PULSAR SUMMIT | Pulsar Summit Asia 2022

Apache Pulsar Development 101 with Python

Timothy Spann, Developer Advocate

Tim Spann
Developer Advocate
StreamNative

https://streamnative.io/pulsar-python/

FLiP(N) Stack = Flink, Pulsar and NiFi Stack
Streaming Systems & Data Architecture Expert

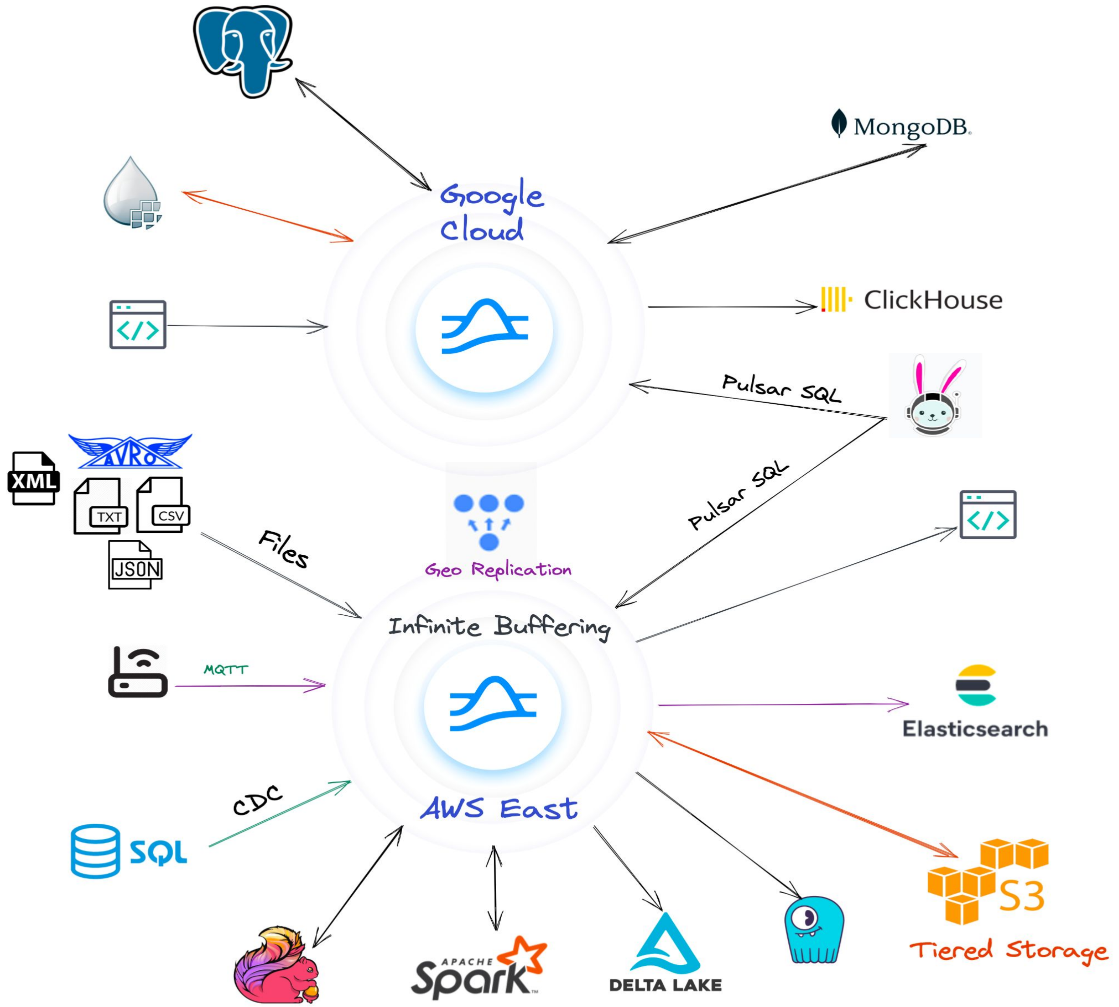
Experience
15+ years of experience with streaming technologies including Pulsar, Flink, Spark, NiFi, Big Data, Cloud, MXNet, IoT, Python and more.

Today, he helps to grow the Pulsar community sharing rich technical knowledge and experience at both global conferences and through individual conversations.

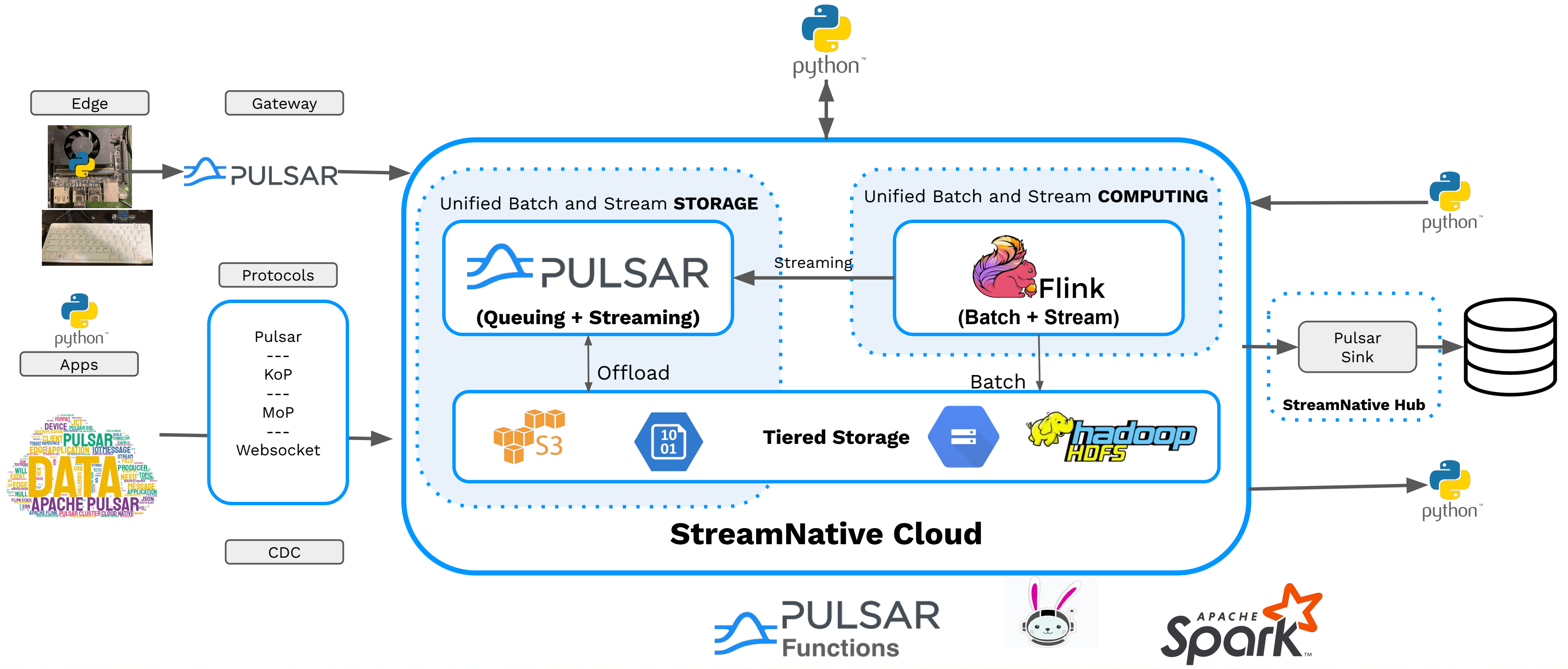
FLiP Stack Weekly

https://bit.ly/32dAJft

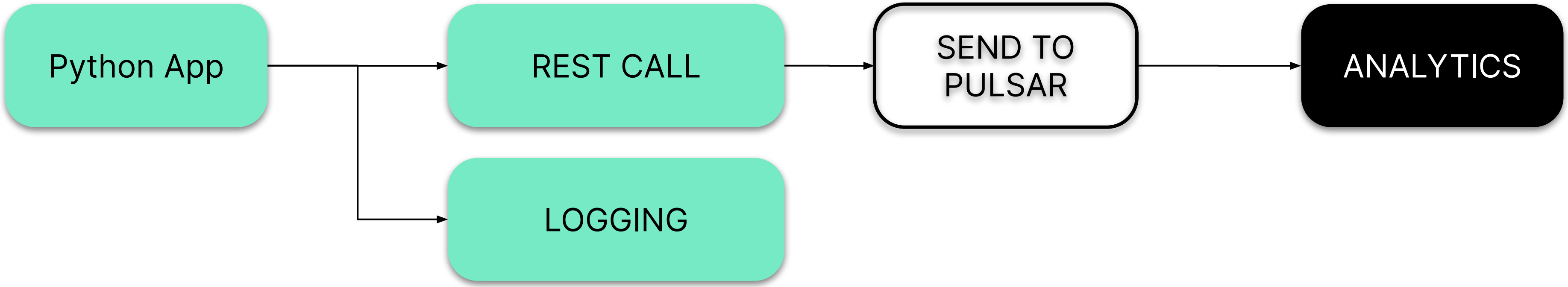
This week in Apache Flink, Apache Pulsar, Apache NiFi, Apache Spark and open source friends.


MongoDB
Google Cloud
ClickHouse
Pulsar SQL
Pulsar SQL
XML
AVRO
TXT
CSV
JSON
Files
Geo Replication
Infinite Buffering
MQTT
Elasticsearch
CDC
SQL
AWS East
S3
Tiered Storage
APACHE Spark
DELTA LAKE

Streaming FLiP-Py Apps

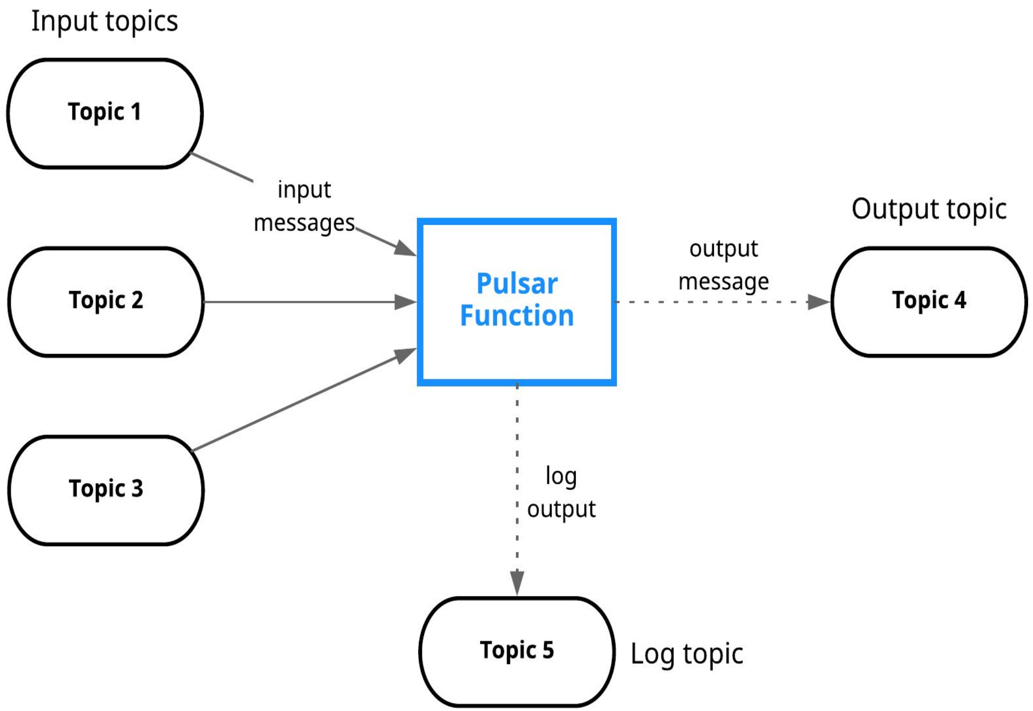


Python Application for ADS-B Data
Diagram



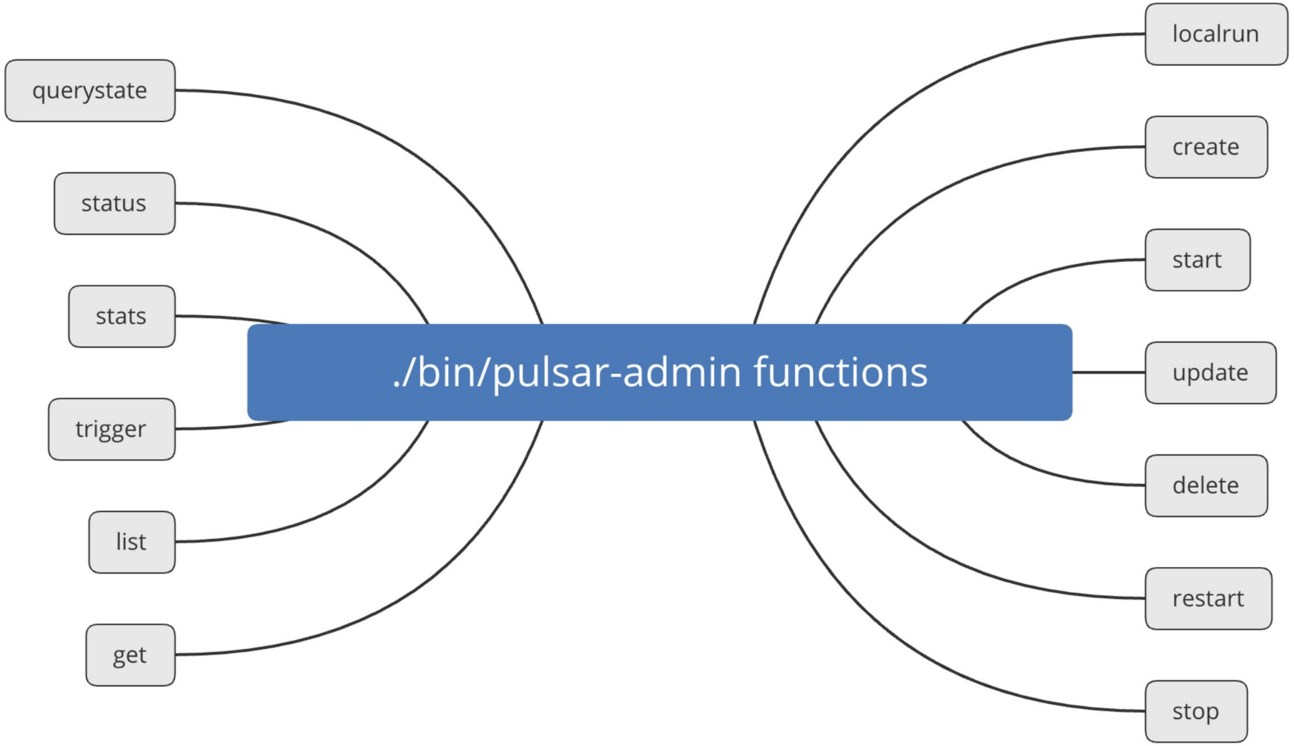
https://github.com/tspannhw/FLiP-Py-ADS-B

Pulsar Functions



- Consume messages from one or more Pulsar topics.
- Apply user-supplied processing logic to each message.
- Publish the results of the computation to another topic.
- Support multiple programming languages (Java, Python, Go)
- Can leverage 3rd-party libraries

Integrated with pulsar-admin CLI



Pulsar Summit
Asia 2022

Pulsar Functions

Entire Function

```
#!/usr/bin/env python

from pulsar import Function
import json

class Chat(Function):
    def __init__(self):
        pass

    def process(self, input, context):
        logger = context.get_logger()
        logger.info("Message Content: {0}".format(input))
        msg_id = context.get_message_id()
        row = { }
        row['id'] = str(msg_id)
        json_string = json.dumps(row)
      return json_string
```

Python 3 Coding

Code Along With Tim
<<DEMO>>

Building Tenant, Namespace, Topics

```
bin/pulsar-admin tenants create conference

bin/pulsar-admin namespaces create conference/pulsarsummit

bin/pulsar-admin tenants list

bin/pulsar-admin namespaces list conference

bin/pulsar-admin topics create persistent://conference/pulsarsummit/first

bin/pulsar-admin topics list conference/pulsarsummit
```

Install Python 3 Pulsar Client

```
pip3 install pulsar-client=='2.10.1[all]'

# Depending on Platform May Need to Build C++ Client
```

For Python on Pulsar on Pi https://github.com/tspannhw/PulsarOnRaspberryPi

https://pulsar.apache.org/docs/en/client-libraries-python/

Building a Python 3 Producer

```
import pulsar
client = pulsar.Client('pulsar://localhost:6650')
producer = client.create_producer('persistent://conference/ps/first')
producer.send(('Simple Text Message').encode('utf-8'))
client.close()
```

Building a Python 3 Cloud Producer Oath

```
python3 prod.py -su pulsar+ssl://name1.name2.snio.cloud:6651 -t
persistent://public/default/pyth --auth-params
'{"issuer_url":"https://auth.streamnative.cloud", "private_key":"my.json",
"audience":"urn:sn:pulsar:name:myclustr"}'

from pulsar import Client, AuthenticationOauth2
parse = argparse.ArgumentParser(prog=prod.py')
parse.add_argument('-su', '--service-url', dest='service_url', type=str,
required=True)
args = parse.parse_args()
client = pulsar.Client(args.service_url,
          authentication=AuthenticationOauth2(args.auth_params))
```

https://github.com/streamnative/examples/blob/master/cloud/python/OAuth2Producer.py

Example Avro Schema Usage

```
import pulsar
from pulsar.schema import *
from pulsar.schema import AvroSchema
class thermal(Record):
    uuid = String()
client = pulsar.Client('pulsar://pulsar1:6650')
thermalschema = AvroSchema(thermal)
producer =
client.create_producer(topic='persistent://public/default/pi-thermal-avro',
         schema=thermalschema,properties={"producer-name": "thrm" })
thermalRec = thermal()
thermalRec.uuid = "unique-name"
producer.send(thermalRec,partition_key=uniqueid)
```

https://github.com/tspannhw/FLiP-Pi-Thermal

Example Json Schema Usage

```
import pulsar
from pulsar.schema import *
from pulsar.schema import JsonSchema
class weather(Record):
    uuid = String()
client = pulsar.Client('pulsar://pulsar1:6650')
wschema = JsonSchema(thermal)
producer =
client.create_producer(topic='persistent://public/default/weathe
r,schema=wschema,properties={"producer-name": "wthr" })
weatherRec = weather()
weatherRec.uuid = "unique-name"
producer.send(weatherRec,partition_key=uniqueid)
```

https://github.com/tspannhw/FLiP-Pi-Weather

Building a Python3 Consumer

```
import pulsar
client = pulsar.Client('pulsar://localhost:6650')
consumer =
client.subscribe('persistent://conference/ps/first',subscription_name='my-
sub')

while True:
    msg = consumer.receive()
    print("Received message: '%s'" % msg.data())
    consumer.acknowledge(msg)
client.close()
```

MQTT from Python

```
pip3 install paho-mqtt

import paho.mqtt.client as mqtt
client = mqtt.Client("rpi4-iot")
row = { }
row['gasKO'] = str(readings)
json_string = json.dumps(row)
json_string = json_string.strip()
client.connect("pulsar-server.com", 1883, 180)
client.publish("persistent://public/default/mqtt-2",
payload=json_string,qos=0,retain=True)
```

https://www.slideshare.net/bunkertor/data-minutes-2-apache-pulsar-with-mqtt-for-edge-computing-lightning-2022

Web Sockets from Python

```
pip3 install websocket-client

import websocket, base64, json
topic = 'ws://server:8080/ws/v2/producer/persistent/public/default/webtopic1'
ws = websocket.create_connection(topic)
message = "Hello Python Web Conference"
message_bytes = message.encode('ascii')
base64_bytes = base64.b64encode(message_bytes)
base64_message = base64_bytes.decode('ascii')
ws.send(json.dumps({'payload' : base64_message,'properties': {'device' :
'jets','protocol' : 'websockets'},'context' : 5}))
response =  json.loads(ws.recv())
```

https://github.com/tspannhw/FLiP-IoT/blob/main/wsreader.py
https://github.com/tspannhw/FLiP-IoT/blob/main/wspulsar.py
https://pulsar.apache.org/docs/en/client-libraries-websocket/

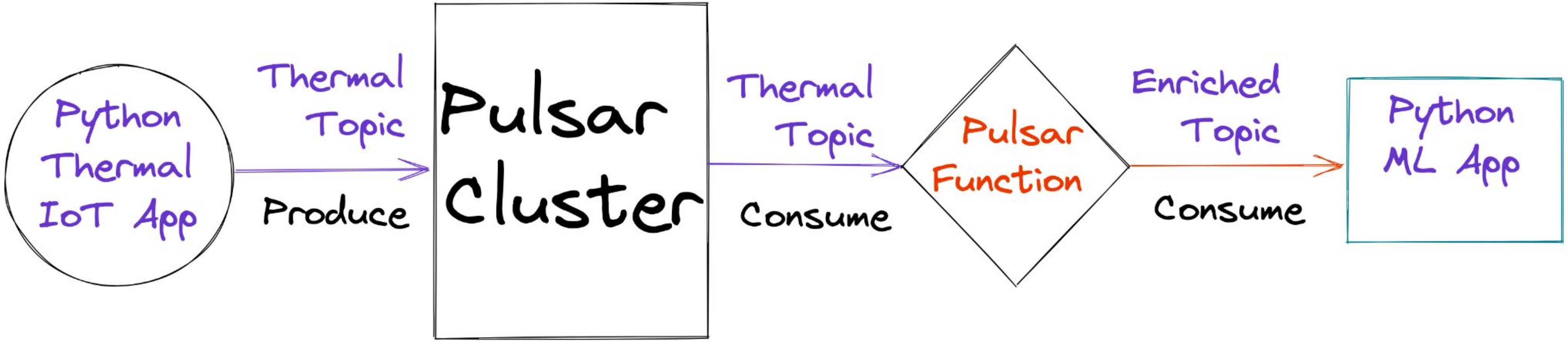
Kafka from Python

```
pip3 install kafka-python

from kafka import KafkaProducer
from kafka.errors import KafkaError

row = { }
row['gasKO'] = str(readings)
json_string = json.dumps(row)
json_string = json_string.strip()

producer = KafkaProducer(bootstrap_servers='pulsar1:9092',retries=3)
producer.send('topic-kafka-1', json.dumps(row).encode('utf-8'))
producer.flush()
```

https://docs.streamnative.io/platform/v1.0.0/concepts/kop-concepts

https://github.com/streamnative/kop

Pulsar IO Functions in Python



https://github.com/tspannhw/pulsar-pychat-function

Pulsar IO Functions in Python

```
from pulsar import Function
import json


class Chat(Function):
    def __init__(self):
        pass

    def process(self, input, context):
        logger = context.get_logger()

        msg_id = context.get_message_id()

        fields = json.loads(input)
```

Pulsar IO Functions in Python

```
bin/pulsar-admin functions create --auto-ack true
--py py/src/sentiment.py --classname
"sentiment.Chat" --inputs
"persistent://public/default/chat" --log-topic
"persistent://public/default/logs" --name Chat
--output "persistent://public/default/chatresult"
```

https://github.com/tspannhw/pulsar-pychat-function

Python For Pulsar on Pi

- https://github.com/tspannhw/FLiP-Pi-BreakoutGarden
- https://github.com/tspannhw/FLiP-Pi-Thermal
- https://github.com/tspannhw/FLiP-Pi-Weather
- https://github.com/tspannhw/FLiP-RP400
- https://github.com/tspannhw/FLiP-Py-Pi-GasThermal
- https://github.com/tspannhw/FLiP-PY-FakeDataPulsar
- https://github.com/tspannhw/FLiP-Py-Pi-EnviroPlus
- https://github.com/tspannhw/PythonPulsarExamples
- https://github.com/tspannhw/pulsar-pychat-function
- https://github.com/tspannhw/FLiP-PulsarDevPython101

PULSAR SUMMIT | Thanks

https://www.linkedin.com/in/timothyspann

https://github.com/tspannhw

@PassDev

https://streamnative.io/pulsar-python/