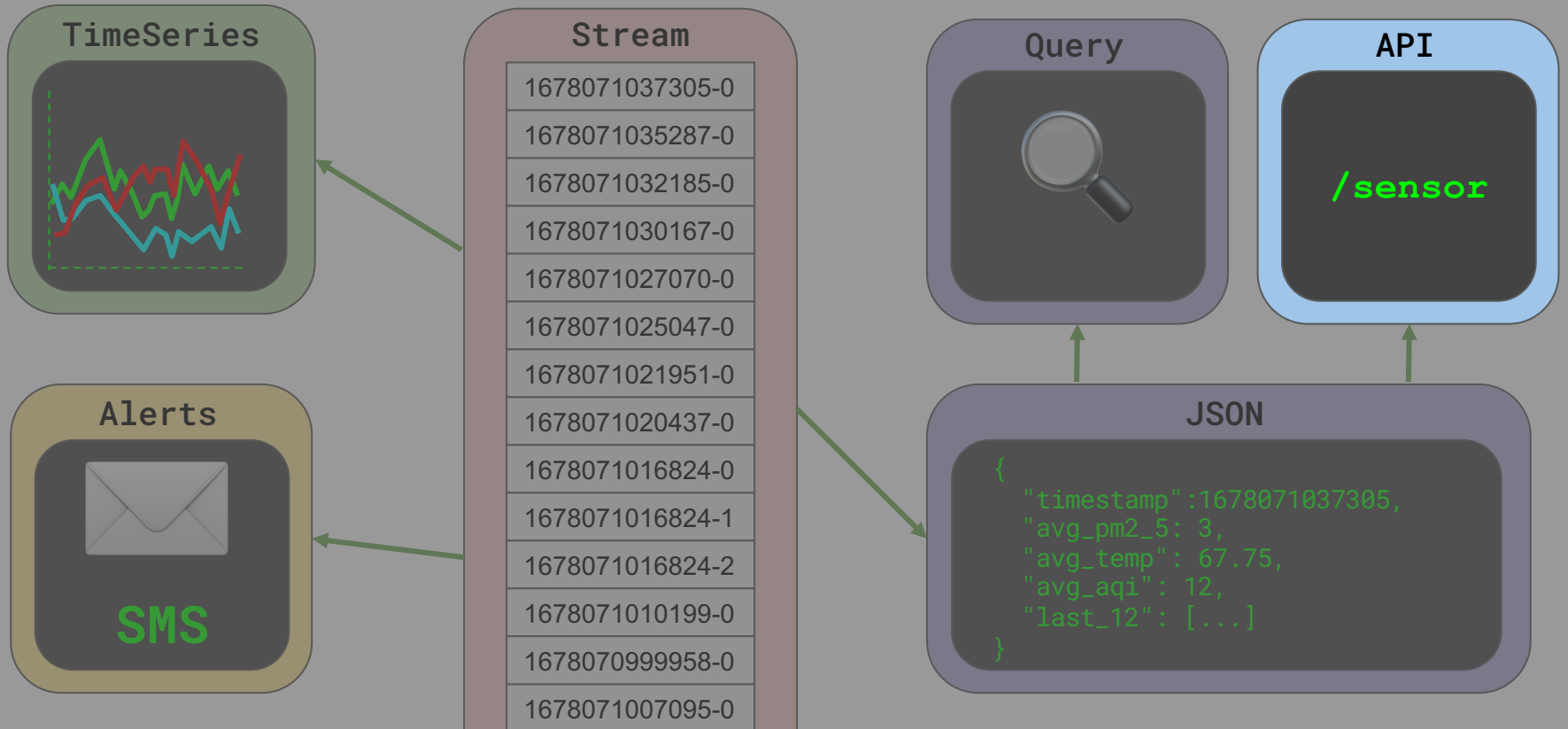
Python Microservices – SMS Alerts!

```
AQI_THRESHOLD = 100*12

last_12 = [-1] * 12
redis.JSON().SET(json_key, '$',{ 'last_12': last_12 })

# "slide" out the oldest aqi reading, add the newest
last_12.append(aqi)
last_12.pop(0)

if sum(last_12) >= AQI_THRESHOLD:
  aqi_average = floor(sum(last_12)/12)
  was_notified = redis.GET('user_notified')

  if not was_notified:
    alert(aqi_average, target)
    r.set('user_notified', 1, 3600)
```

Python Microservices – SMS Alerts!

```
from twilio.rest import Client

client = Client(account_sid, auth_token)

def alert(value, location):
  message = client.messages.create(
    from_='my-phone-number',
    body=f'Hello, the current AQI is {value} at the {location} location.',
    to=phone_number)
  return message
```

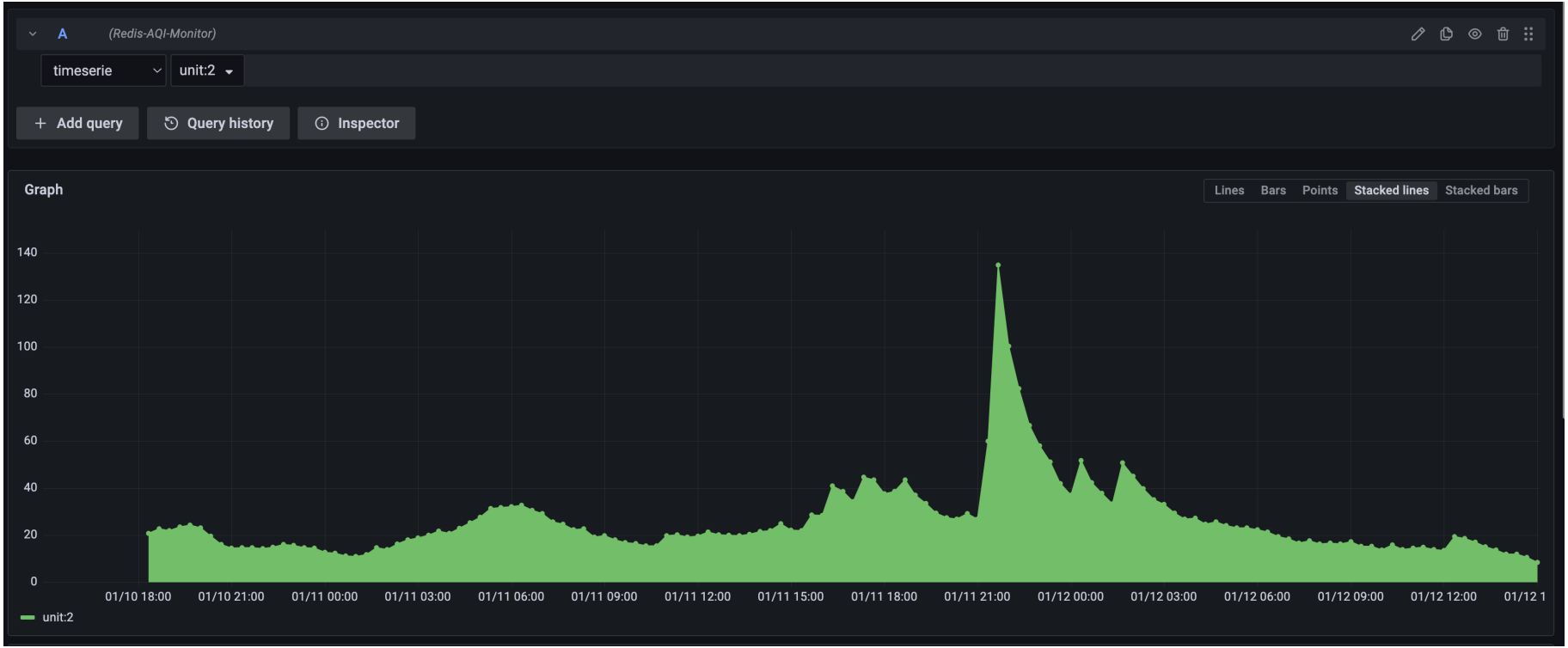
Python Microservices – Provide API to Grafana



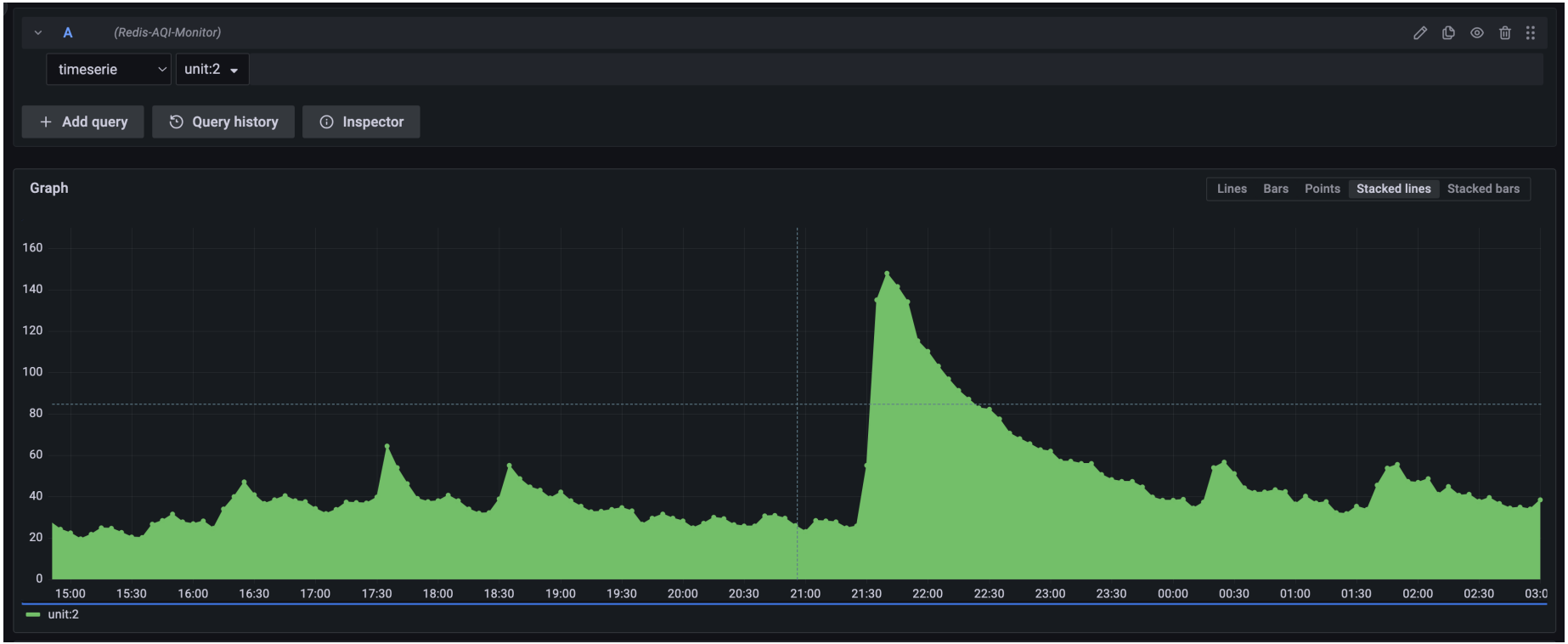
Python Microservices – Provide API to Grafana

```
@app.post("/search")
  active_sensors = redis.KEYS('ttl:*’) # gathers all active keys

@app.post("/query") # sends from, to, and interval size

  results = redis.TS().RANGE(ts_key,
  from_time,
  to_time,
  aggregation_type = 'avg’,
  bucket_size_msec = int(interval))
```

Python Microservices – Provide API to Grafana

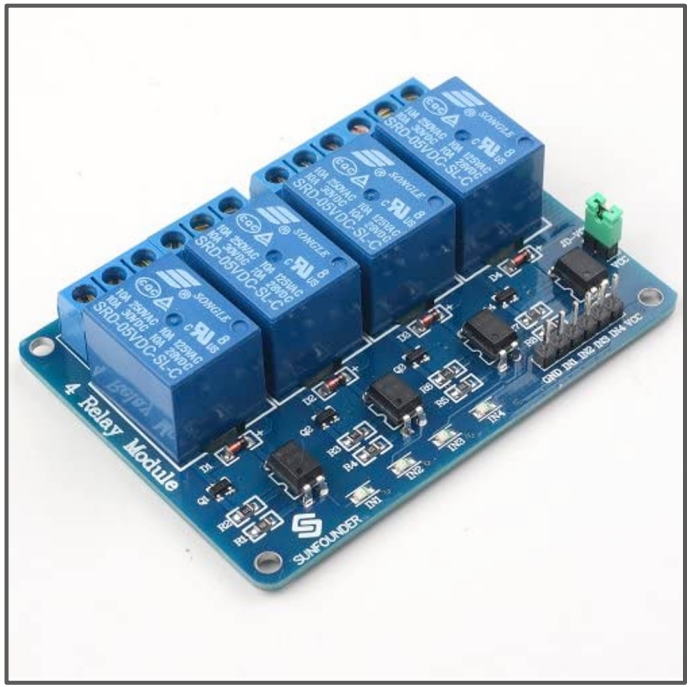


Python Microservices – Provide API to Grafana



What else can you do with this data?

- Trigger an electric relay to activate a fan, air purifier, window opener, or HVAC system.
- Share outdoor locations with crowdsourced AQI maps, such as PurpleAir.
- Send notifications to Alexa to alert rooms of high AQI values
- Email notifications
- Create a heat map of a building of changing AQI values
- Create a predictive model of air dispersal/ventilation



Shoutout to PurpleAir!

Shoutout to PurpleAir!

https://map.purpleair.com/

Learn more about this project

Github repository:

- Pico W code
- Consumer services code
- API code
- Instructions on assembling your own unit
- .STL files for printing a box at home
- Data sources of statistics

https://github.com/redis-developer/redis-aqi-monitor.git

Learn more about Redis

Redis:
https://redis.com

Redis University:
https://university.redis.com

Youtube:
https://youtube.com/redis

Discord
https://discord.gg/redis

Justin Castilla

Developer Advocate @ Redis

justin@redis.com