



snowflake[®] +  = Streamlit

Streamlit Overview

Diana Shaw & Jeff Carpenter
September 2022

Enormous opportunity to help put data to work

73%

of data is rendered
effectively unused
because it's just too hard
to build the tools to use it

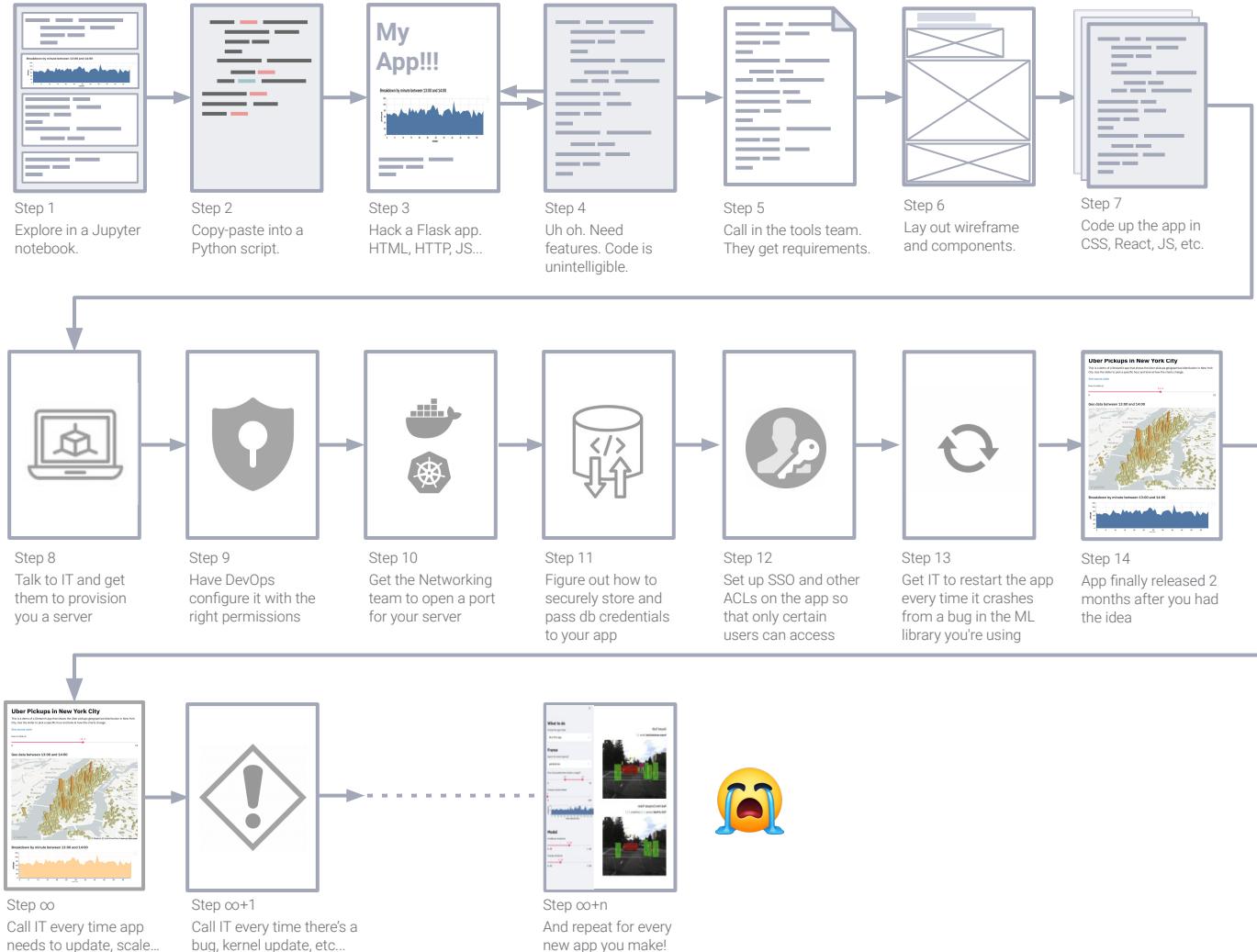
[Source](#)



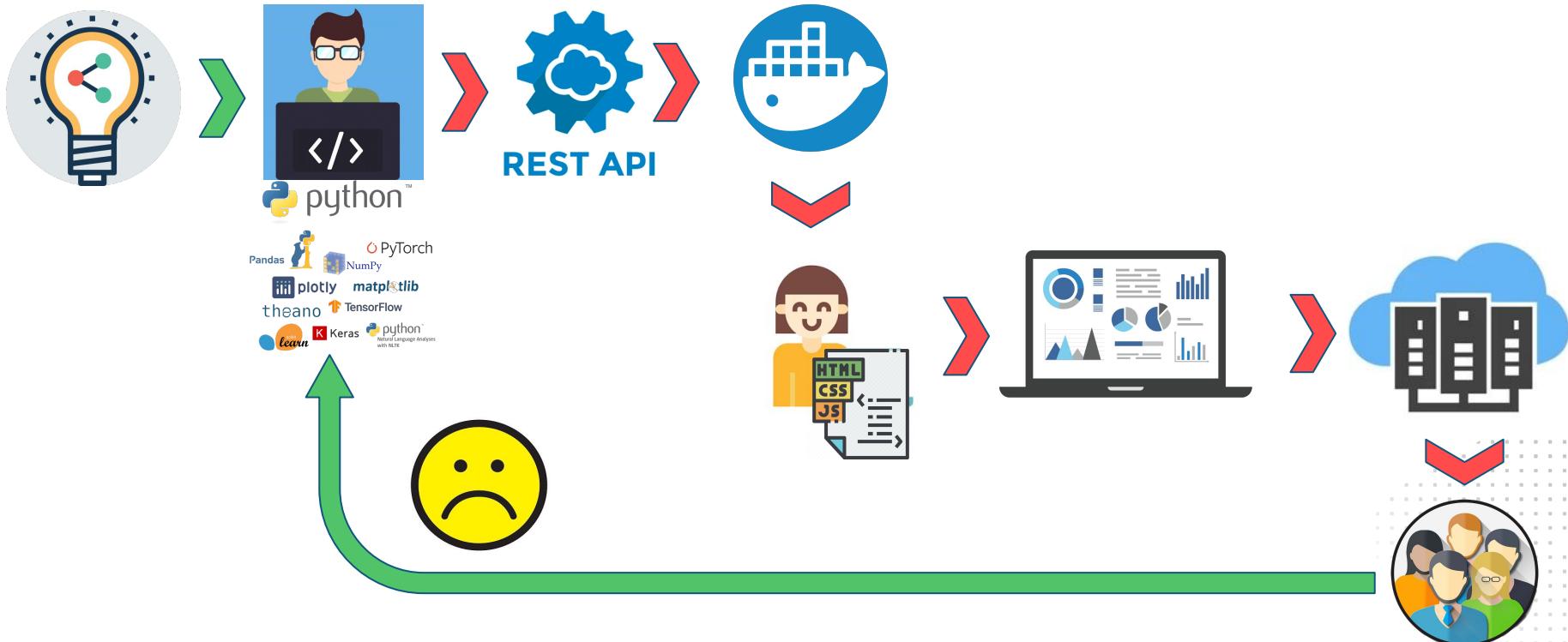
© 2022 Snowflake Inc. All Rights Reserved

Getting from data to action is hard

Only a fraction of data questions are ever answered since companies haven't built tools to capture or analyze the data. 73% of data is rendered effectively unused* because it's just too hard to build the tools to use it.



The before times...



[Source](#)



© 2022 Snowflake Inc. All Rights Reserved

WHAT IF BUILDING TOOLS WERE AS EASY AS WRITING PYTHON SCRIPTS?

The image shows a Streamlit application running in a browser at localhost:8501. The main title is "Pickup locations between 4:00 and 5:00". Below the title is a map of New York City with several red hexagonal markers indicating pickup locations. A sidebar on the left contains Python code for generating the map.

```
# FETCHING DATA
use_query = "USE WAREHOUSE S"
QUERY = """
SELECT STARTTIME, START_STATION_LATITUDE AS LAT, START_STATION_LONGITUDE AS LON FROM DANIE
"""

cur.execute(use_query)
cur.execute(QUERY)
results = cur.fetchall()

# TRANSFORMING DATA FOR HOURLY
columns = ['Starttime', 'Lat', 'Lon']
data = pd.DataFrame(results, columns=columns)
data['hour'] = data['Starttime'].dt.hour

hour_selected = st.sidebar.slider("Select hour of pickup", 0, 23)

data_hour = data[data.hour == hour_selected]

# CREATING MAP STYLE
def map(data, lat, lon, zoom, scale, type, pitch) -> pkdeck:
    MAP_STYLE = "mapbox://styles/mapbox/streets-v11"
    view_state = dict(latitude=lat, longitude=lon, zoom=zoom, pitch=pitch)
    data_config = dict(data=data, get_position=["Lon", "Lat"])
    hex_config = dict(radius=75, elevation_scale=scale, elevation_range=[0, 1000])
    scatter_config = dict(get_color=[0, 78, 255], get_radius=100)
    mouse_config = dict(pickable=False, extruded=True)
    if type:
        layer = pk.Layer("HexagonLayer", **data_config, **hex_config, **mouse_config)
    else:
        layer = pk.Layer("ScatterplotLayer", **data_config, **scatter_config, **mouse_config)
    return pk.Deck(map_style=MAP_STYLE, initial_view_state=view_state, layers=[layer])

# SHOW PICKUP LOCATIONS
st.header("Pickup locations between {hour_selected}:00 and {(hour_selected + 1) % 24}:00")
st.write(map(data_hour, 40.7359, -74.0164, 12, 5, True, 40))
st.write("")

# SLICING DATA FOR TOP PICKUP LOCATIONS
def data_top(number):
    data_top = data_hour[['Lat', 'Lon', 'hour']].groupby(['Lat', 'Lon']).count().reset_index().re
    data_top = data_top[['Lat', 'Lon']].head(number)
    return data_top

top_locations = st.sidebar.number_input("Select number of top locations", min_value=1, max_value=60)
data_top = data_top[int(top_locations)]

# FOLIUM MAP CODE
def folium_map(lat, lon):
    loc = folium.Map(location=[lat, lon], zoom_start=16)
    folium.Marker([
```



Using Streamlit



[Source](#)

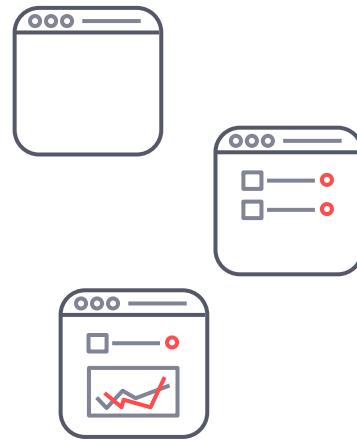


© 2022 Snowflake Inc. All Rights Reserved

Iterate quickly with Streamlit's fast prototyping flow



Get coding in minutes



Share apps with your team and iterate on them together



Publish apps to your company, get feedback, and quickly make updates to expand and improve them



Streamlit is built by and for data practitioners

Doing Data Science

Coding in Python

Looking for speed!

If they are in the Snowpark preview for Python that's a good signal! If they are not using Python already, then they are probably better off (at the moment) using their existing workflows.

There are many options for making data apps. Snowflake supports and will continue to support lots of options. You don't have to remake apps in Streamlit, but it's a good place to start if you're in Python and want something fast.



Common use cases for Streamlit

Rapid prototyping

Data science teams quickly build data apps in Python to test hypotheses, gather feedback, and iterate quickly to deliver insights to their business stakeholders.

Demo-ing work

Creating tools for business users

Data teams struggle to share the outputs of their models in an easily consumable way for their stakeholders. Folks from sales, operations, marketing, and support can often benefit from interacting with models the data team has built. Streamlit creates a fast and easy way to create new tools from data that empower business users.



Streamlit is an open-source Python library built on 3 core principles

(1)

Embrace Python
scripting

(2)

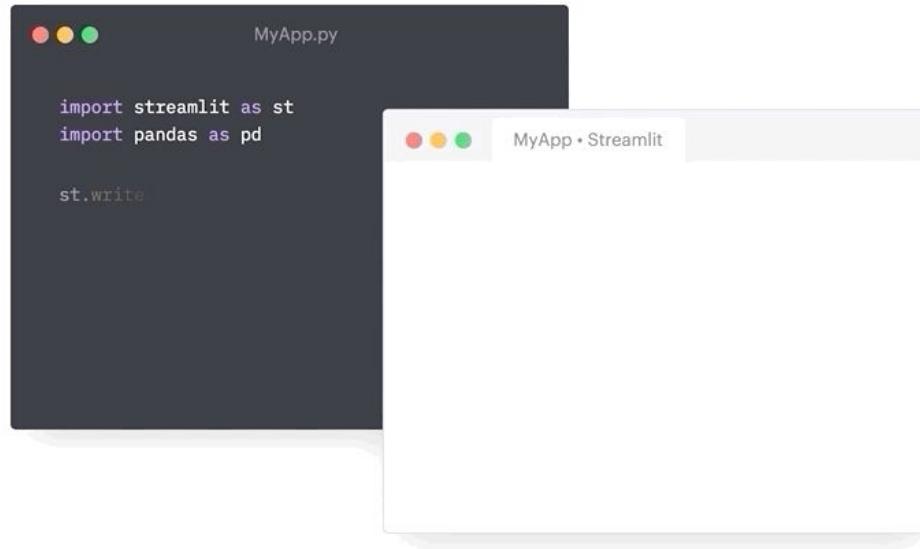
Treat widgets as
variables

(3)

Reuse data and
computation



1. Embrace Python scripting



Streamlit lets you go straight from Python script to web app

```
import streamlit as st
import pandas as pd
import altair as alt

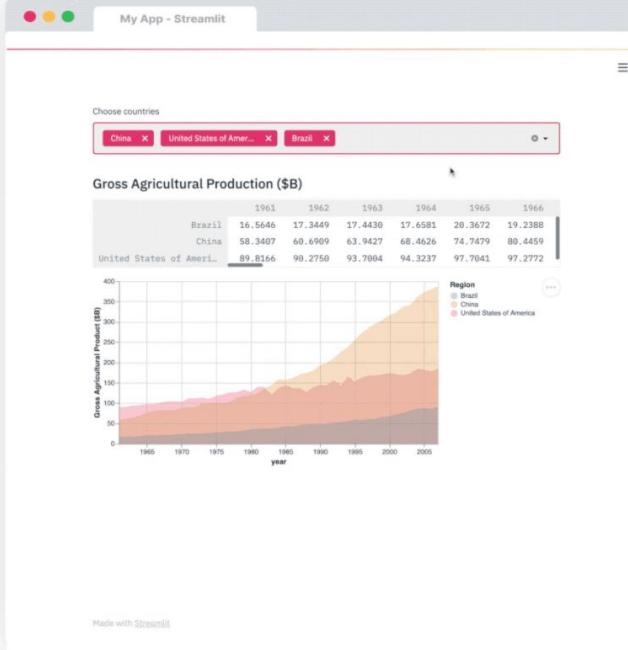
@st.cache
def get_UN_data():
    AWS_BUCKET_URL = "https://streamlit-demo-data.s3-us-west-2.amazonaws.com"
    df = pd.read_csv(AWS_BUCKET_URL + "/agri.csv.gz")
    return df.set_index("Region")

df = get_UN_data()

countries = st.multiselect(
    "Choose countries", list(df.index), ["China", "United States of America"]
)

data = df.loc[countries]
data /= 1000000.0
st.write("### Gross Agricultural Production ($B)", data.sort_index())

data = data.T.reset_index()
data = pd.melt(data, id_vars=["index"]).rename(
    columns={"index": "year", "value": "Gross Agricultural Product ($B)"}
)
chart = (
    alt.Chart(data)
    .mark_area(opacity=0.3)
    .encode(
        x="year",
        y=alt.Y("Gross Agricultural Product ($B):Q", stack=None),
        color="Region:N",
    )
)
st.altair_chart(chart, use_container_width=True
```



From code to app in minutes!



And it works with all your favorite Python libraries



DECK.GL

pandas
 $y_t = \beta^T x_t + \mu_t + \epsilon_t$

Vega-Lite

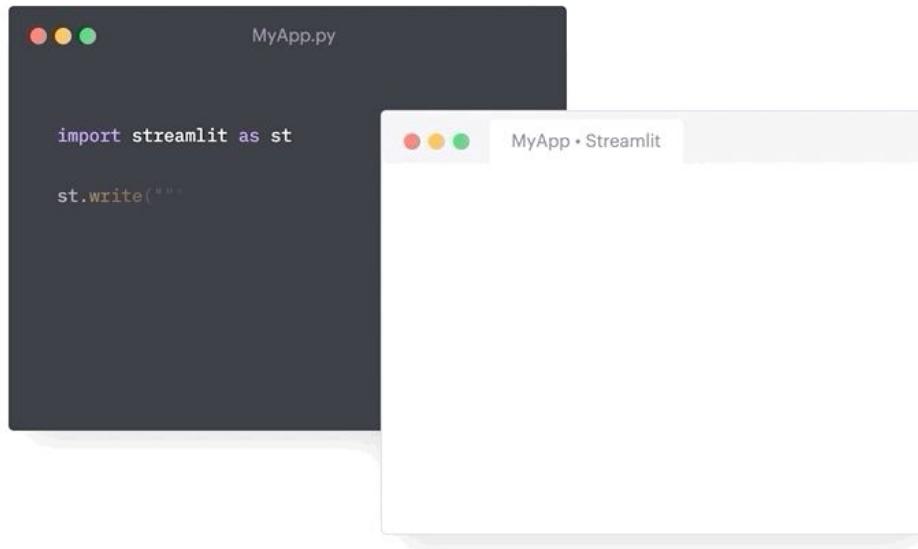
matplotlib



And many more!



2. Treat widgets as variables



It's easy to add in interaction, layout, and themes

The image displays six Streamlit UI components arranged in two columns:

- Top Left:** A slider titled "Pick a number" with a value of 10. Below it is the code: `number = st.slider("Pick a number", 0, 100)`.
- Top Middle:** A file uploader titled "Pick a file" with a placeholder "Drag and drop files here". Below it is the code: `file = st.file_uploader("Pick a file")`.
- Top Right:** A color picker titled "Pick a color" showing a red square. Below it is the code: `color = st.color_picker("Pi`.
- Middle Left:** An Altair chart showing a distribution of values across categories. Below it is the code: `st.altair_chart(my_chart)`.
- Middle Middle:** A pet selection using radio buttons. "Dog" is selected. Below it is the code: `pet = st.radio("Pick a pet", ["Dog", "Cat", "Bird"])`.
- Middle Right:** A date picker for May 2021. The 27th is highlighted with a red circle. Below it is the code: `date = st.date_input("Pick a date")`.

A curved arrow points from the "color" component towards the date picker, indicating a theme or styling connection.

Choose from light
or dark theme or
set your own
custom colors



3. Reuse data and computation

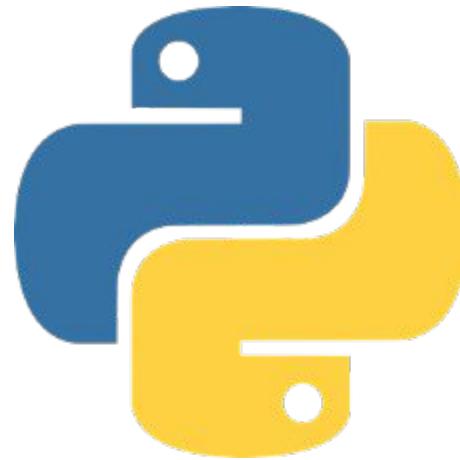
Instantiate DB/model
connectors only once

```
● ● ●  
@st.experimental_singleton  
def sfconn(creds):  
    return snowflake.connector.connect(**creds)  
  
@st.experimental_memo  
def query_database(sql, _connector):  
    return pd.read_sql(sql, _connector)  
  
conn = sfconn(**st.secrets['sfdevrel'])  
df = query_database("SELECT * FROM users", conn)
```

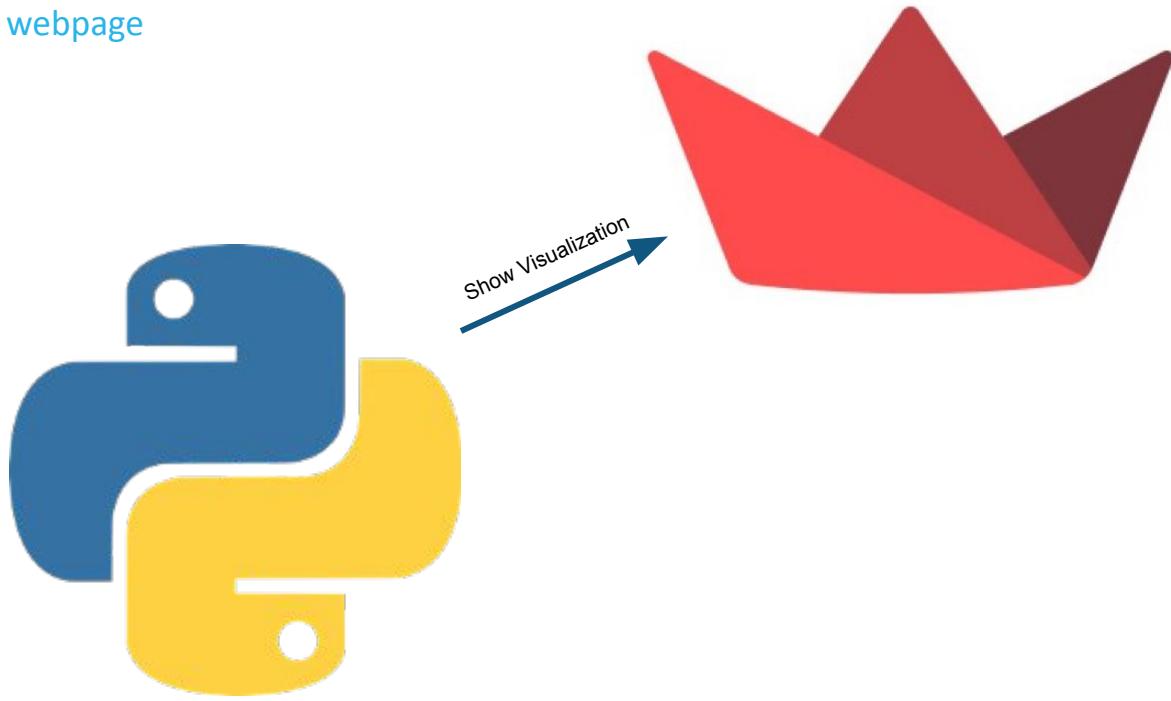
Store query results in cache



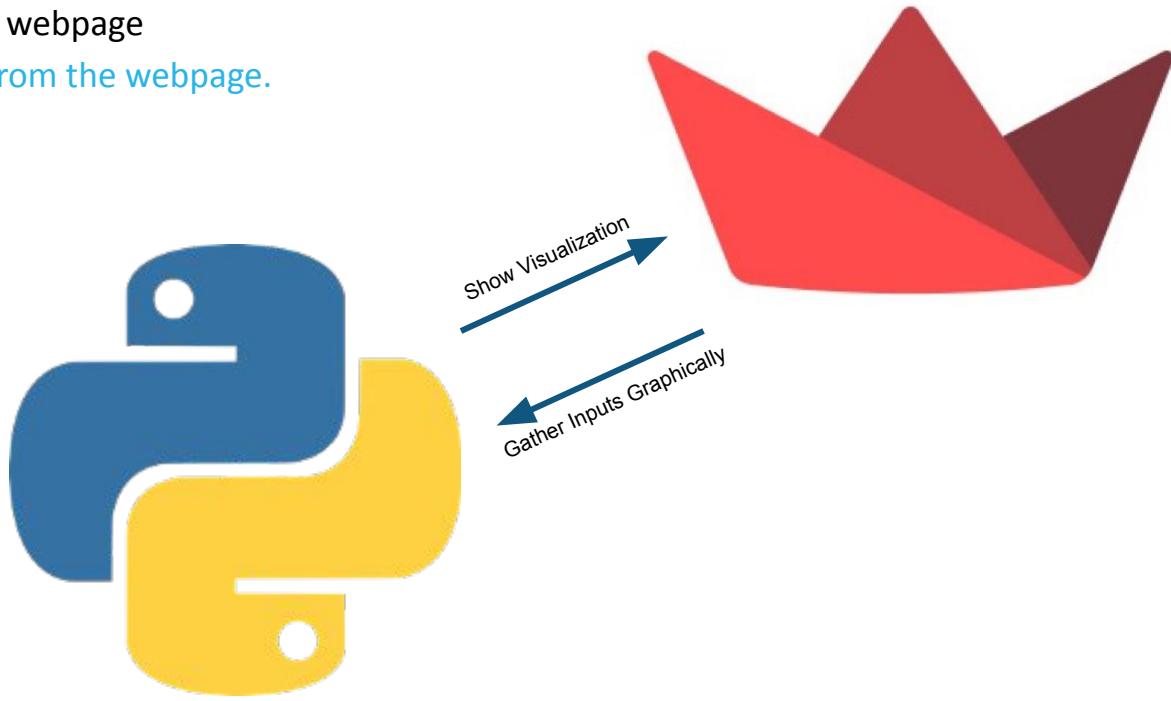
A Streamlit app is just a Python program



A Streamlit app is just a Python program
that uses the Streamlit library to display in a webpage

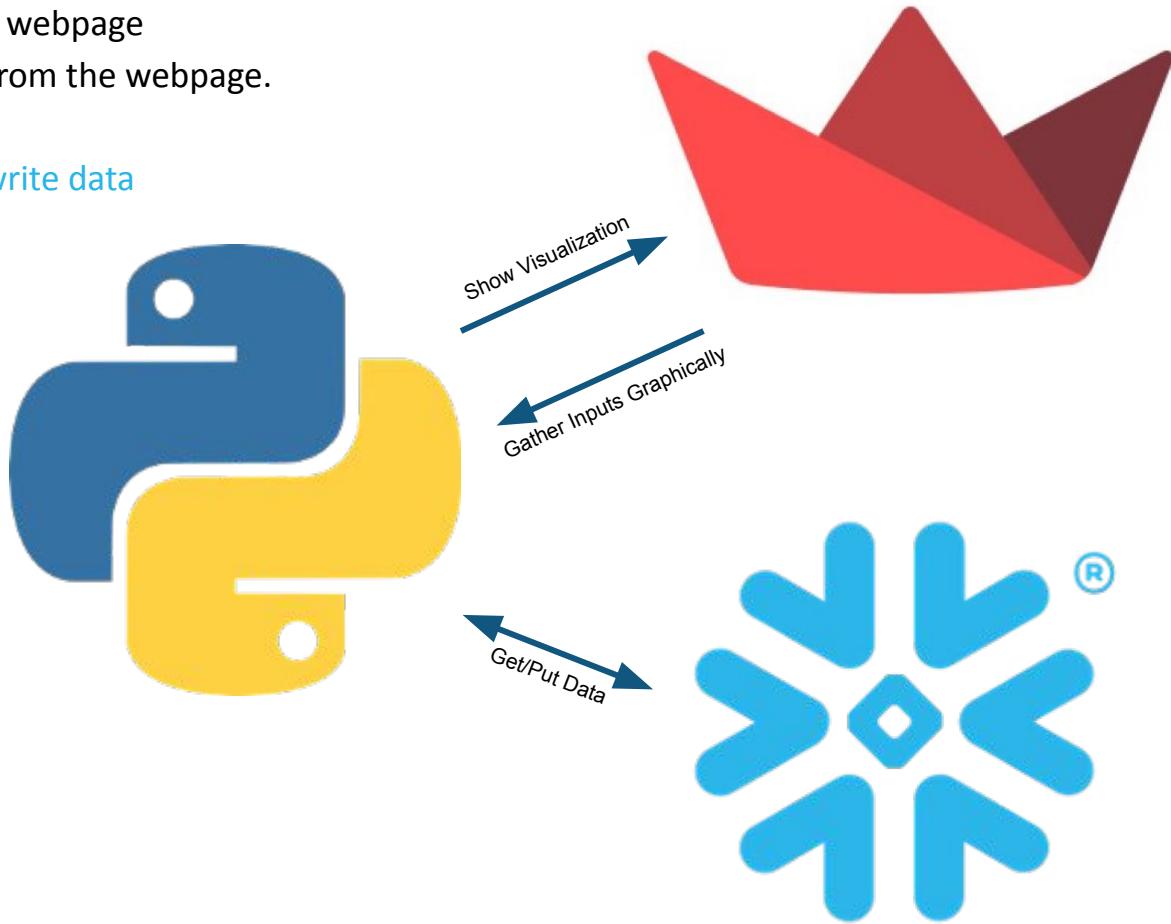


A Streamlit app is just a Python program
that uses the Streamlit library to display in a webpage
and uses the Streamlit library to get inputs from the webpage.

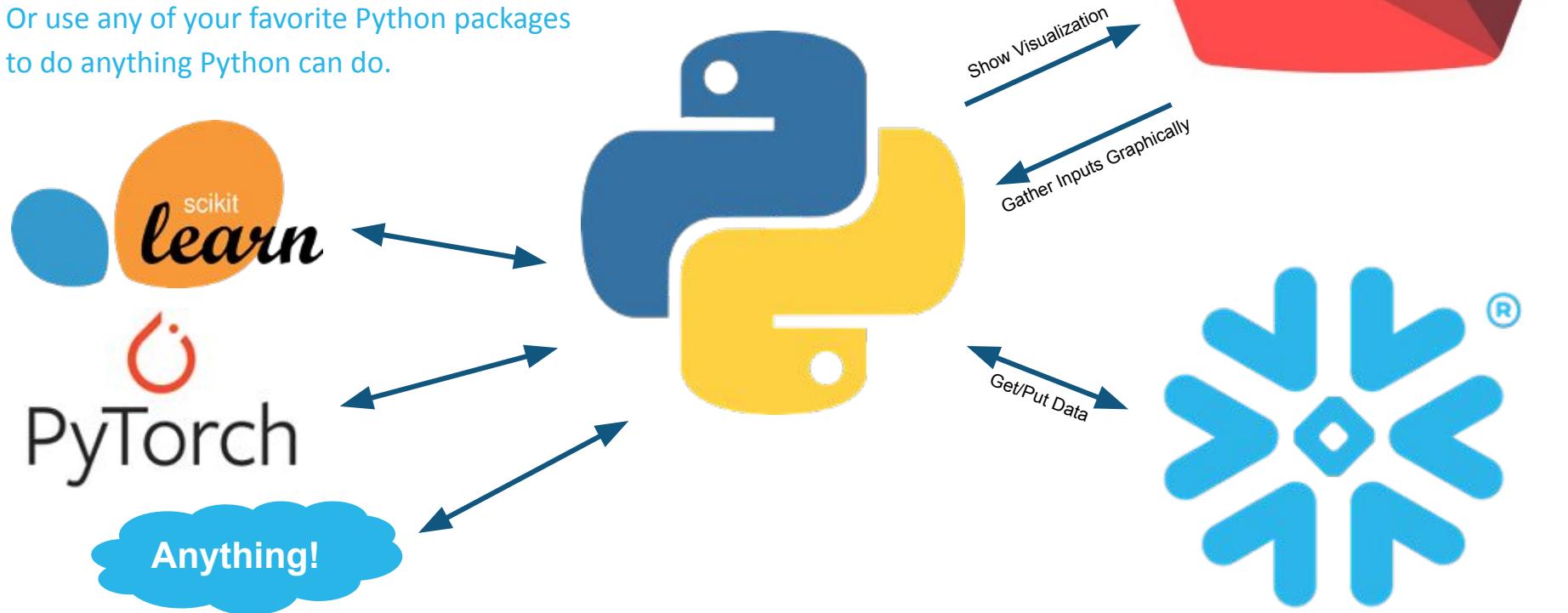


A Streamlit app is just a Python program
that uses the Streamlit library to display in a webpage
and uses the Streamlit library to get inputs from the webpage.

It can do anything Python can do
including connecting to Snowflake to read/write data



A Streamlit app is just a Python program
that uses the Streamlit library to display in a webpage
and uses the Streamlit library to get inputs from the webpage.
It can do anything Python can do
including connecting to Snowflake to read/write data
Or use any of your favorite Python packages
to do anything Python can do.



It only takes minutes to get started!

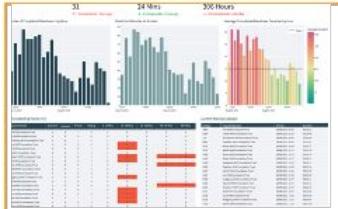
```
$ pip install streamlit  
$ streamlit hello
```

See our [docs](#) for more getting started info
with the Streamlit open-source library

Internal resources here.

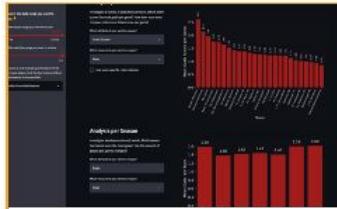


Streamlit Drives Rapid Insights



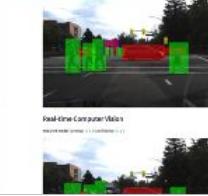
Monitoring apps

Tools that display trends and real-time insights.



Analysis tools

Apps that use machine learning to analyze new data types.



Detection apps

Tools that use computer vision to detect and track objects.



Prediction tools

Apps that predict prices, stock shortages, quality issues, etc.



Explanatory apps

Apps that analyze large datasets and present easy-to-understand insights.



Interactive tools

Tools that allow you to interact with the data to gain new insights.



COMING SOON: Streamlit + Snowflake = 😍

The near-term focus for Streamlit + Snowflake is about hosting Streamlit apps inside the Snowflake UI (i.e., Snowsight)

- Editing
 - Sharing
 - Permissions

• Make it easy!

IN DEVELOPMENT EDITION *

ACCOUNTADMIN XSMALL Share Updated 6 seconds ago

```
APP_DB.ML_TRAINING ~ Python

39     return data.to_pandas(), alloc.to_pandas(), rois.to_pandas(), last
37
38 @st.cache()
39 def predictBudgets():
40     pred = session.sql("SELECT predict_roi(array_construct((budget / 100000) * 100, 1))").values[0]
41     pred = pred["PREDICTED_ROI"].values[0] / 100000
42     change = round((pred / rois["ROI"].iloc[-1]) - 1) * 100, 1)
43     return pred, change
44
45 def chart(chart_data):
46     base = alt.Chart(chart_data).encode(alt.X("MONTH", sort=List(calend
47     bars = base.mark_bar().encode(y=alt.Y("BUDGET", title="Budget", sc
48     lines = base.mark_line(size=3).encode(y=alt.Y("ROI", title="Revenue
49     points = base.mark_point(strokeWidth=3).encode(y=alt.Y("ROI"), str
50     chart = alt.Layer(bars, lines + points).resolveScale(y="independe
51     st.altair_chart(chart, use_container_width=True)
52
53 st.image("https://i.imgur.com/dBDOHH3.png", width=80)
54 st.title("SportsCo Ad Spend Optimizer")
55 session = st.session_state.get("snowflake_demo.get_snowpark_session()")
56 data, alloc, rois, last_alloc = load()
57 last_alloc = last_alloc.replace(channels_upper, channels)
58
59 st.header("Advertising budgets")
60 col1, _, col2 = st.columns([4, 1, 4])
61 budgets = []
62 for alloc, col in zip(last_alloc.iterrows(), [col1, col1, col2, col2
63     budgets.append(col.number_input(alloc.CHANNEL, 0, 100, alloc.BUDGE
64
65 st.write("")
66 st.header("Predicted revenue")
67 pred, change = predictBudgets()
68 st.metric("$ " + str(pred) + " million", f"(change: {change} % vs last month"
69 july = pd.DataFrame({"MONTH": ["July"], "CHANNEL": channels_upper})
70 chart_data.append(july).reset_index(drop=True).replace(channels_upper,
71
72 if st.button("Save to Snowflake"):
73     with st.spinner("Inflating balloons..."):
74         session.sql("INSERT INTO FINANCE.PUBLIC.BUDGET_ALLOCATIONS_AND"
75         st.success("Saved budgets & prediction to your Snowflake account")
76         st.balloons()
```



SportsCo Ad Spend Optimizer

Advertising budgets

Search engine	Video		
35	- +	35	- +

Social media

Email			
50	- +	85	- +

Predicted revenue

\$ 8.22 million

↑ 0.0 % vs last month

Email Search engine Social media Video



COMING SOON: Streamlit + Snowflake = 😍

The screenshot shows the Snowflake UI interface. On the left, the sidebar displays various database and schema names such as ALOOMA, BITCOIN, CCHANGE_TEST_DB, DEMO_DB, DEMYSTDATA_COVID19_US_ZI..., HEALTHCARE_HIGI_SAMPLE, INFORMATION_SCHEMA, PUBLIC, and Streamlit. Within the Streamlit folder, two objects named streamlit-1 and streamlit-2 are listed. The main panel is titled "Untitled" and contains a single line of SQL code: "snowflake.demo * select 1". The bottom navigation bar includes tabs for Objects, Query, Results, and Chart, along with search and filter icons.



KEY TAKEAWAYS: Snowflake + Streamlit =

- ❖ Streamlit brings your data to life – build interactive apps with Python
- ❖ Today Snowflake customers can get started with the Streamlit Open Source Library
- ❖ Coming soon Snowflake integration to build, deploy, and share Streamlit apps in Snowflake



Join our community and see tons of examples in code!

<https://docs.streamlit.io>

<https://discuss.streamlit.io>

<https://blog.streamlit.io>

<https://streamlit.io/gallery>

