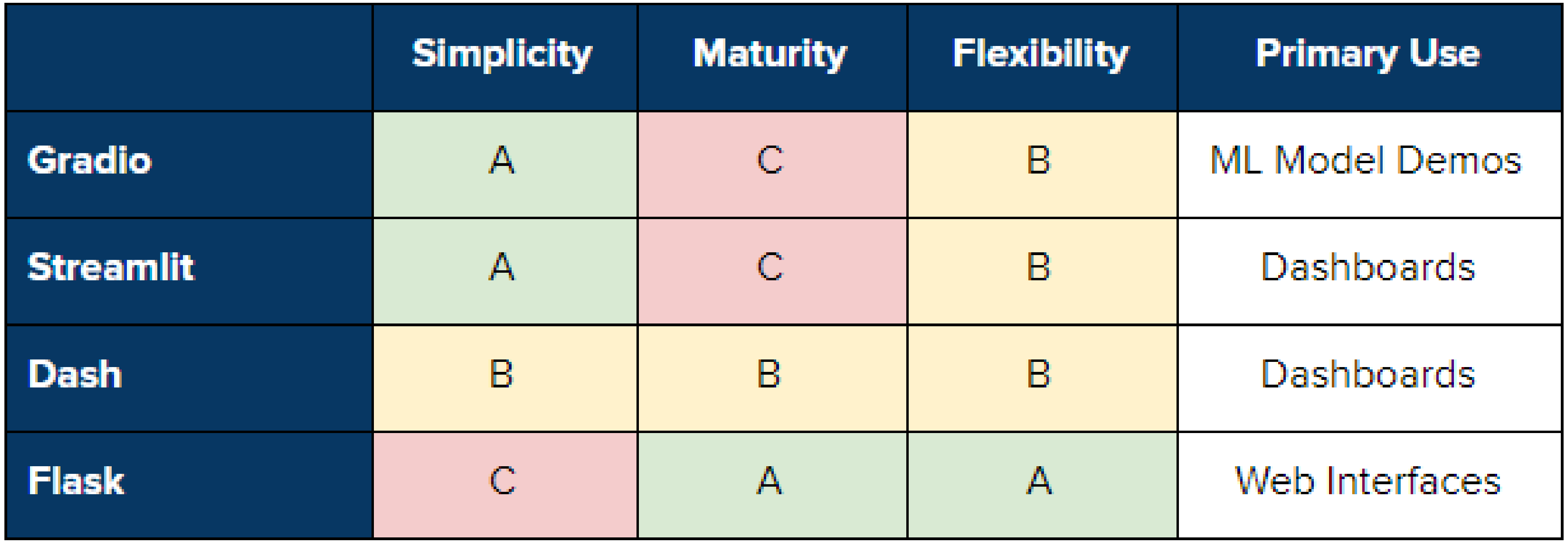
AFRICA DATA SCHOOL

Streamlit

The fastest way to build and share data apps

Streamlit is an open-source Python library that makes it easy to create and share beautiful, custom web apps for machine learning and data science. In just a few minutes you can build and deploy powerful data apps

Comparing several web UI tools for data science!



	Simplicity	Maturity	Flexibility	Primary Use
Gradio	A	C	B	ML Model Demos
Streamlit	A	C	B	Dashboards
Dash	B	B	B	Dashboards
Flask	C	A	A	Web Interfaces

https://towardsdatascience.com/gradio-vs-streamlit-vs-dash-vs-flask-d3defb1209a2

Install Streamlit

```
pip install streamlit
```

Import Streamlit

Now that everything's installed, let's create a new Python script and import Streamlit.

1. Create a new Python file named `first_app.py`, then open it with your IDE or text editor.
2. Next, import Streamlit.

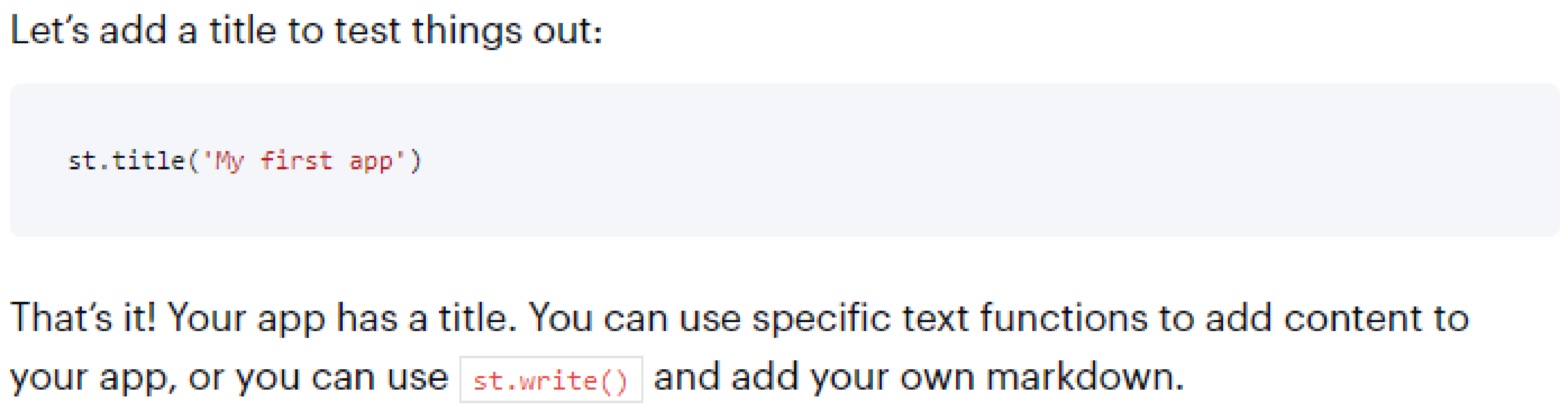
```python
import streamlit as st
# To make things easier later, we're also importing numpy and pandas for
# working with sample data.
import numpy as np
import pandas as pd
```

3. Run your app. A new tab will open in your default browser. It'll be blank for now. That's OK.

```
streamlit run first_app.py
```

Running a Streamlit app is no different than any other Python script. Whenever you need to view the app, you can use this command.

Add a title

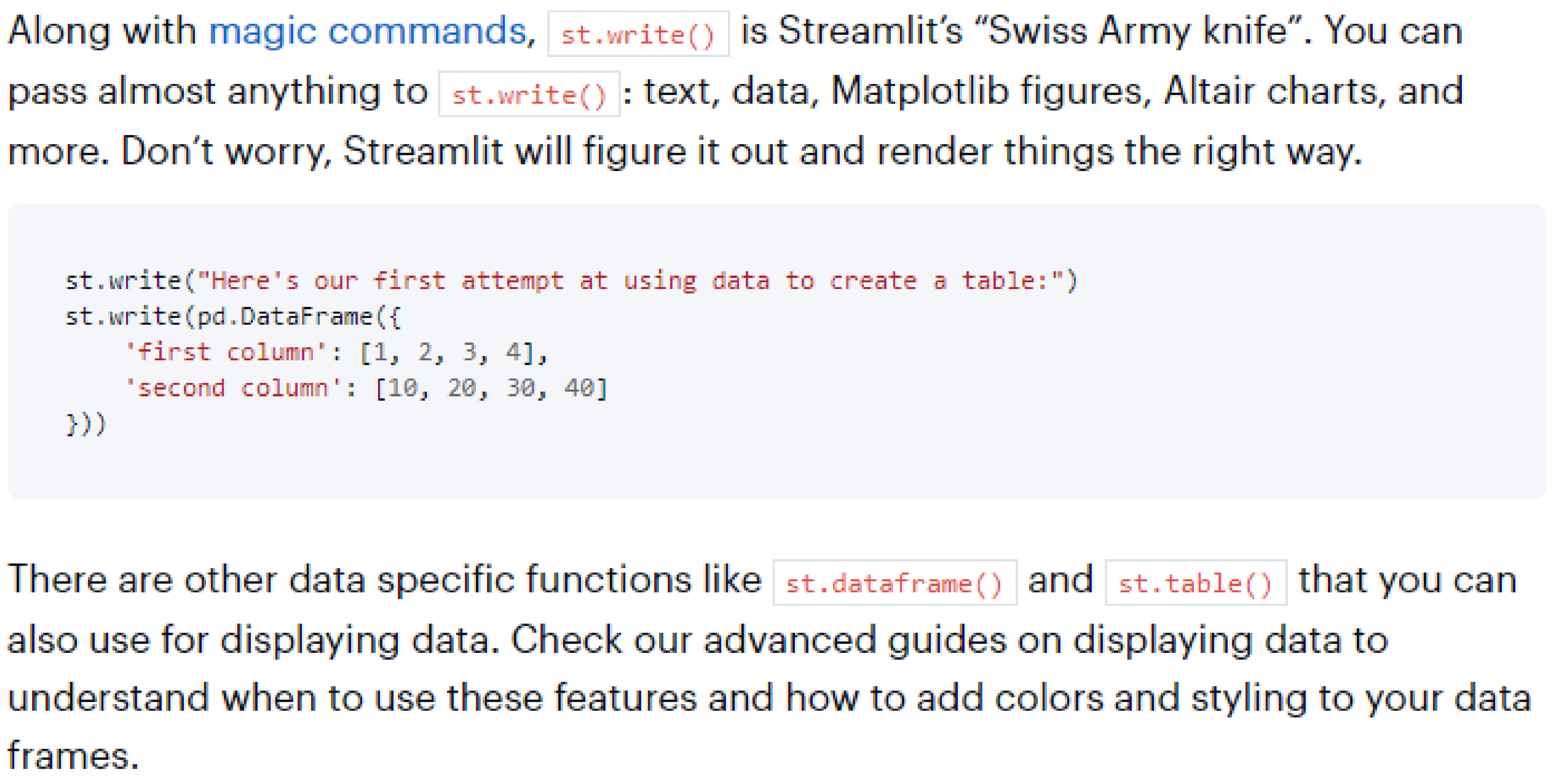


Let's add a title to test things out:

```
st.title('My first app')
```

That's it! Your app has a title. You can use specific text functions to add content to your app, or you can use `st.write()` and add your own markdown.

Write a data frame

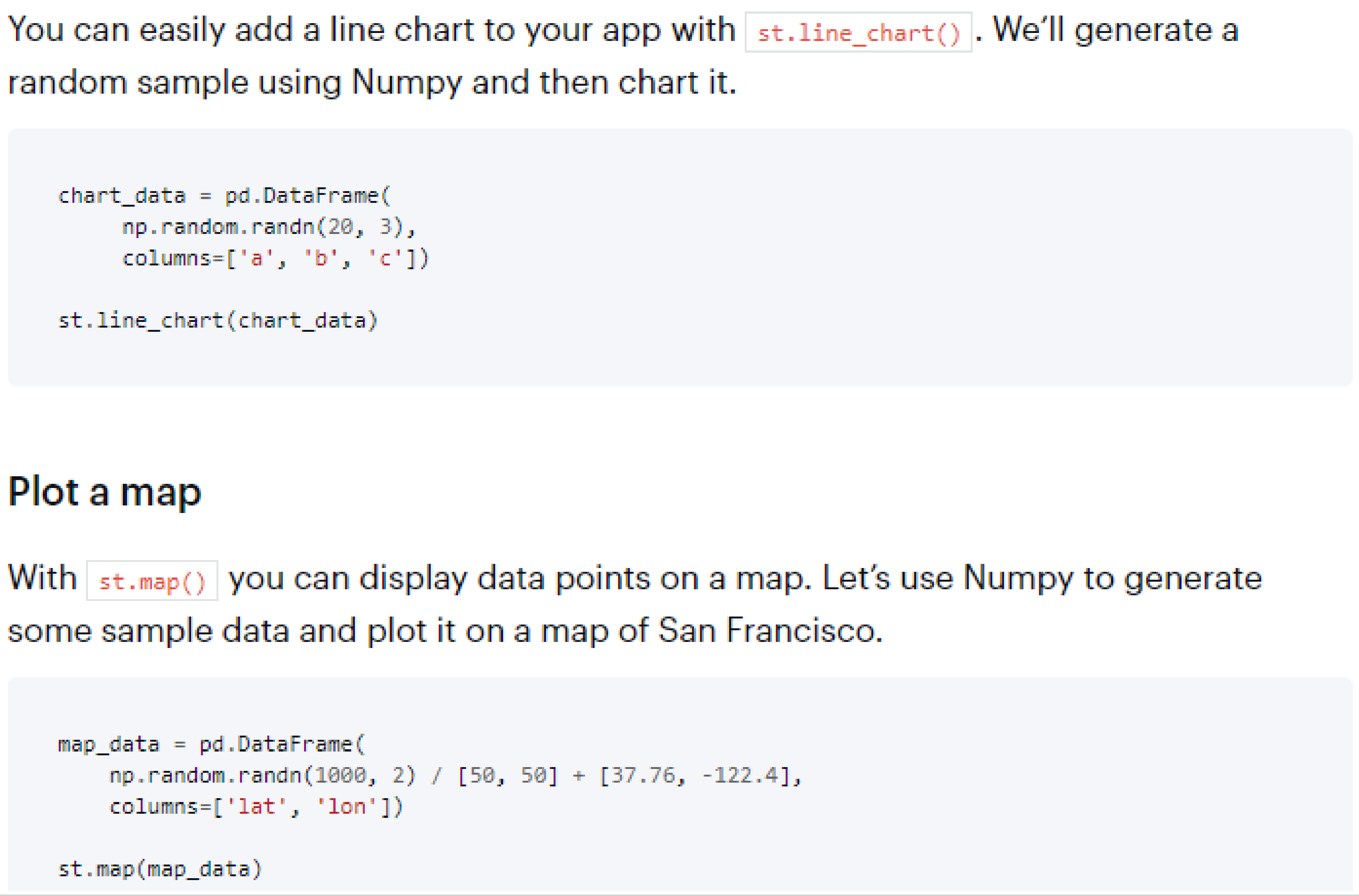


Along with magic commands, `st.write()` is Streamlit's "Swiss Army knife". You can pass almost anything to `st.write()`: text, data, Matplotlib figures, Altair charts, and more. Don't worry, Streamlit will figure it out and render things the right way.

```
st.write("Here's our first attempt at using data to create a table:")
st.write(pd.DataFrame({
    'first column': [1, 2, 3, 4],
    'second column': [10, 20, 30, 40]
}))
```

There are other data specific functions like `st.dataframe()` and `st.table()` that you can also use for displaying data. Check our advanced guides on displaying data to understand when to use these features and how to add colors and styling to your data frames.

Draw charts and maps



You can easily add a line chart to your app with `st.line_chart()`. We'll generate a random sample using Numpy and then chart it.

```
chart_data = pd.DataFrame(
     np.random.randn(20, 3),
     columns=['a', 'b', 'c'])

st.line_chart(chart_data)
```

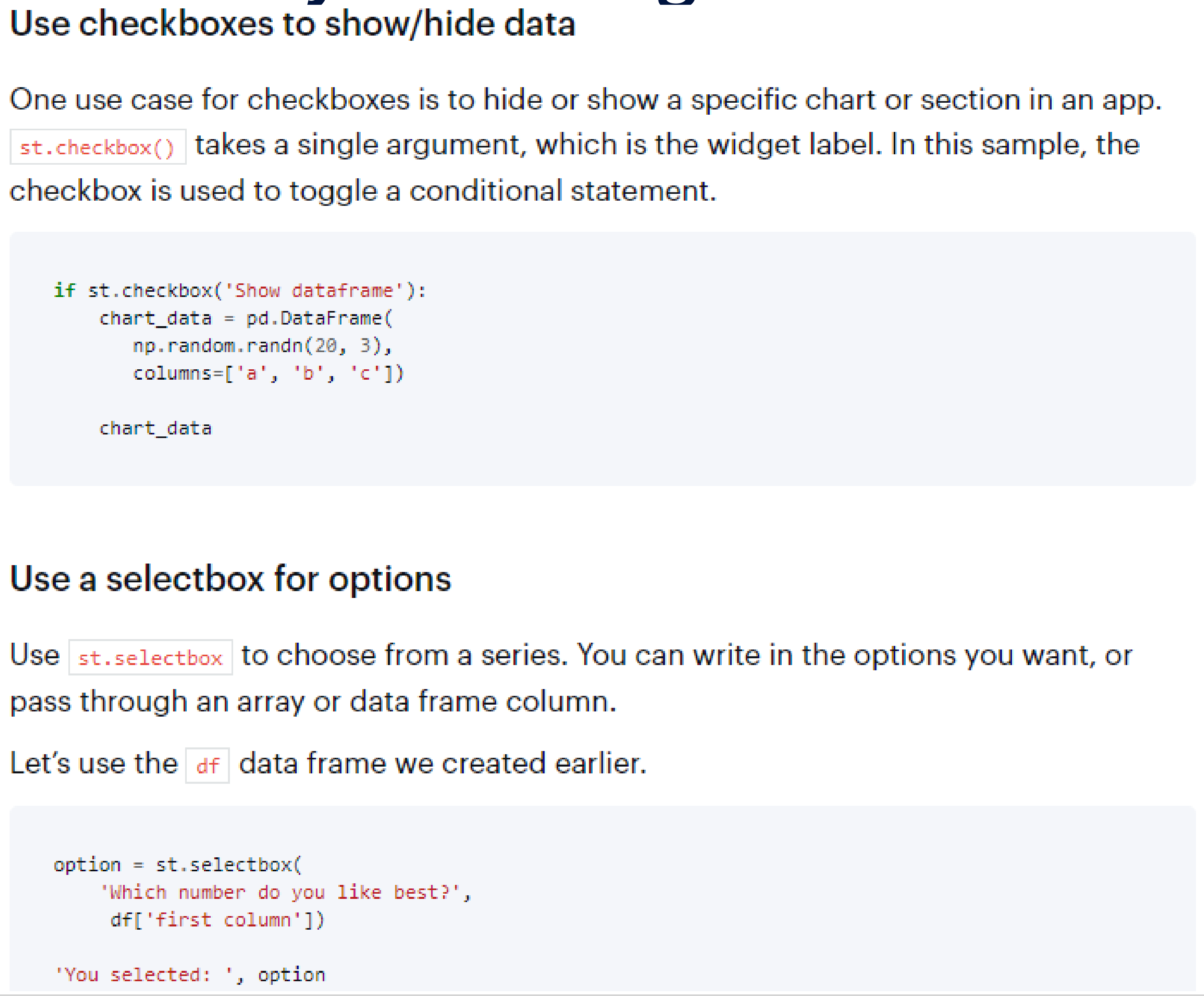
Plot a map

With `st.map()` you can display data points on a map. Let's use Numpy to generate some sample data and plot it on a map of San Francisco.

```
map_data = pd.DataFrame(
    np.random.randn(1000, 2) / [50, 50] + [37.76, -122.4],
    columns=['lat', 'lon'])

st.map(map_data)
```

Add interactivity with widgets



Use checkboxes to show/hide data

One use case for checkboxes is to hide or show a specific chart or section in an app. `st.checkbox()` takes a single argument, which is the widget label. In this sample, the checkbox is used to toggle a conditional statement.

```
if st.checkbox('Show dataframe'):
    chart_data = pd.DataFrame(
       np.random.randn(20, 3),
       columns=['a', 'b', 'c'])

    chart_data
```

Use a selectbox for options

Use `st.selectbox` to choose from a series. You can write in the options you want, or pass through an array or data frame column.

Let’s use the `df` data frame we created earlier.

```
option = st.selectbox(
    'Which number do you like best?',
     df['first column'])

'You selected: ', option
```

Lay out your app (Sidebar)

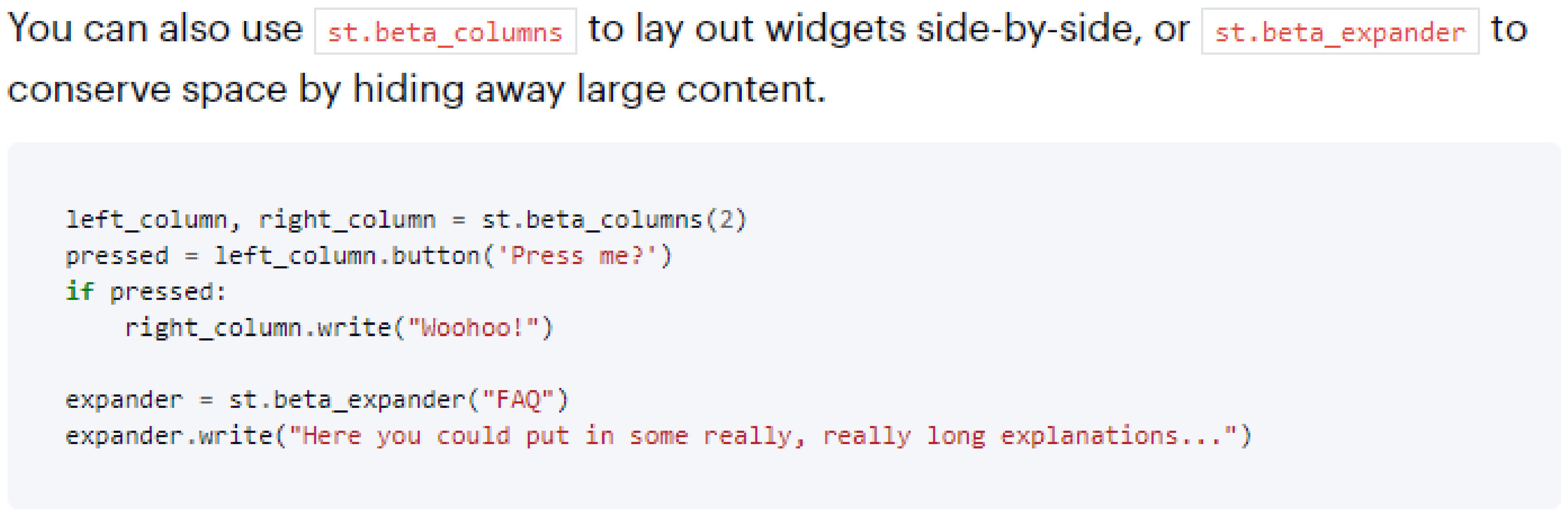
For a cleaner look, you can move your widgets into a sidebar. This keeps your app central, while widgets are pinned to the left. Let's take a look at how you can use `st.sidebar` in your app.

```
option = st.sidebar.selectbox(
    'Which number do you like best?',
     df['first column'])

'You selected:', option
```

Most of the elements you can put into your app can also be put into a sidebar using this syntax: `st.sidebar.[element_name]()`. Here are a few examples that show how it's used: `st.sidebar.markdown()`, `st.sidebar.slider()`, `st.sidebar.line_chart()`.

Lay out your app (widgets side-by-side)

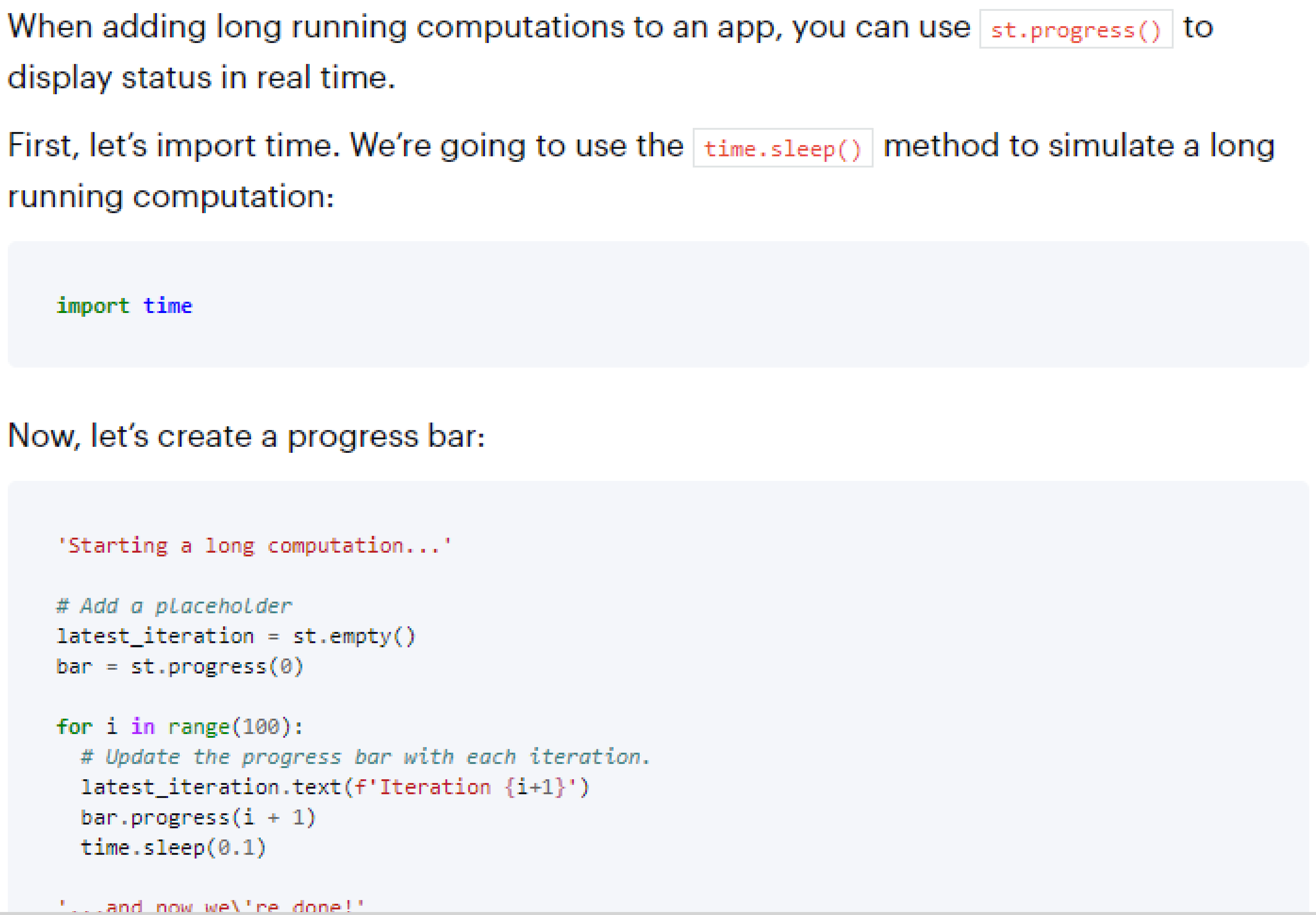


You can also use `st.beta_columns` to lay out widgets side-by-side, or `st.beta_expander` to conserve space by hiding away large content.

```python
left_column, right_column = st.beta_columns(2)
pressed = left_column.button('Press me?')
if pressed:
    right_column.write("Woohoo!")

expander = st.beta_expander("FAQ")
expander.write("Here you could put in some really, really long explanations...")
```

Show progress



When adding long running computations to an app, you can use `st.progress()` to display status in real time.

First, let's import time. We're going to use the `time.sleep()` method to simulate a long running computation:

```
import time
```

Now, let's create a progress bar:

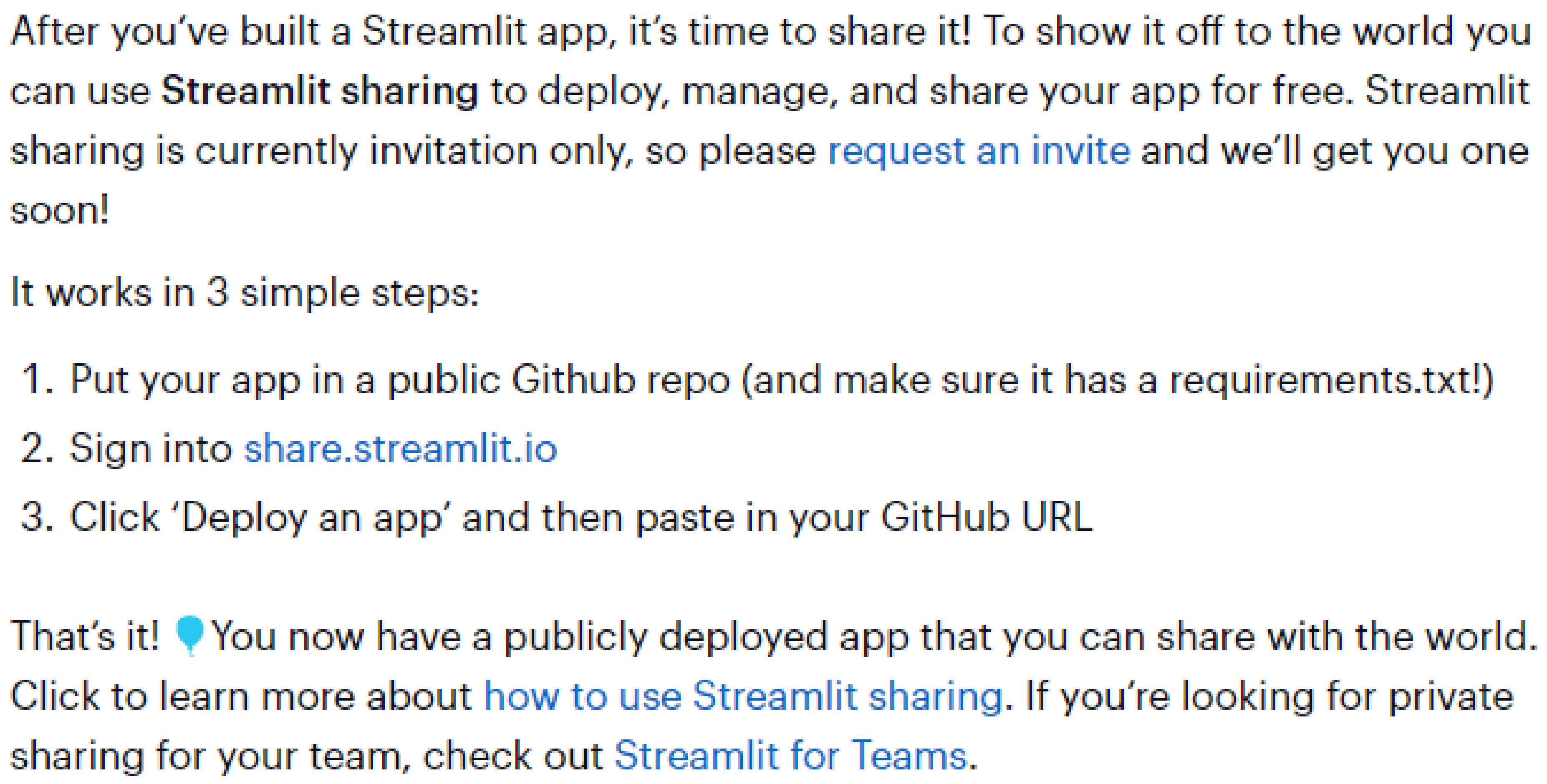
```
'Starting a long computation...'

# Add a placeholder
latest_iteration = st.empty()
bar = st.progress(0)

for i in range(100):
  # Update the progress bar with each iteration.
  latest_iteration.text(f'Iteration {i+1}')
  bar.progress(i + 1)
  time.sleep(0.1)

'...and now we\'re done!'
```

Share your app



After you've built a Streamlit app, it's time to share it! To show it off to the world you can use **Streamlit sharing** to deploy, manage, and share your app for free. Streamlit sharing is currently invitation only, so please request an invite and we'll get you one soon!

It works in 3 simple steps:

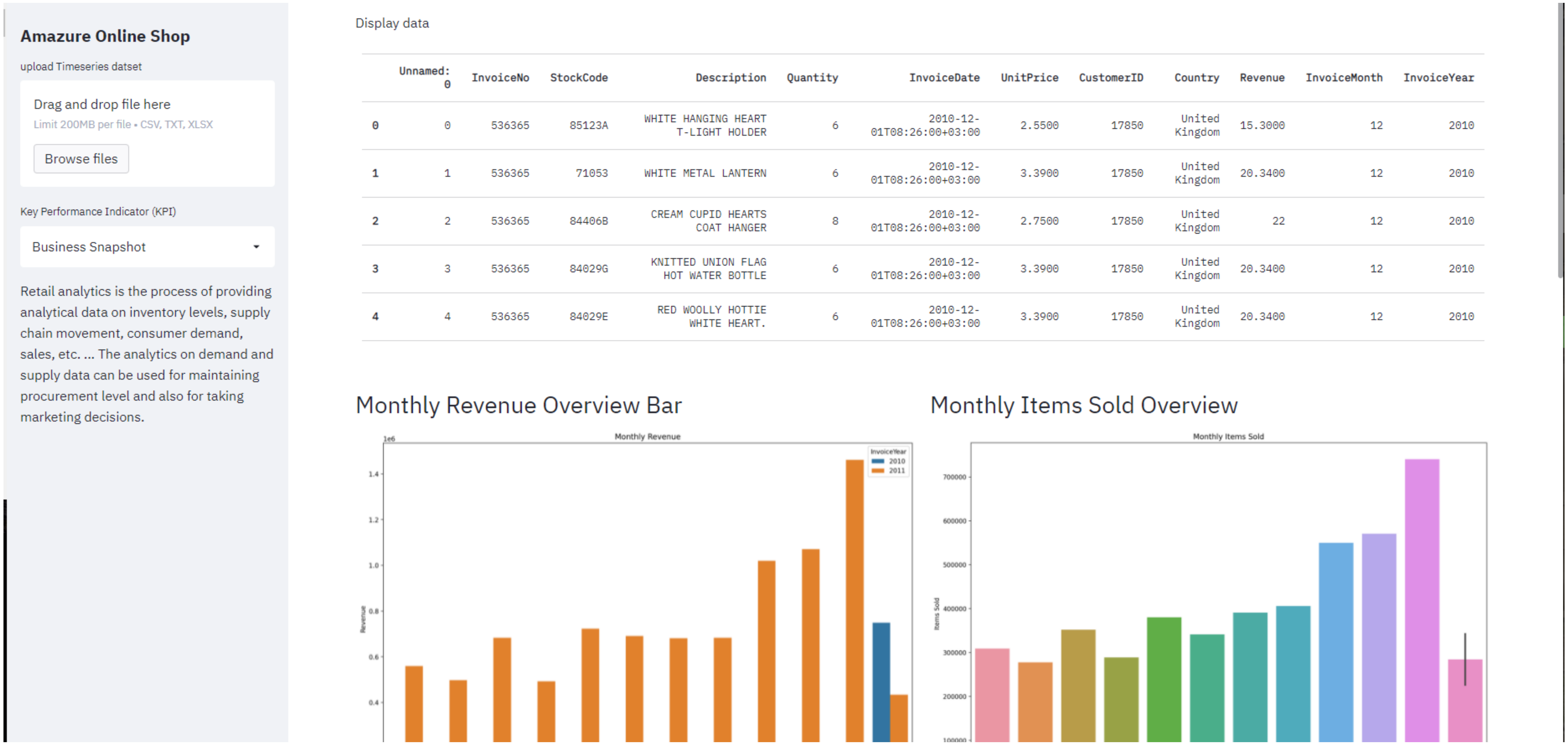
1. Put your app in a public Github repo (and make sure it has a requirements.txt!)
2. Sign into share.streamlit.io
3. Click 'Deploy an app' and then paste in your GitHub URL

That's it! 🎈You now have a publicly deployed app that you can share with the world. Click to learn more about how to use Streamlit sharing. If you're looking for private sharing for your team, check out Streamlit for Teams.

Practical

Amazure Online Shop   Dashbaord

Amazure Online Shop Dashbaord



Amazure Online Shop

upload Timeseries datset

Drag and drop file here

Limit 200MB per file • CSV, TXT, XLSX

Browse files

Key Performance Indicator (KPI)

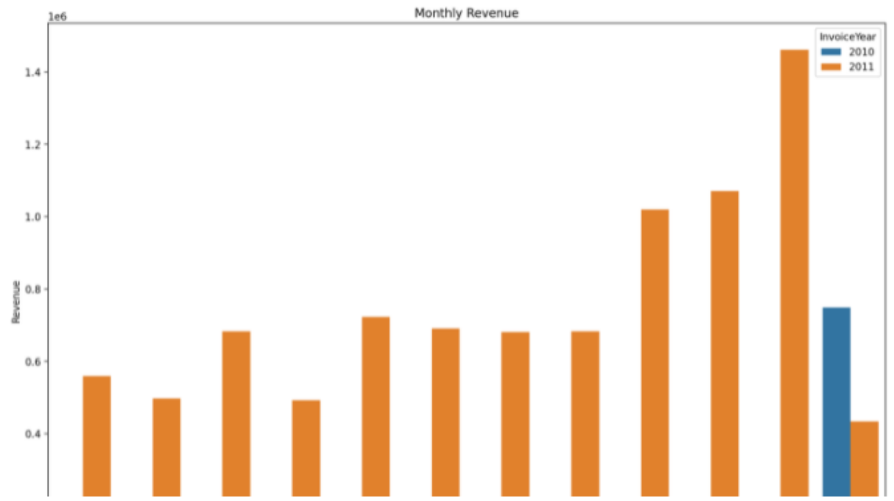
Business Snapshot

Retail analytics is the process of providing analytical data on inventory levels, supply chain movement, consumer demand, sales, etc. ... The analytics on demand and supply data can be used for maintaining procurement level and also for taking marketing decisions.

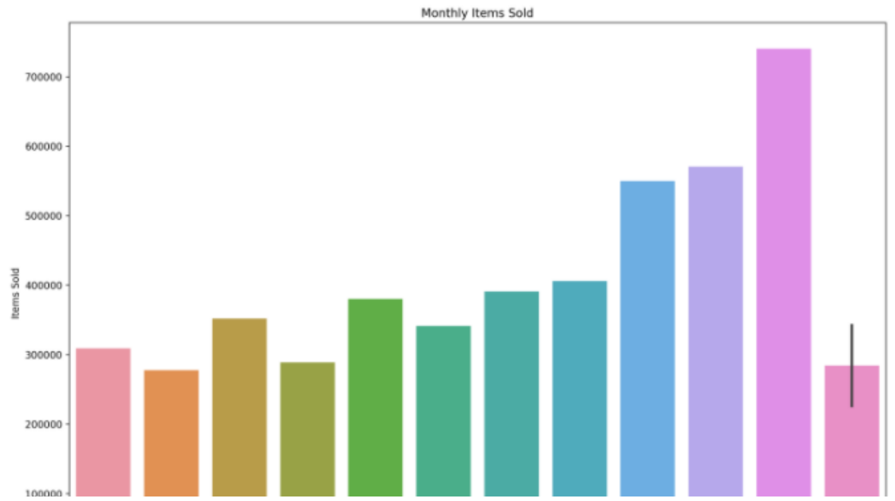
Display data

	Unnamed: 0	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	Revenue	InvoiceMonth	InvoiceYear
0	0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01T08:26:00+03:00	2.5500	17850	United Kingdom	15.3000	12	2010
1	1	536365	71053	WHITE METAL LANTERN	6	2010-12-01T08:26:00+03:00	3.3900	17850	United Kingdom	20.3400	12	2010
2	2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01T08:26:00+03:00	2.7500	17850	United Kingdom	22	12	2010
3	3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01T08:26:00+03:00	3.3900	17850	United Kingdom	20.3400	12	2010
4	4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01T08:26:00+03:00	3.3900	17850	United Kingdom	20.3400	12	2010

Monthly Revenue Overview Bar



Monthly Items Sold Overview


Monthly Items Sold
Items Sold