Carnegie Mellon University

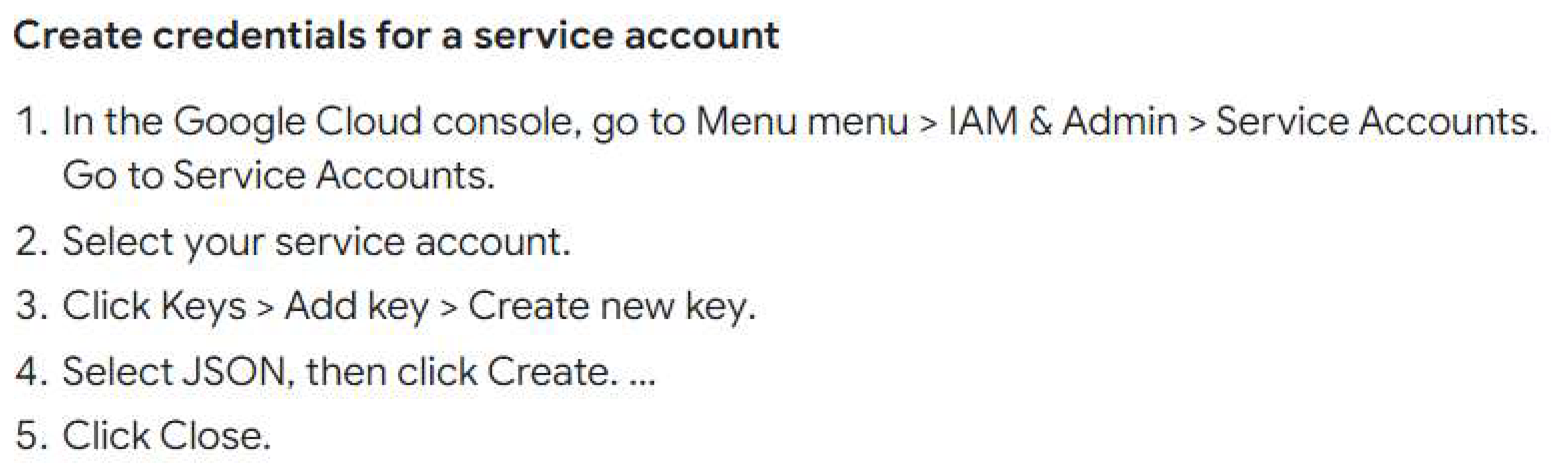
14-848 Cloud Infrastructure

CONFLUENT KAFKA LAB

Agenda

- Environment Setup and Required Dependencies
- Simple Kafka Example
- YouTube Comment Streaming
- Streamlit

1. Manage Authentication to Cloud Services: Generate Authentication JSON File



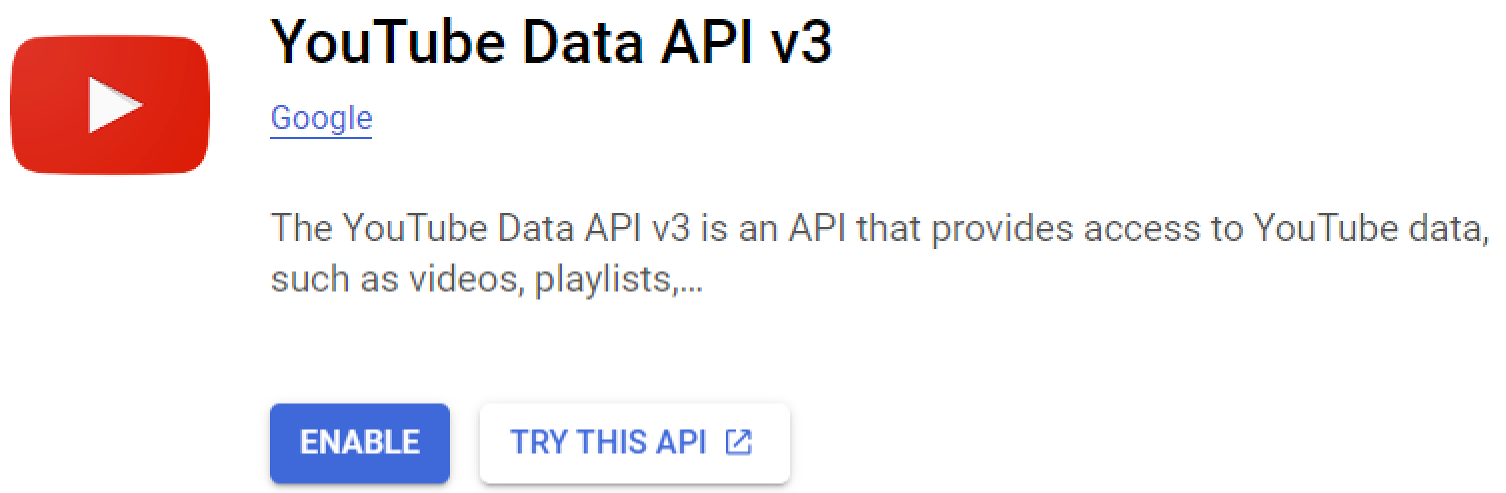
Create credentials for a service account

1. In the Google Cloud console, go to Menu menu > IAM & Admin > Service Accounts.
   Go to Service Accounts.
2. Select your service account.
3. Click Keys > Add key > Create new key.
4. Select JSON, then click Create. ...
5. Click Close.

- Place the generated JSON file where your python code will be located.

Carnegie Mellon University

2. APIs to Enable on Google Cloud


YouTube Data API v3

Google

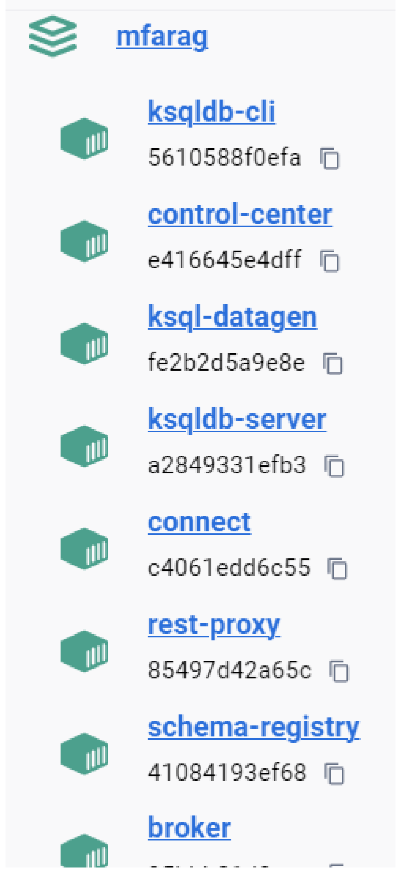
The YouTube Data API v3 is an API that provides access to YouTube data, such as videos, playlists,...

- Don't forget to keep the billing enabled during the YouTube exercise (and disable it afterwards).

3. Run Confluent-Kafka on Your Local Machine

- Follow the steps provided last lecture to start your local Confluent-Kafka cluster: https://docs.confluent.io/platform/current/get-started/platform-quickstart.html

- Verify your Kafka cluster is running.



mfarag

ksqldb-cli
5610588f0efa

control-center
e416645e4dff

ksql-datagen
fe2b2d5a9e8e

ksqldb-server
a2849331efb3

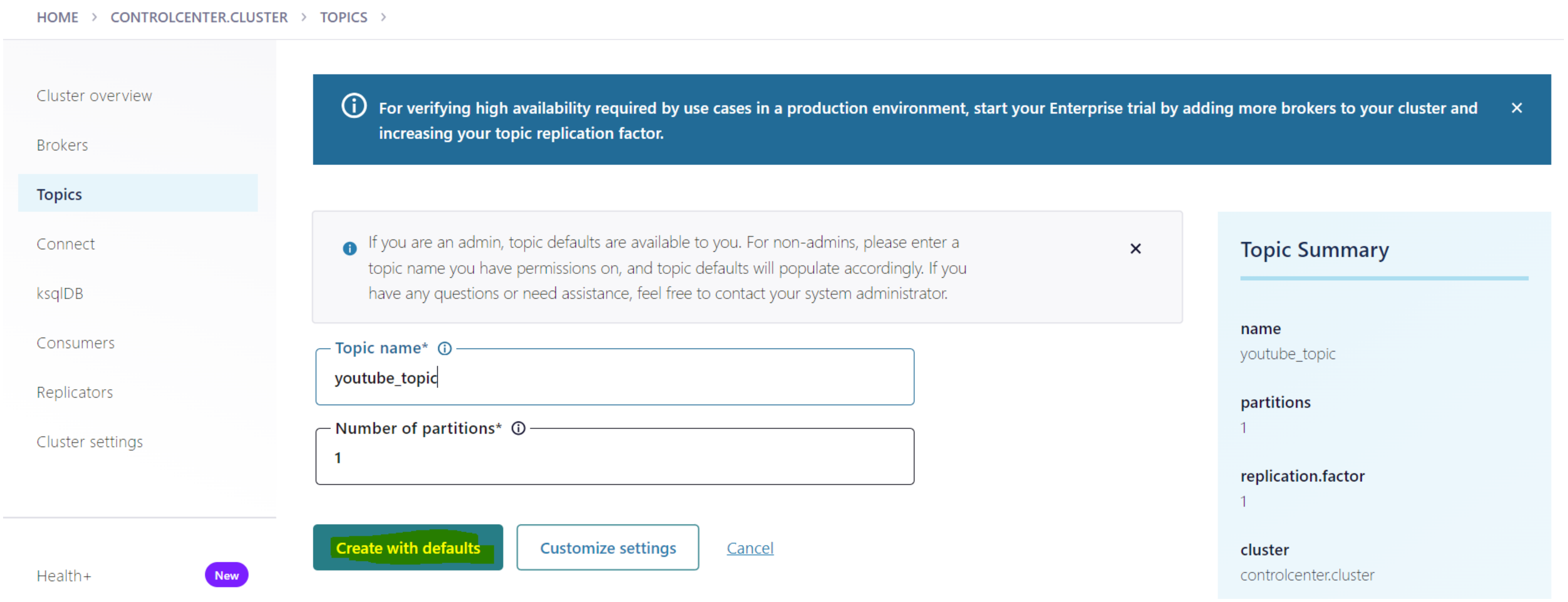
connect
c4061edd6c55

rest-proxy
85497d42a65c

schema-registry
41084193ef68

broker

4. Create a Confluent-Kafka Topic



HOME › CONTROLCENTER.CLUSTER › TOPICS ›

For verifying high availability required by use cases in a production environment, start your Enterprise trial by adding more brokers to your cluster and increasing your topic replication factor.

If you are an admin, topic defaults are available to you. For non-admins, please enter a topic name you have permissions on, and topic defaults will populate accordingly. If you have any questions or need assistance, feel free to contact your system administrator.

Topic name*: youtube_topic

Number of partitions*: 1

Topic Summary

name
youtube_topic

partitions
1

replication.factor
1

cluster
controlcenter.cluster

Carnegie Mellon University

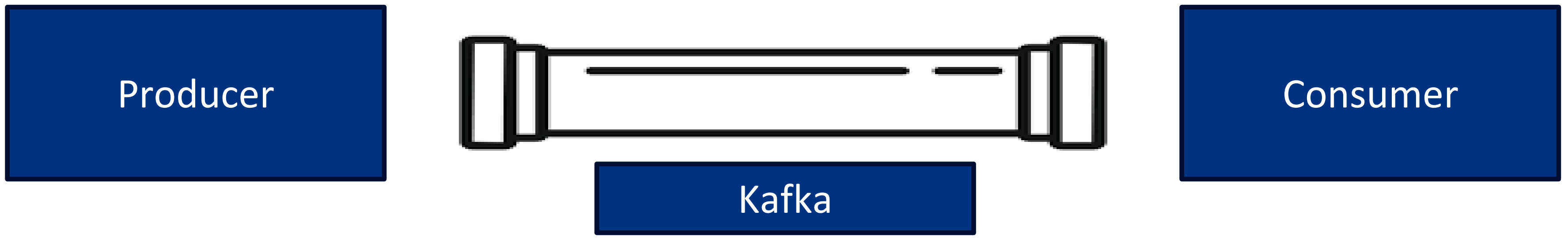
5. Install Required Libraries

Run the following command at the terminal where you will run your python code from:

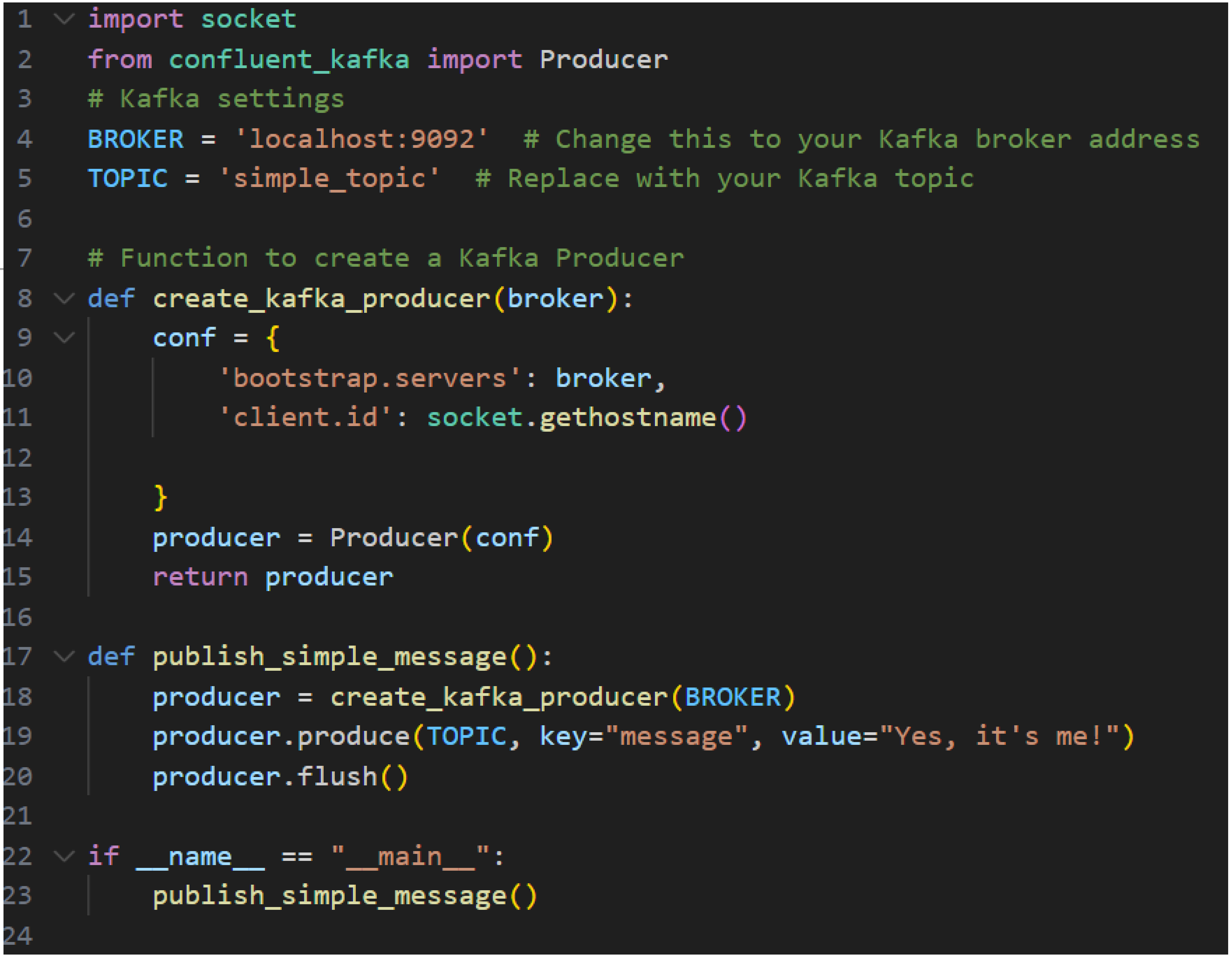
```
pip install confluent_kafka google-api-python-client streamlit
```

- If you plan to run your code in Jupyter notebook, 1) run this command in your Jupyter Notebook terminal or 2) prefix it with **!** And run it in a separate code cell

Simple Producer/Consumer Example

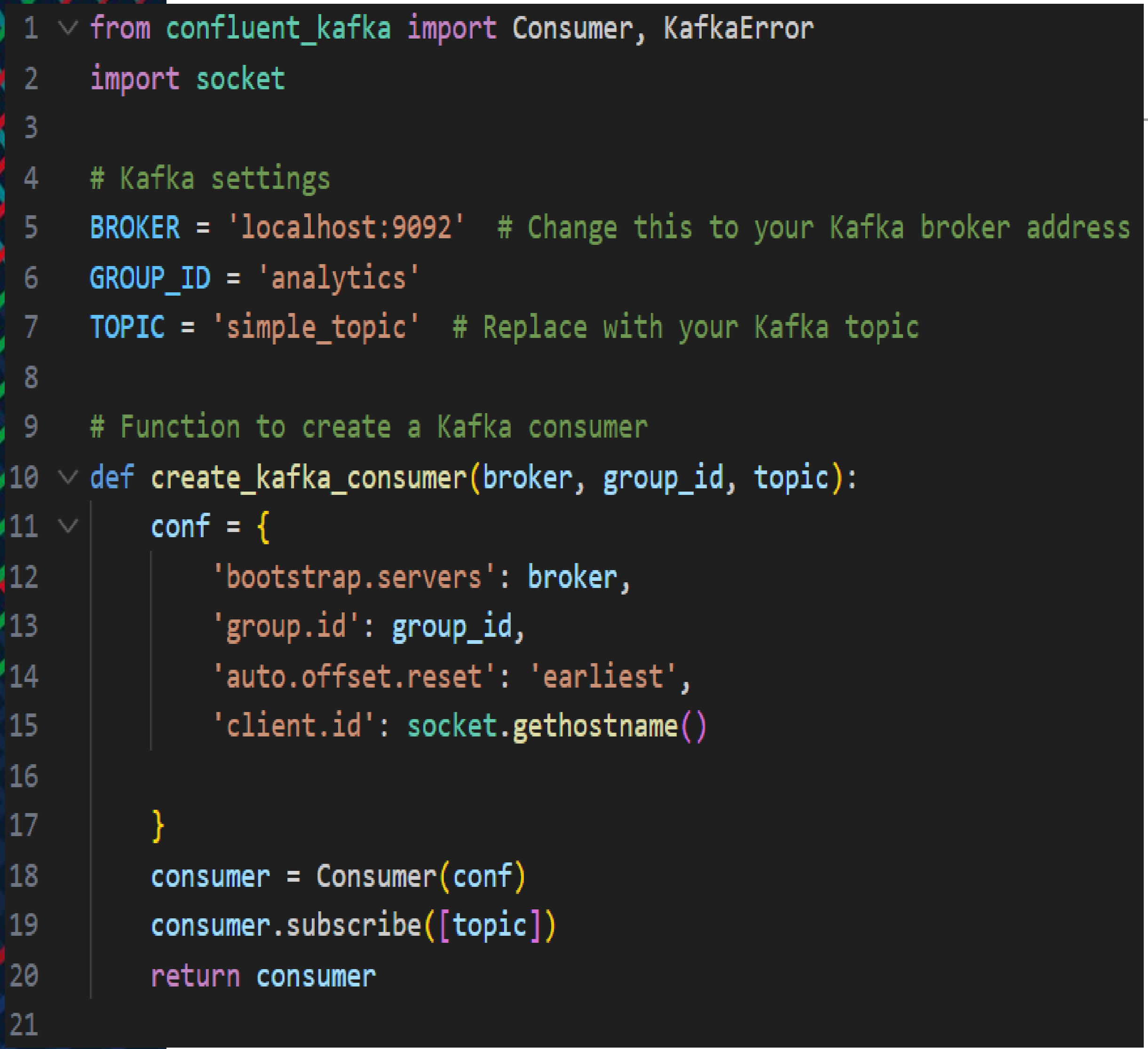


Producer

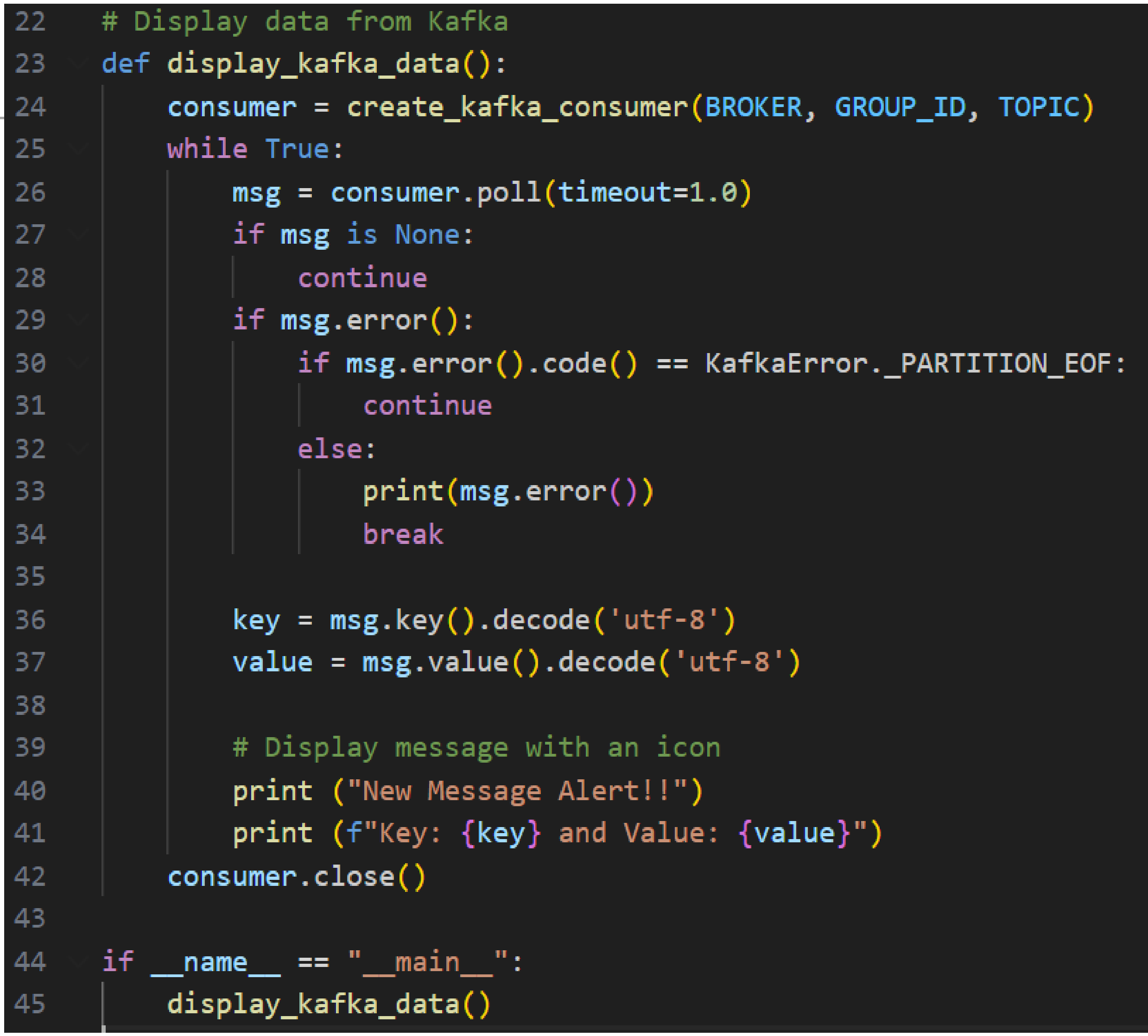


```python
import socket
from confluent_kafka import Producer
# Kafka settings
BROKER = 'localhost:9092'  # Change this to your Kafka broker address
TOPIC = 'simple_topic'  # Replace with your Kafka topic

# Function to create a Kafka Producer
def create_kafka_producer(broker):
    conf = {
        'bootstrap.servers': broker,
        'client.id': socket.gethostname()

    }
    producer = Producer(conf)
    return producer

def publish_simple_message():
    producer = create_kafka_producer(BROKER)
    producer.produce(TOPIC, key="message", value="Yes, it's me!")
    producer.flush()

if __name__ == "__main__":
    publish_simple_message()

```

Consumer

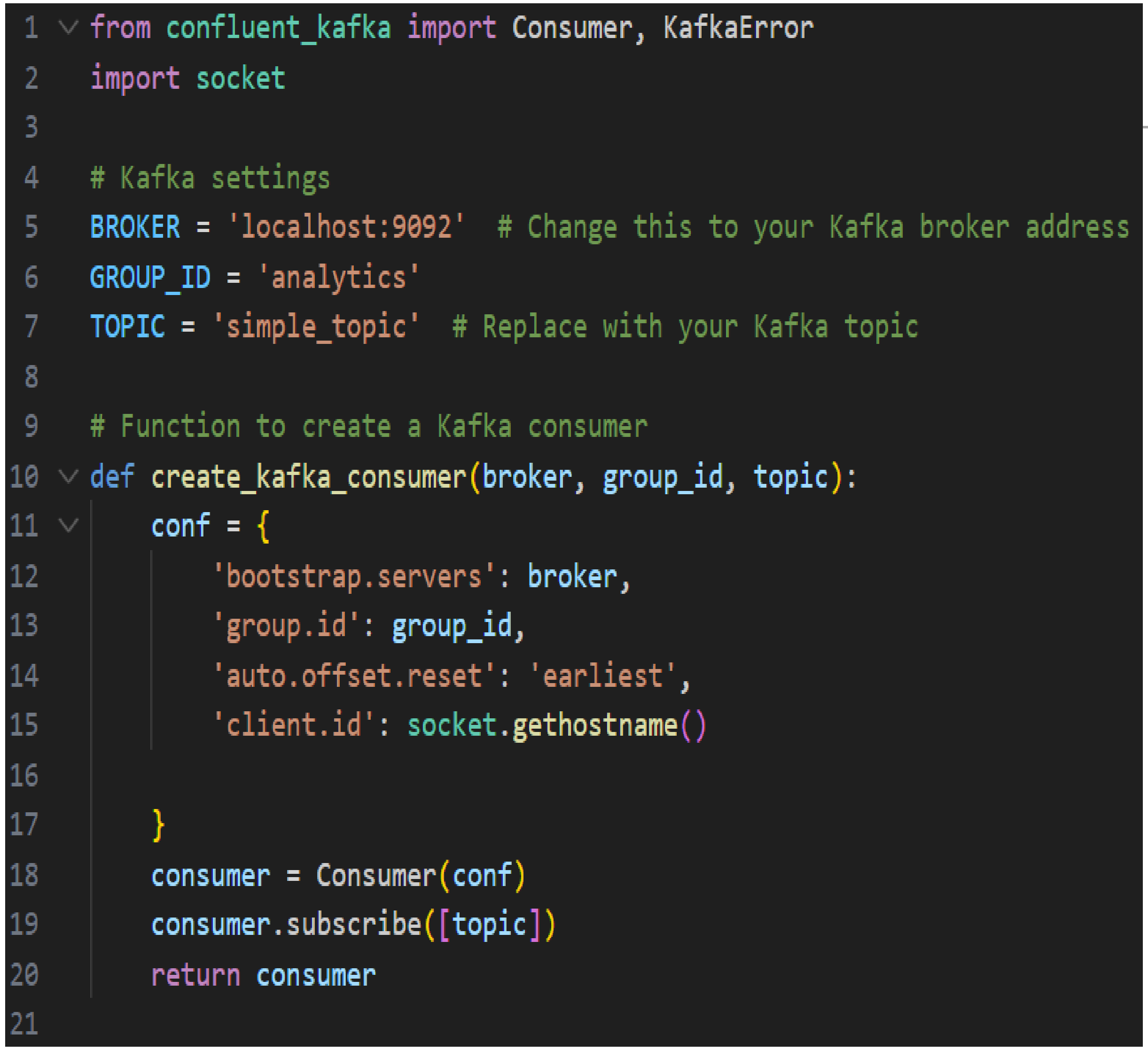


```python
from confluent_kafka import Consumer, KafkaError
import socket

# Kafka settings
BROKER = 'localhost:9092'  # Change this to your Kafka broker address
GROUP_ID = 'analytics'
TOPIC = 'simple_topic'  # Replace with your Kafka topic

# Function to create a Kafka consumer
def create_kafka_consumer(broker, group_id, topic):
    conf = {
        'bootstrap.servers': broker,
        'group.id': group_id,
        'auto.offset.reset': 'earliest',
        'client.id': socket.gethostname()

    }
    consumer = Consumer(conf)
    consumer.subscribe([topic])
    return consumer

```



```python
# Display data from Kafka
def display_kafka_data():
    consumer = create_kafka_consumer(BROKER, GROUP_ID, TOPIC)
    while True:
        msg = consumer.poll(timeout=1.0)
        if msg is None:
            continue
        if msg.error():
            if msg.error().code() == KafkaError._PARTITION_EOF:
                continue
            else:
                print(msg.error())
                break

        key = msg.key().decode('utf-8')
        value = msg.value().decode('utf-8')

        # Display message with an icon
        print ("New Message Alert!!")
        print (f"Key: {key} and Value: {value}")
    consumer.close()

if __name__ == "__main__":
    display_kafka_data()
```

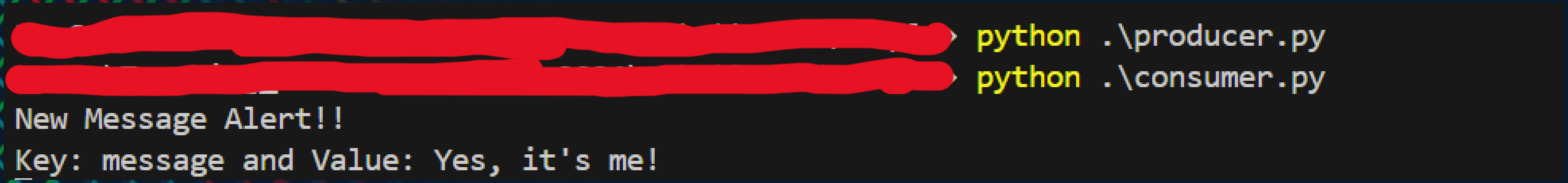
Consumer



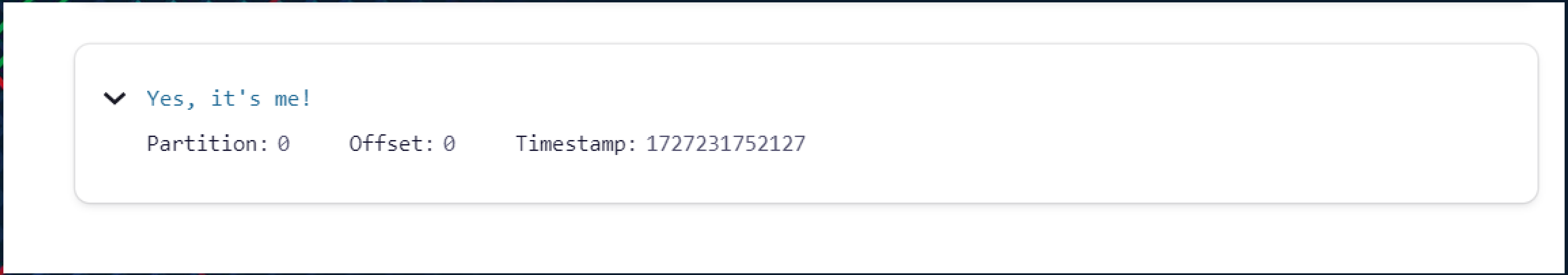
```python
from confluent_kafka import Consumer, KafkaError
import socket

# Kafka settings
BROKER = 'localhost:9092'  # Change this to your Kafka broker address
GROUP_ID = 'analytics'
TOPIC = 'simple_topic'  # Replace with your Kafka topic

# Function to create a Kafka consumer
def create_kafka_consumer(broker, group_id, topic):
    conf = {
        'bootstrap.servers': broker,
        'group.id': group_id,
        'auto.offset.reset': 'earliest',
        'client.id': socket.gethostname()

    }
    consumer = Consumer(conf)
    consumer.subscribe([topic])
    return consumer

```

Q. What is the benefit of using group.id?

Output



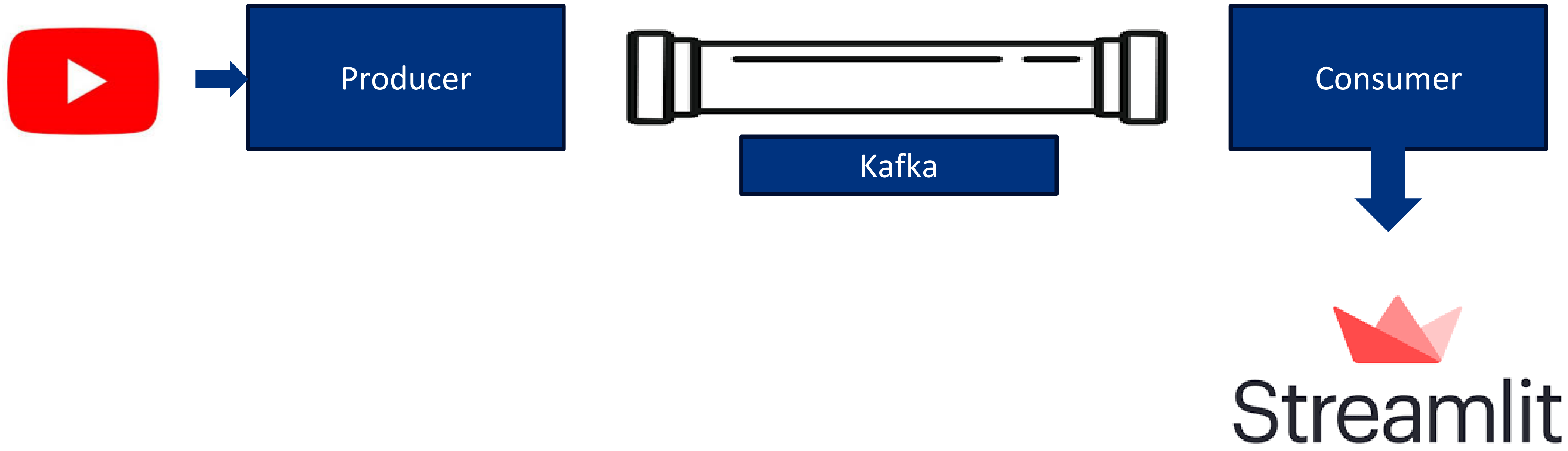
```
[redacted] python .\producer.py
[redacted] python .\consumer.py
New Message Alert!!
Key: message and Value: Yes, it's me!
```



```
Yes, it's me!
Partition: 0     Offset: 0     Timestamp: 1727231752127
```

Carnegie Mellon University

Producer/Consumer Example (2)



Digression: What is Streamlit?

- Streamlit is a web application framework for Python web applications.
- Provides built-in methods for handling user inputs like text and dates.
- Enables displaying interactive graphs with popular Python graphing libraries.

Digression: Why Streamlit?

- Static visualizations are limited for complex analyses requiring user input.
- Word documents or exported Jupyter notebooks lack user interaction and challenge reproducibility.
- Building a web application with Flask or Django and deploying on the Cloud is complex and time-consuming.
- Many traditional methods are slow, lack user input handling, or aren't optimal for decision-making.

Digression: More on Streamlit

- Streamlit runs our Python files from top to down as a script, so we can perform data manipulation with libraries such as pandas in the same way that we might do it in a Jupyter notebook or a regular Python script.
- Develop in Streamlit and use st.write() as a debugger.
- Explore in Jupyter and then copy to Streamlit.
- If you face issues running Streamlit on your M1/M2 machines, you can try the VM or different installation options: https://docs.streamlit.io/get-started/installation

Streamlit cheat sheet

streamlit.io

This cheat sheet is a summary of the docs

I also recommend streamlitopedia

How to install and import

```
$ pip install streamlit
```

Import convention
```
>>> import streamlit as st
```

Add widgets to sidebar

```
st.sidebar.<widget>

>>> my_val = st.sidebar.text_input('I:')
```

Command line

```
$ streamlit --help
$ streamlit run your_script.py
$ streamlit hello
$ streamlit config show
$ streamlit cache clear
$ streamlit docs
$ streamlit --version
```

Pre-release features

To access beta and experimental features

```
pip uninstall streamlit
pip install streamlit-nightly --upgrade
```

Magic commands

Magic commands allow you to implicitly `st.write()`

```
''' _This_ is some __Markdown__ '''

a=3
'a', a

'dataframe:', data
```

Display text

```
st.text('Fixed width text')
st.markdown('_Markdown_') # see *
st.latex(r''' e^{i\pi} + 1 = 0 ''')
st.write('Most objects') # df, err, func, keras!
st.write(['st', 'is <', 3]) # see *
st.title('My title')
st.header('My header')
st.subheader('My sub')
st.code('for i in range(8): foo()')
```

* optional kwarg unsafe_allow_html = True

Display data

```
st.dataframe(data)
st.table(data.iloc[0:10])
st.json({'foo':'bar','fu':'ba'})
```

Display charts

```
st.line_chart(data)
st.area_chart(data)
st.bar_chart(data)
st.pyplot(fig)
st.altair_chart(data)
st.vega_lite_chart(data)
st.plotly_chart(data)
st.bokeh_chart(data)
st.pydeck_chart(data)
st.deck_gl_chart(data)
st.graphviz_chart(data)
st.map(data)
```

Display media

```
st.image('./header.png')
st.audio(data)
st.video(data)
```

Display interactive widgets

```
st.button('Hit me')
st.checkbox('Check me out')
st.radio('Radio', [1,2,3])
st.selectbox('Select', [1,2,3])
st.multiselect('Multiselect', [1,2,3])
st.slider('Slide me', min_value=0, max_value=10)
st.text_input('Enter some text')
st.number_input('Enter a number')
st.text_area('Area for textual entry')
st.date_input('Date input')
st.time_input('Time entry')
st.file_uploader('File uploader')
st.beta_color_picker('Pick a color')
```

Use widgets' returned values in variables:

```
>>> for i in range(int(st.number_input('Num:'))): foo()
>>> if st.sidebar.selectbox('I:',['f']) == 'f': b()
>>> my_slider_val = st.slider('Quinn Mallory', 1, 88)
>>> st.write(slider_val)
```

Control flow

```
st.stop()
```

Display code

```
st.echo()

>>> with st.echo():
>>>     # Code below both executed and printed
>>>     foo = 'bar'
>>>     st.write(foo)
```

Display progress and status

```
st.progress(progress__variable_1_to_100)

st.spinner()

>>> with st.spinner(text='In progress'):
>>>     time.sleep(5)
>>>     st.success('Done')

st.balloons()
st.error('Error message')
st.warning('Warning message')
st.info('Info message')
st.success('Success message')
st.exception(e)
```

Placeholders, help, and options

```
st.empty()

>>> my_placeholder = st.empty()
>>> my_placeholder.text('Replaced!')

st.help(pandas.DataFrame)

st.get_option(key)
st.set_option(key)

st.beta_set_page_config(layout='wide')
```

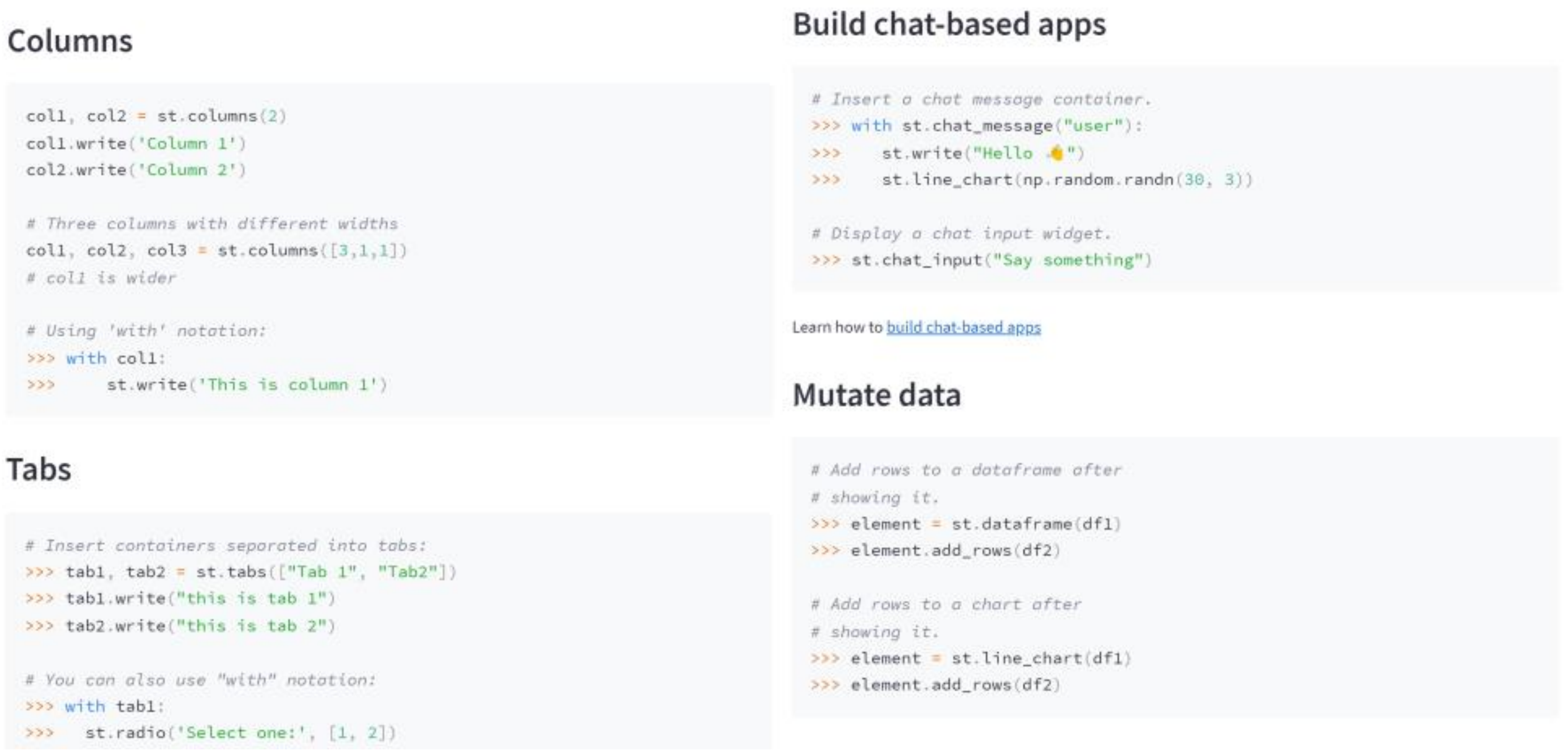
Mutate data

```
DeltaGenerator.add_rows(data)

>>> my_table = st.table(df1)
>>> my_table.add_rows(df2)

>>> my_chart = st.line_chart(df1)
>>> my_chart.add_rows(df2)
```

Optimize performance

```
@st.cache

>>> @st.cache
... def foo(bar):
...     # Mutate bar
...     return data
...
>>> d1 = foo(ref1)
>>> # Executes as first time
>>>
>>> d2 = foo(ref1)
>>> # Does not execute; returns cached value, d1==d2
>>>
>>> d3 = foo(ref2)
>>> # Different arg, so function executes
```

Streamlit cheat sheet v1.0 | August 2020
https://github.com/daniellewisDL/streamlit-cheat-sheet



Columns

```
col1, col2 = st.columns(2)
col1.write('Column 1')
col2.write('Column 2')

# Three columns with different widths
col1, col2, col3 = st.columns([3,1,1])
# col1 is wider

# Using 'with' notation:
>>> with col1:
>>>     st.write('This is column 1')
```

Tabs

```
# Insert containers separated into tabs:
>>> tab1, tab2 = st.tabs(["Tab 1", "Tab2"])
>>> tab1.write("this is tab 1")
>>> tab2.write("this is tab 2")

# You can also use "with" notation:
>>> with tab1:
>>>   st.radio('Select one:', [1, 2])
```

Build chat-based apps

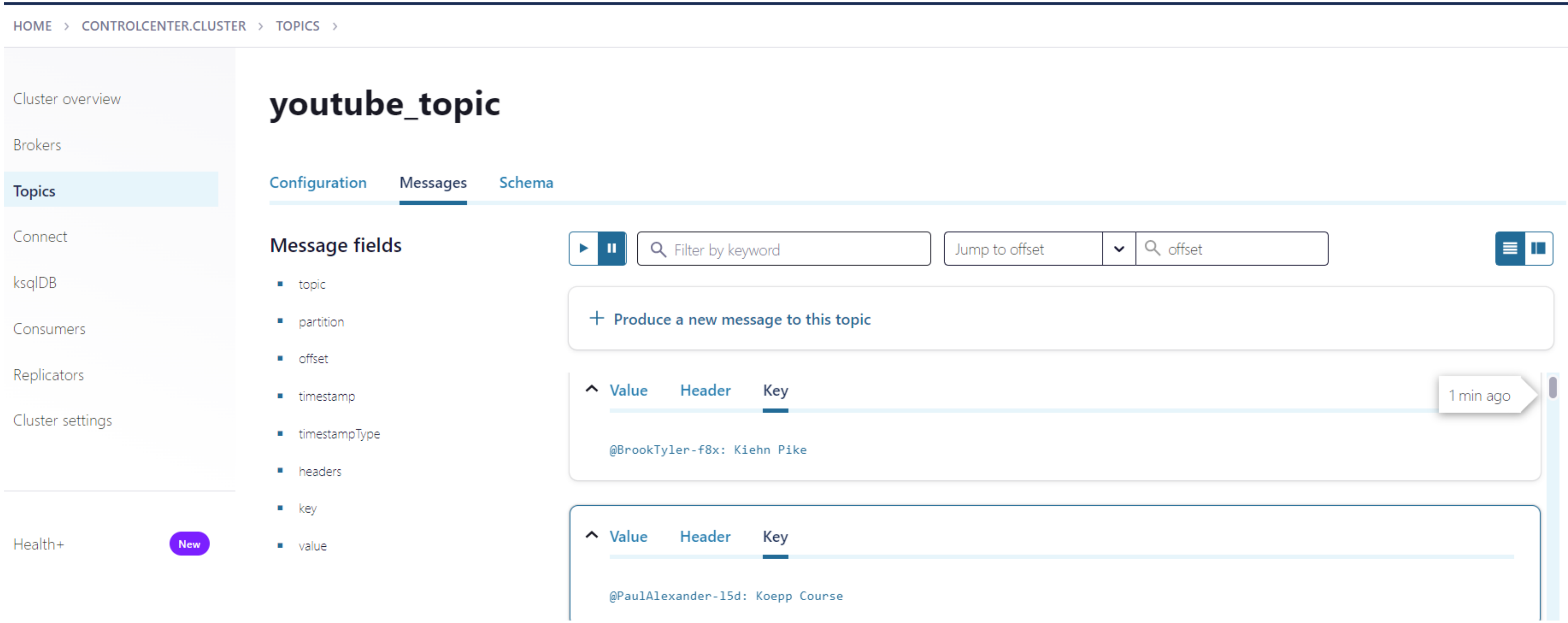
```
# Insert a chat message container.
>>> with st.chat_message("user"):
>>>    st.write("Hello 👋")
>>>    st.line_chart(np.random.randn(30, 3))

# Display a chat input widget.
>>> st.chat_input("Say something")
```

Learn how to [build chat-based apps]

Mutate data

```
# Add rows to a dataframe after
# showing it.
>>> element = st.dataframe(df1)
>>> element.add_rows(df2)

# Add rows to a chart after
# showing it.
>>> element = st.line_chart(df1)
>>> element.add_rows(df2)
```

Producer/Consumer Example (2)



youtube_topic

Message fields

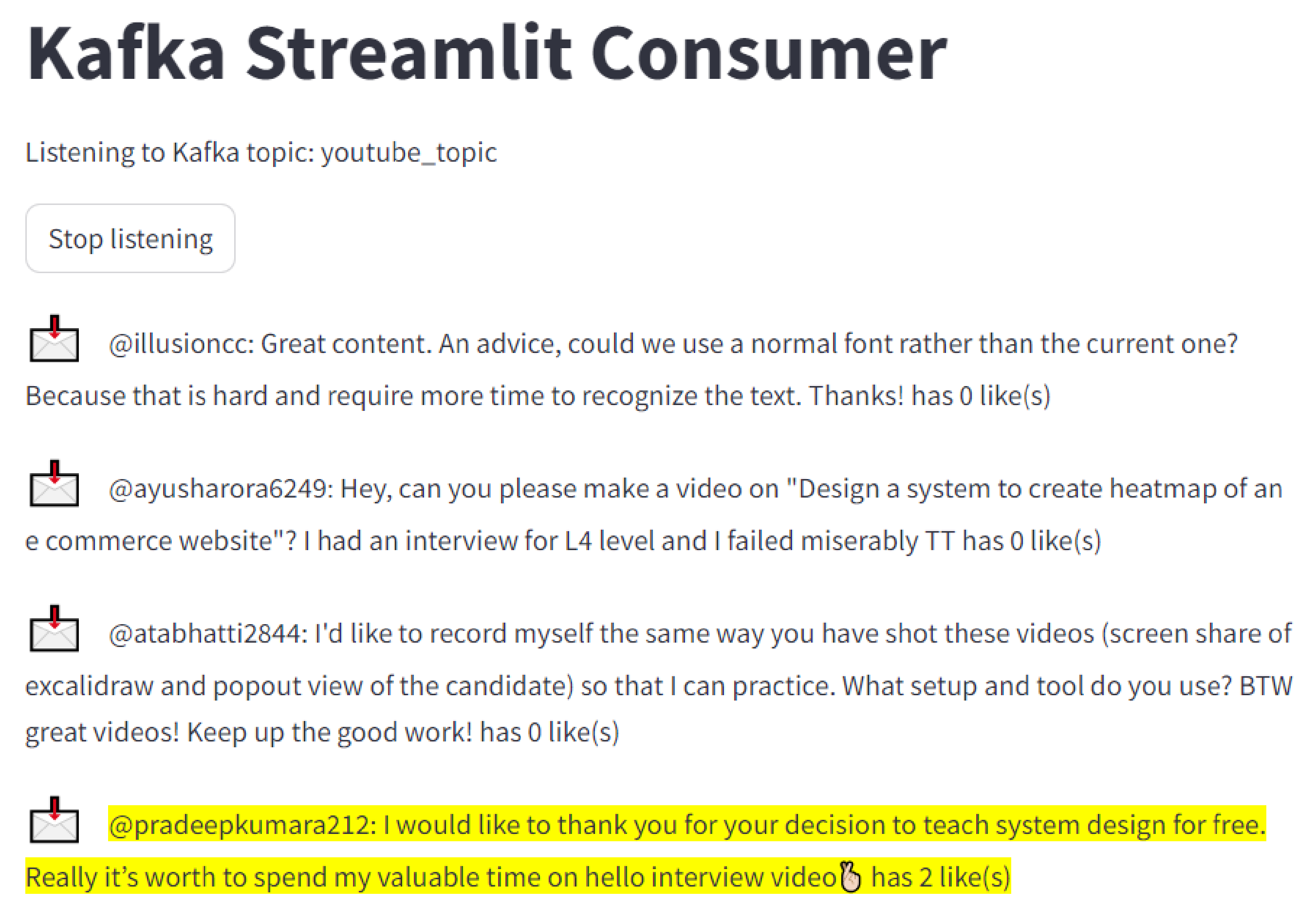
- topic
- partition
- offset
- timestamp
- timestampType
- headers
- key
- value

```
@BrookTyler-f8x: Kiehn Pike
```

```
@PaulAlexander-15d: Koepp Course
```

Carnegie Mellon University

Producer/Consumer Example (2)
Sample Output



Kafka Streamlit Consumer

Listening to Kafka topic: youtube_topic

📩 @illusioncc: Great content. An advice, could we use a normal font rather than the current one? Because that is hard and require more time to recognize the text. Thanks! has 0 like(s)

📩 @ayusharora6249: Hey, can you please make a video on "Design a system to create heatmap of an e commerce website"? I had an interview for L4 level and I failed miserably TT has 0 like(s)

📩 @atabhatti2844: I'd like to record myself the same way you have shot these videos (screen share of excalidraw and popout view of the candidate) so that I can practice. What setup and tool do you use? BTW great videos! Keep up the good work! has 0 like(s)

📩 @pradeepkumara212: I would like to thank you for your decision to teach system design for free. Really it's worth to spend my valuable time on hello interview video🤞 has 2 like(s)

Exercise: Deploy Confluent Kafka to GKE:

- Follow the below guidelines to deploy Kafka to GKE:
    - https://docs.confluent.io/operator/current/co-quickstart.html