

Nano Movie Recommender System



CLASS 3 - PART 3

HOW TO SET UP THE DEVELOPMENT
ENVIRONMENT (STREAMLIT)

Yaoyao(Freax) Qian
h-freax.github.io

What is Streamlit?

<https://streamlit.io>



Playground

Gallery

Components

Cloud

Community

Docs ↗

Deploying? Try:

Free

Pro

A faster way to build and share data apps

Turn your data scripts into shareable web apps in minutes.

All in pure Python. No front-end experience required.

Get start



Try the live playground!

Playground

Try out a limited version of Streamlit right in your browser.

Just edit the code below and the app on the right updates automatically.

For the real thing, [install our Python library](#).

EXAMPLES Blank **Hello** Charts Dataframes LLM chat Computer vision Geospatial

```
import streamlit as st
st.title("Hello Streamlit-er👋")
st.markdown(
    """
    This is a playground for you to try Streamlit and have fun.

    **There's :rainbow[so much] you can build!**

    We prepared a few examples for you to get started. Just
    click on the buttons above and discover what you can do
    with Streamlit.
    """
)
if st.button("Send balloons!"):
    st.balloons()
```

Hello Streamlit-er👋

This is a playground for you to try Streamlit and have fun.

There's **so much** you can build!

We prepared a few examples for you to get started. Just click on the buttons
above and discover what you can do with Streamlit.

Send balloons!

EDIT HERE APP



```
import streamlit as st  
  
st.title("This is title")
```

Share

This is title

This line of code tells Python that you want to use the streamlit tool.

Analogy:

You download an app on your phone (streamlit is installed).

But before using it, you need to open the app.

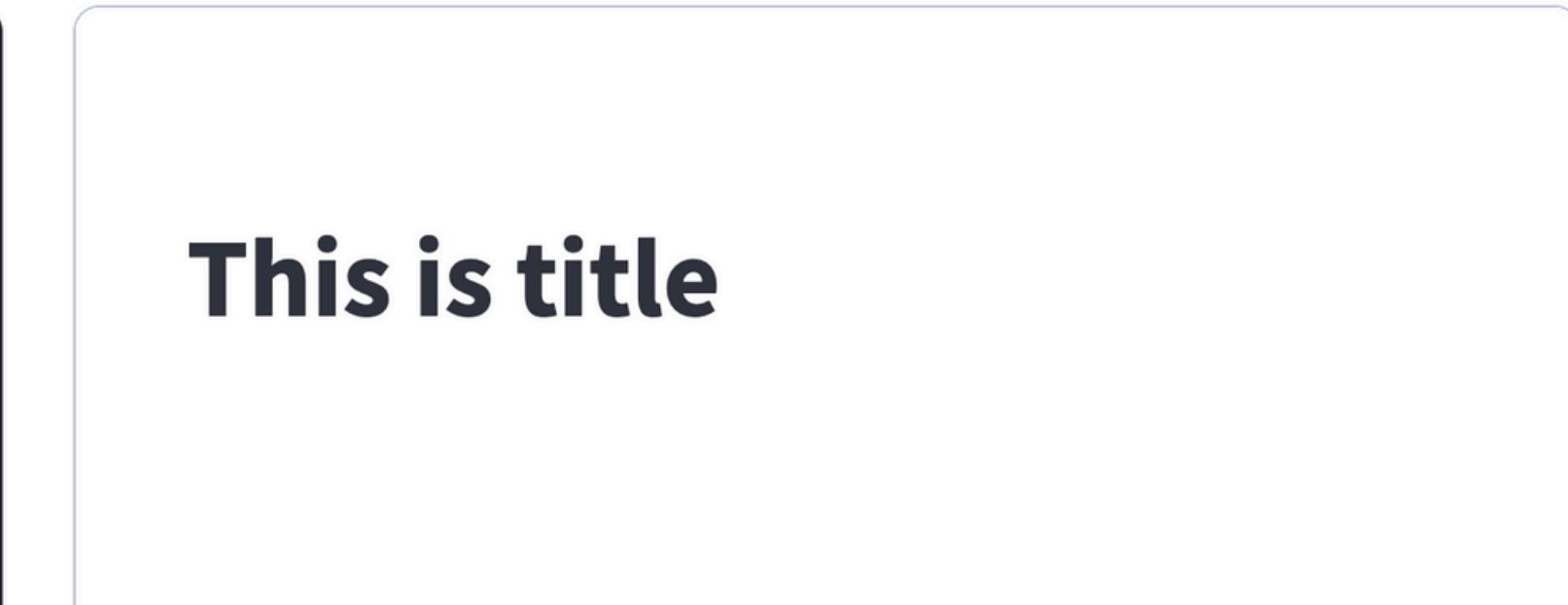
`import streamlit as st` is like opening the Streamlit app in Python.



```
import streamlit as st

st.title("This is title")
```

Share



What is **import**?

- ◆ import is a command in Python that allows you to use a tool (library).

✓ Analogy:

You have a rice cooker at home.

But if you don't plug it in, it won't work.

import is like "plugging in" the rice cooker, so Python knows you want to use streamlit.

📌 In Python, many useful tools (like streamlit) must be imported before you can use them.



```
import streamlit as st  
  
st.title("This is title")
```

Share

This is title

What does `as st` do?

- ◆ `as st` is a shortcut to rename `streamlit` as `st`, making it easier to use.

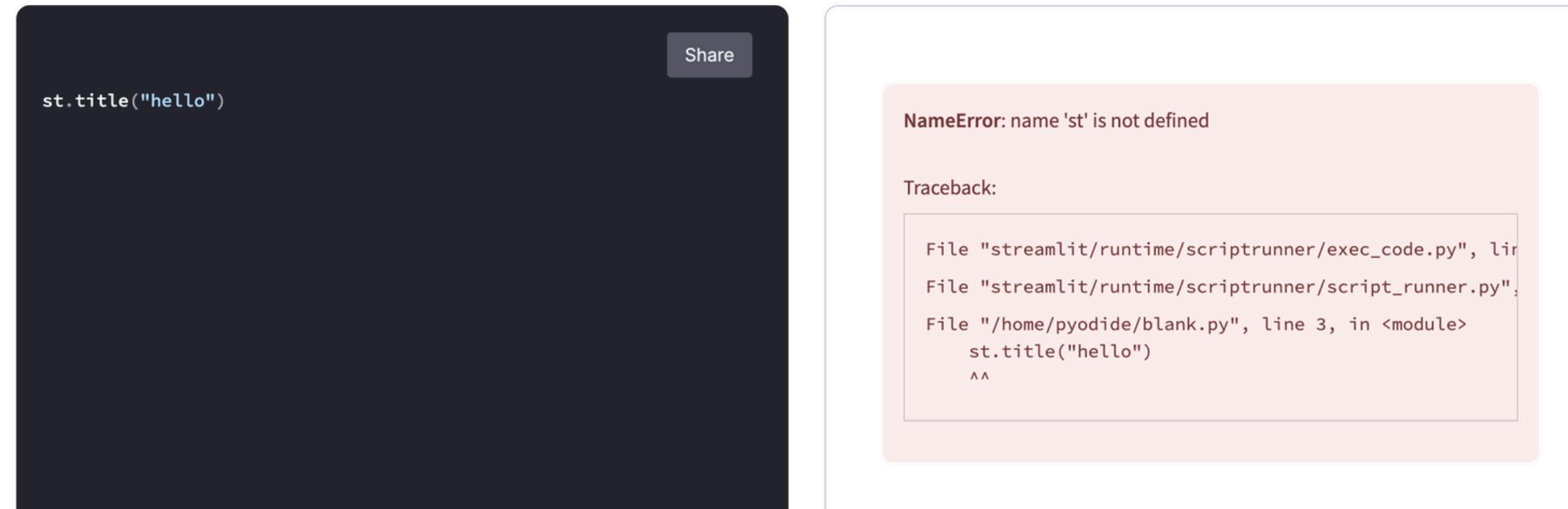
✓ **Analogy:**

Imagine your friend's full name is "Alexander Smith".

You don't want to say the full name every time, so you call him "Alex".

`as st` means you can now type `st` instead of `streamlit`, making your code shorter.

What happens if you don't write import streamlit as st?



A screenshot of a Streamlit application window. On the left, a dark panel contains the Python code: `st.title("hello")`. On the right, a light-colored panel shows the resulting error message and traceback:

```
NameError: name 'st' is not defined

Traceback:
File "streamlit/runtime/scriptrunner/exec_code.py", line ...
File "streamlit/runtime/scriptrunner/script_runner.py", line ...
File "/home/pyodide/blank.py", line 3, in <module>
    st.title("hello")
    ^^
```

 This is like trying to cook without plugging in the rice cooker—it won't work.

The screenshot shows the Streamlit playground interface. On the left, a dark sidebar displays the Python code for the 'Hello' example:

```
import streamlit as st
st.title("Hello Streamlit-er 🙌")
st.markdown(
"""
    This is a playground for you to try Streamlit and have fun.

    **There's :rainbow[so much] you can build!**

    We prepared a few examples for you to get started. Just
    click on the buttons above and discover what you can do
    with Streamlit.
"""
)
if st.button("Send balloons!"):
    st.balloons()
```

A yellow hand icon points to the first line of code, `st.title("Hello Streamlit-er 🙌")`. In the top right corner of the sidebar, there is a 'Share' button.

On the right, the generated Streamlit app is shown. It features a large title "Hello Streamlit-er" with a waving hand emoji. Below the title, the explanatory text and code output are displayed. A 'Send balloons!' button is visible at the bottom.

`st.title("Hello Streamlit-er 🙌")`

👉 This creates a big title at the top of the webpage.

✓ Analogy:

Think of a webpage like a book or article.

The title of a book (like "Harry Potter") is big and stands out.

`st.title()` makes a big, bold title at the top of your Streamlit app.

The screenshot shows the Streamlit playground interface. On the left, there's a dark sidebar containing the Python code for the 'Hello' example. A yellow hand icon points to the first line of code, `import streamlit as st`. The main area displays the rendered Streamlit app. It features a title "Hello Streamlit-er 🙌", a "Share" button, and a "Send balloons!" button. The text in the app includes "This is a playground for you to try Streamlit and have fun.", "There's :rainbow[so much] you can build!", and "We prepared a few examples for you to get started. Just click on the buttons above and discover what you can do with Streamlit.".

st.markdown(...)

👉 This adds some text to the webpage, but with rich formatting (bold, colors, and more).

✓ Analogy:

Imagine you are writing a blog post.

Some words are bold, some are italic, and some are 🎨 colorful.

st.markdown() lets you write text with formatting, just like in a blog or document.

The image shows a Streamlit playground interface. On the left, a dark-themed code editor displays the following Python code:

```
import streamlit as st

st.title("Hello Streamlit-er 🙌")
st.markdown(
    """
        This is a playground for you to try Streamlit and have fun.

        **There's :rainbow[so much] you can build!**

        We prepared a few examples for you to get started. Just
        click on the buttons above and discover what you can do
        with Streamlit.
    """
)

if st.button("Send balloons!"):
    st.balloons()
```

A yellow hand icon points to the line `if st.button("Send balloons!"):`. In the top right corner of the code editor, there is a "Share" button.

On the right, the corresponding Streamlit app is running. It features a large title "Hello Streamlit-er" with a waving hand emoji. Below the title, the text "This is a playground for you to try Streamlit and have fun." is displayed. A bolded section "There's so much you can build!" follows. At the bottom, a button labeled "Send balloons!" is visible. Above the button, the text "We prepared a few examples for you to get started. Just click on the buttons above and discover what you can do with Streamlit." is shown.

`if st.button("Send balloons!"):`

👉 This creates a button that a user can click.

✓ Analogy:

Imagine an elevator button. When you press it, something happens (the elevator moves).

This button works the same way: when clicked, it does something special.

Hello Streamlit-er 🙌

This is a playground for you to try Streamlit and have fun.

There's **so much** you can build!

We prepared a few examples for you to get started. Just click on the buttons above and discover what you can do with Streamlit.

Send balloons!



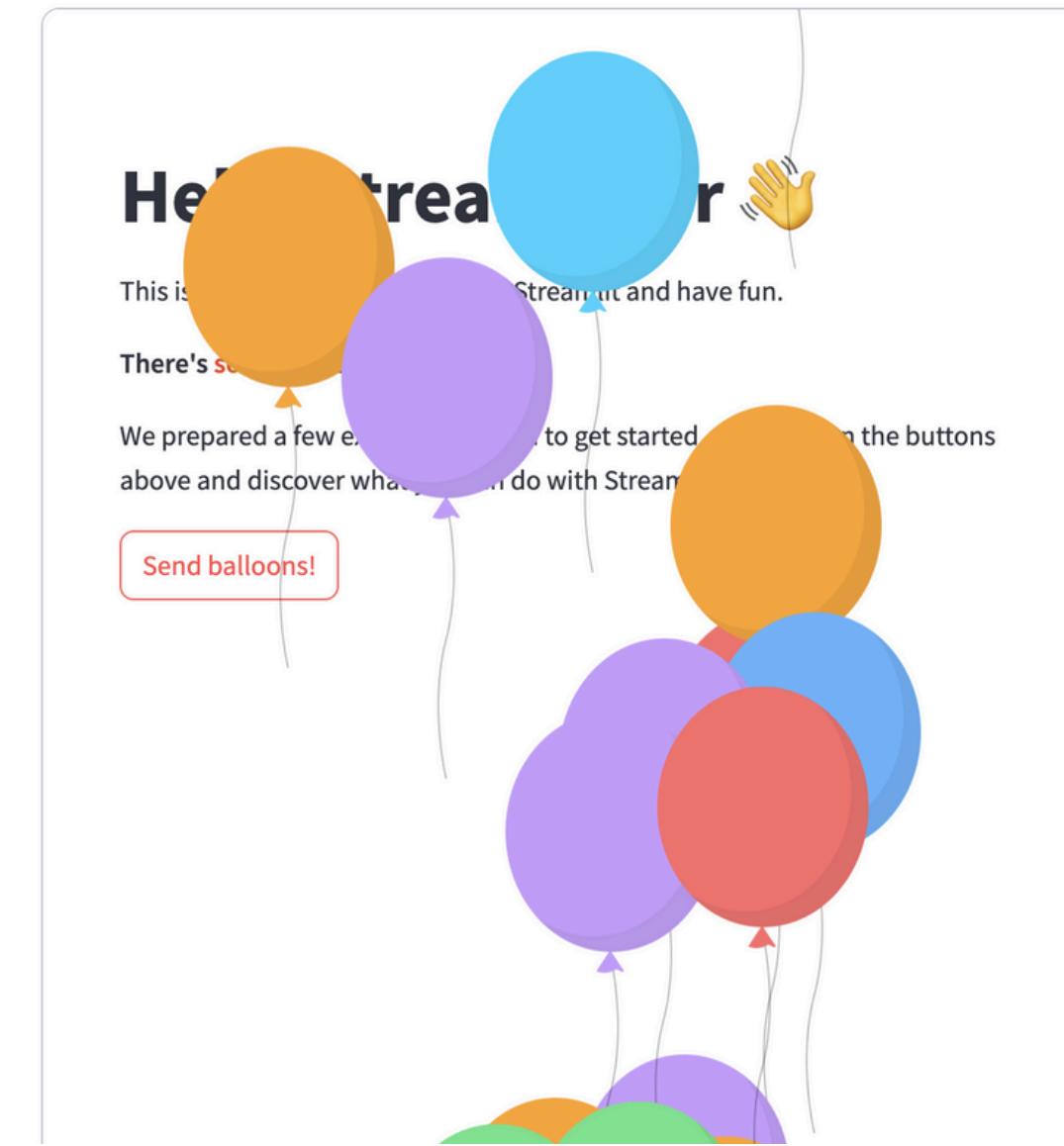
Hello Streamlit-er 🙌

This is a playground for you to try Streamlit and have fun.

There's **so much** you can build!

We prepared a few examples for you to get started. Just click on the buttons above and discover what you can do with Streamlit.

Send balloons!



st.balloons()

👉 This makes animated balloons fly across the screen! 🎈🎈🎈

✓ **Analogy:**

Imagine clicking a button at a birthday party, and suddenly, balloons fill the room!
This command adds a fun animation when the button is pressed.

```
import streamlit as st
import pandas as pd
import numpy as np

st.write("Streamlit supports a wide range of data visualizations, i

all_users = ["Alice", "Bob", "Charly"]
with st.container(border=True):
    users = st.multiselect("Users", all_users, default=all_users)
    rolling_average = st.toggle("Rolling average")

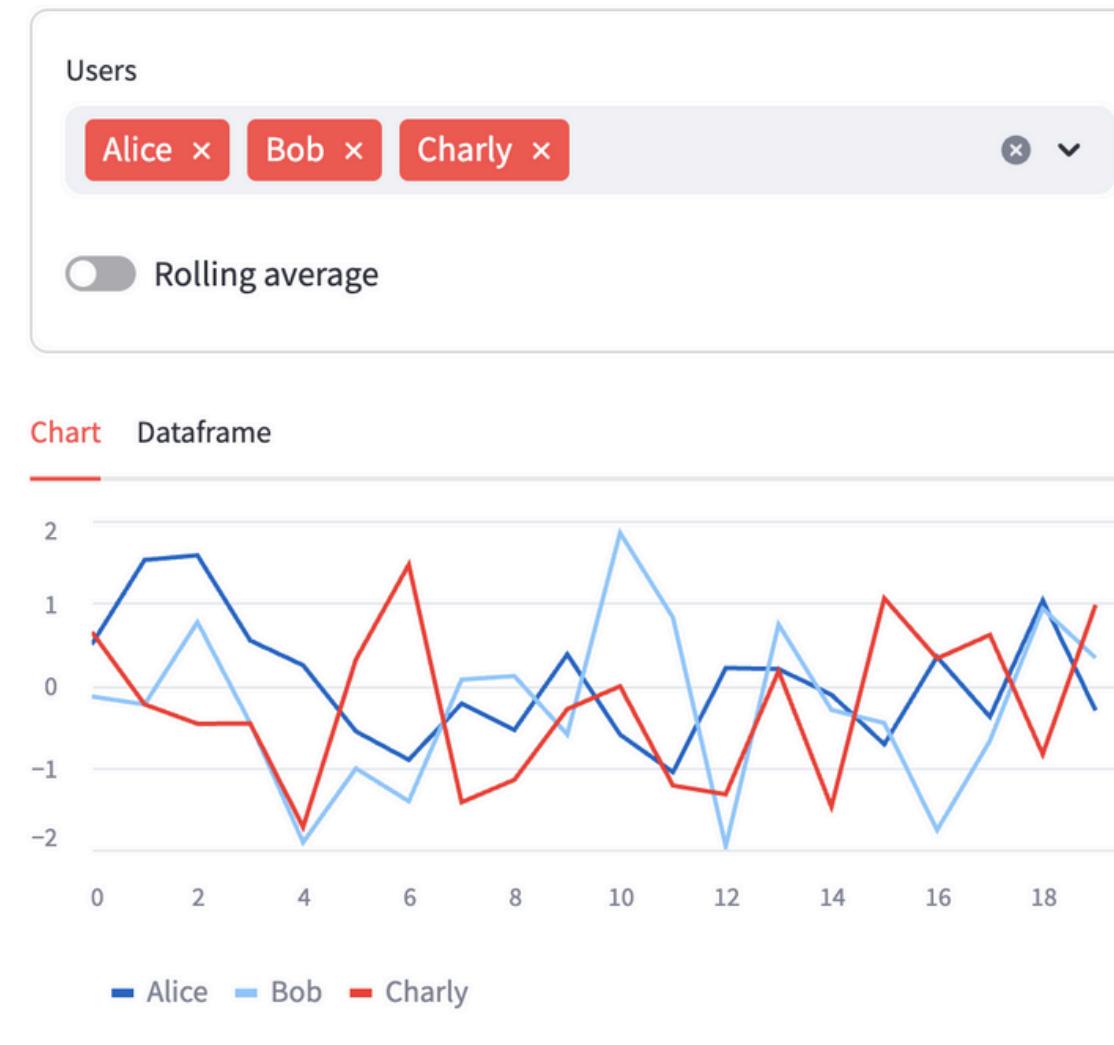
np.random.seed(42)
data = pd.DataFrame(np.random.randn(20, len(users)), columns=users)
if rolling_average:
    data = data.rolling(7).mean().dropna()

tab1, tab2 = st.tabs(["Chart", "Dataframe"])
tab1.line_chart(data, height=250)
tab2.dataframe(data, height=250, use_container_width=True)
```



Share

Streamlit supports a wide range of data visualizations, including [Plotly](#), [Altair](#), and [Bokeh charts](#).  And with over 20 input widgets, you can easily make your data interactive!



all_users = ["Alice", "Bob", "Charly"]

👉 Creates a list of users.

✓ Analogy:

Imagine you are listing your three best friends:
This list will be used later for selection.

```

import streamlit as st
import pandas as pd
import numpy as np

st.write("Streamlit supports a wide range of data visualizations, i
all_users = ["Alice", "Bob", "Charly"]
with st.container(border=True):
    users = st.multiselect("Users", all_users, default=all_users)
    rolling_average = st.toggle("Rolling average")

np.random.seed(42)
data = pd.DataFrame(np.random.randn(20, len(users)), columns=users)
if rolling_average:
    data = data.rolling(7).mean().dropna()

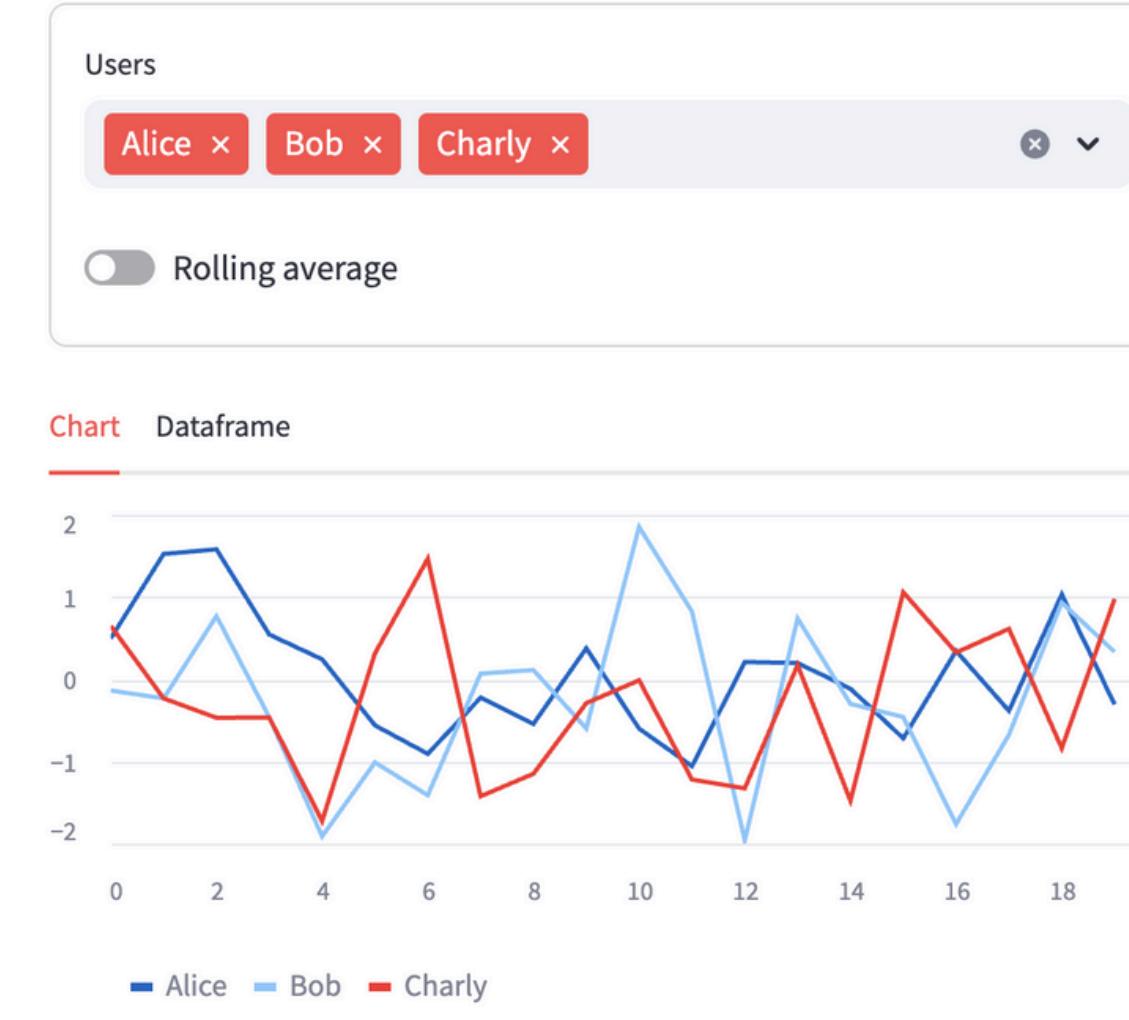
tab1, tab2 = st.tabs(["Chart", "Dataframe"])
tab1.line_chart(data, height=250)
tab2.dataframe(data, height=250, use_container_width=True)

```



Share

Streamlit supports a wide range of data visualizations, including [Plotly](#), [Altair](#), and [Bokeh charts](#). And with over 20 input widgets, you can easily make your data interactive!



with st.container(border=True):

👉 Creates a section with a border to keep UI elements grouped.

✓ Analogy:

This is like a box on a webpage that holds related content.

```
import streamlit as st
import pandas as pd
import numpy as np

st.write("Streamlit supports a wide range of data visualizations, i

all_users = ["Alice", "Bob", "Charly"]
with st.container(border=True):
    users = st.multiselect("Users", all_users, default=all_users)
    rolling_average = st.toggle("Rolling average")

np.random.seed(42)
data = pd.DataFrame(np.random.randn(20, len(users)), columns=users)
if rolling_average:
    data = data.rolling(7).mean().dropna()

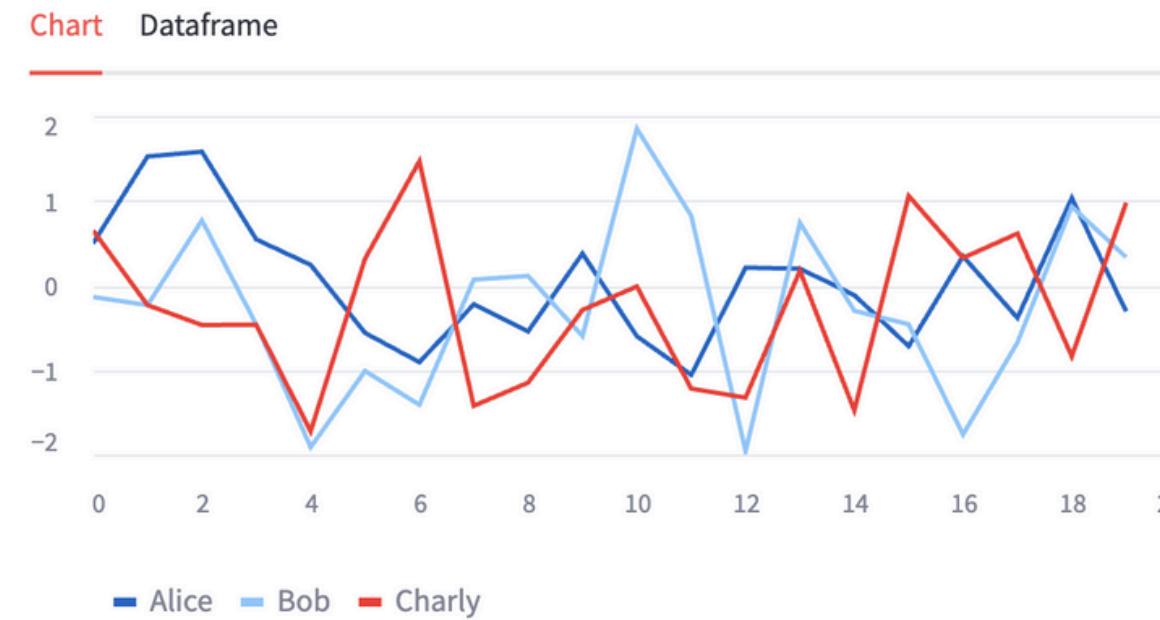
tab1, tab2 = st.tabs(["Chart", "Dataframe"])
tab1.line_chart(data, height=250)
tab2.dataframe(data, height=250, use_container_width=True)
```



Share

Streamlit supports a wide range of data visualizations, including [Plotly](#), [Altair](#), and [Bokeh charts](#). And with over 20 input widgets, you can easily make your data interactive!

A screenshot of a Streamlit application interface. At the top, there is a header labeled "Users". Below it is a multi-select dropdown menu containing three items: "Alice", "Bob", and "Charly", each with a red background and white text. To the right of the dropdown is a small "X" button and a downward arrow. Below the dropdown is a toggle switch labeled "Rolling average" with a grey background and a white circle.



`users = st.multiselect("Users", all_users, default=all_users)`

👉 Creates a multi-selection dropdown menu.

◆ What this does:

Shows a list: Alice, Bob, Charly.

Users can select one, two, or all names.

By default, all three names are selected.

```
import streamlit as st
import pandas as pd
import numpy as np

st.write("Streamlit supports a wide range of data visualizations, i

all_users = ["Alice", "Bob", "Charly"]
with st.container(border=True):
    users = st.multiselect("Users", all_users, default=all_users)
    rolling_average = st.toggle("Rolling average")

np.random.seed(42)
data = pd.DataFrame(np.random.randn(20, len(users)), columns=users)
if rolling_average:
    data = data.rolling(7).mean().dropna()

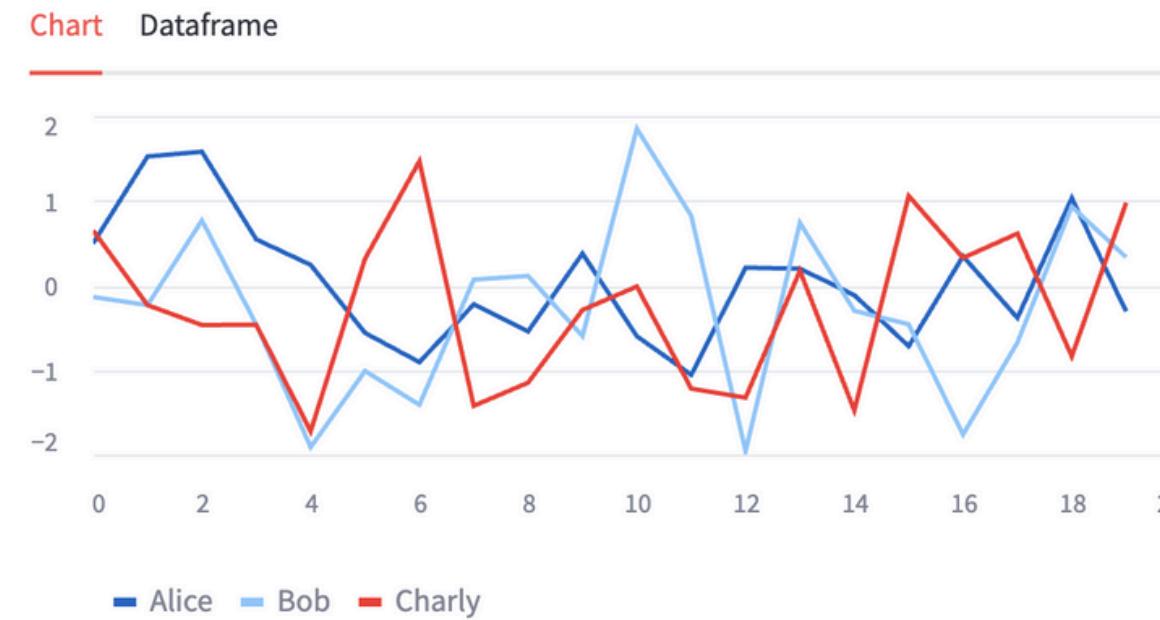
tab1, tab2 = st.tabs(["Chart", "Dataframe"])
tab1.line_chart(data, height=250)
tab2.dataframe(data, height=250, use_container_width=True)
```



Share

Streamlit supports a wide range of data visualizations, including [Plotly](#), [Altair](#), and [Bokeh charts](#).  And with over 20 input widgets, you can easily make your data interactive!

A screenshot of the Streamlit app interface. At the top, there's a header with the word 'Users'. Below it is a multiselect dropdown containing three items: 'Alice' (red), 'Bob' (blue), and 'Charly' (green). To the right of the dropdown is a small 'x' button and a dropdown arrow. Below the dropdown is a toggle switch labeled 'Rolling average' with a grey circle indicating it is off.



`rolling_average = st.toggle("Rolling average")`

👉 Creates an ON/OFF switch.

✓ Analogy:

Like a light switch  → Turn it ON or OFF.

Here, it controls whether we apply a rolling average to the data.

```

import streamlit as st
import pandas as pd
import numpy as np

st.write("Streamlit supports a wide range of data visualizations, i
all_users = ["Alice", "Bob", "Charly"]
with st.container(border=True):
    users = st.multiselect("Users", all_users, default=all_users)
    rolling_average = st.toggle("Rolling average")

np.random.seed(42)
data = pd.DataFrame(np.random.randn(20, len(users)), columns=users)
if rolling_average:
    data = data.rolling(7).mean().dropna()

tab1, tab2 = st.tabs(["Chart", "Dataframe"])
tab1.line_chart(data, height=250)
tab2.dataframe(data, height=250, use_container_width=True)

```



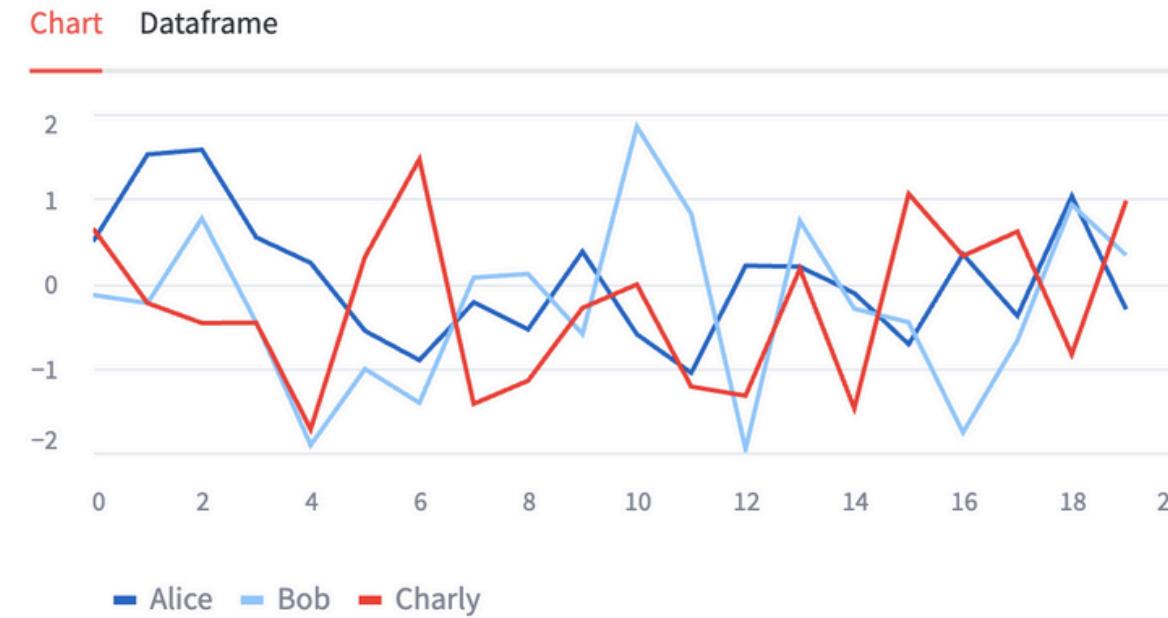
Share

Streamlit supports a wide range of data visualizations, including [Plotly](#), [Altair](#), and [Bokeh charts](#).  And with over 20 input widgets, you can easily make your data interactive!

Users

Alice ×
Bob ×
Charly ×
x ▾

Rolling average



np.random.seed(42)

👉 Makes sure the random numbers are always the same.

✓ Analogy:

Like fixing the shuffle order of a playlist 🎵 so every time you play it, the order stays the same.

```

import streamlit as st
import pandas as pd
import numpy as np

st.write("Streamlit supports a wide range of data visualizations, i")

all_users = ["Alice", "Bob", "Charly"]
with st.container(border=True):
    users = st.multiselect("Users", all_users, default=all_users)
    rolling_average = st.toggle("Rolling average")

np.random.seed(42)
data = pd.DataFrame(np.random.randn(20, len(users)), columns=users)
if rolling_average:
    data = data.rolling(7).mean().dropna()

tab1, tab2 = st.tabs(["Chart", "Dataframe"])
tab1.line_chart(data, height=250)
tab2.dataframe(data, height=250, use_container_width=True)

```



Share

Streamlit supports a wide range of data visualizations, including [Plotly](#), [Altair](#), and [Bokeh charts](#). And with over 20 input widgets, you can easily make your data interactive!

data = pd.DataFrame(np.random.randn(20, len(users)), columns=users)

👉 Creates a table with random numbers.

✓ Breaking it down:

`np.random.randn(20, len(users))` → Generates 20 rows of random numbers.

`columns=users` → Uses the selected users as column names.

🔊 Example Output (random numbers):

	Alice	Bob	Charly
0	0.49	-0.13	1.58
1	-0.86	0.76	0.12
2	0.46	0.34	-1.44
...

```

import streamlit as st
import pandas as pd
import numpy as np

st.write("Streamlit supports a wide range of data visualizations, i")

all_users = ["Alice", "Bob", "Charly"]
with st.container(border=True):
    users = st.multiselect("Users", all_users, default=all_users)
    rolling_average = st.toggle("Rolling average")

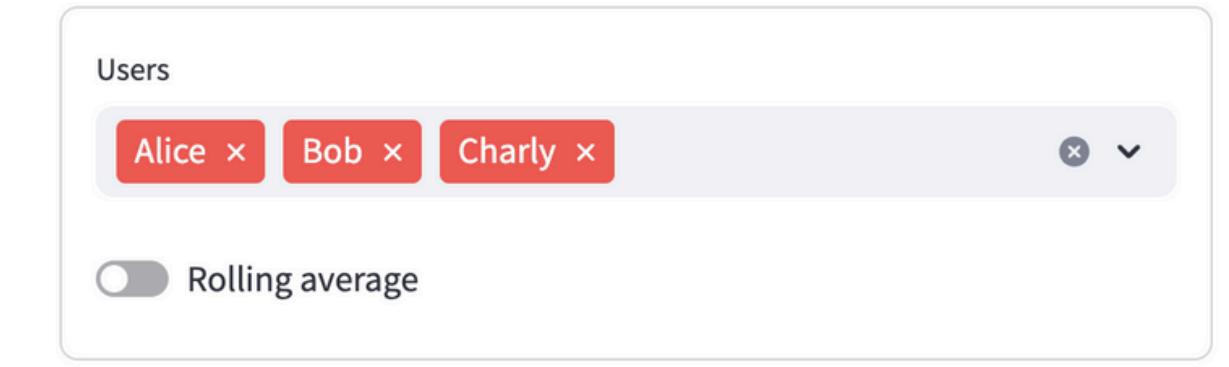
np.random.seed(42)
data = pd.DataFrame(np.random.randn(20, len(users)), columns=users)
if rolling_average:
    data = data.rolling(7).mean().dropna()

tab1, tab2 = st.tabs(["Chart", "Dataframe"])
tab1.line_chart(data, height=250)
tab2.dataframe(data, height=250, use_container_width=True)

```



Streamlit supports a wide range of data visualizations, including [Plotly](#), [Altair](#), and [Bokeh charts](#). And with over 20 input widgets, you can easily make your data interactive!



The screenshot shows a Streamlit interface with a sidebar titled 'Users' containing three selected items: Alice, Bob, and Charly. Below the sidebar is a toggle switch labeled 'Rolling average'. The main area displays two tabs: 'Chart' and 'Dataframe'. The 'Chart' tab is active, showing a line chart with three data series: Alice (blue), Bob (light blue), and Charly (red). The data points are highly volatile. The 'Dataframe' tab shows the same data in a tabular format.

if rolling_average:

- 👉 If the user turned ON the rolling average, apply it.
- ✓ What this does:

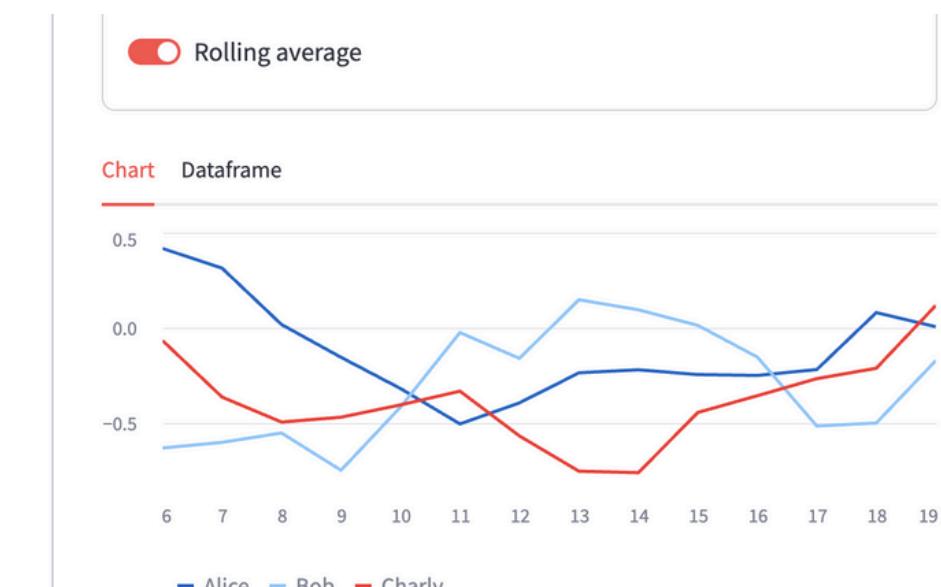
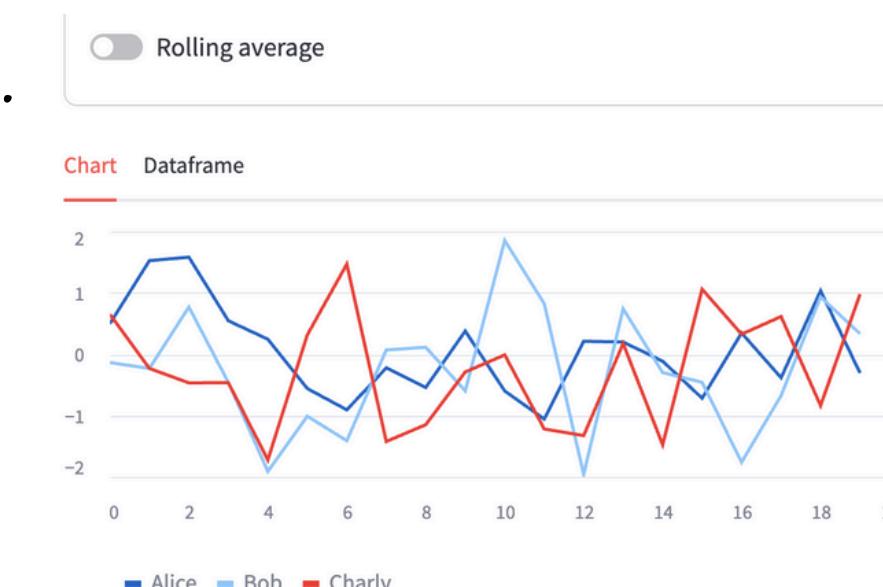
`data.rolling(7).mean()` → Averages the last 7 rows.

`.dropna()` → Removes empty values.

📢 Effect:

If the switch is ON, the data gets smoothed.

If the switch is OFF, the data stays random.



```

import streamlit as st
import pandas as pd
import numpy as np

st.write("Streamlit supports a wide range of data visualizations, i")

all_users = ["Alice", "Bob", "Charly"]
with st.container(border=True):
    users = st.multiselect("Users", all_users, default=all_users)
    rolling_average = st.toggle("Rolling average")

np.random.seed(42)
data = pd.DataFrame(np.random.randn(20, len(users)), columns=users)
if rolling_average:
    data = data.rolling(7).mean().dropna()

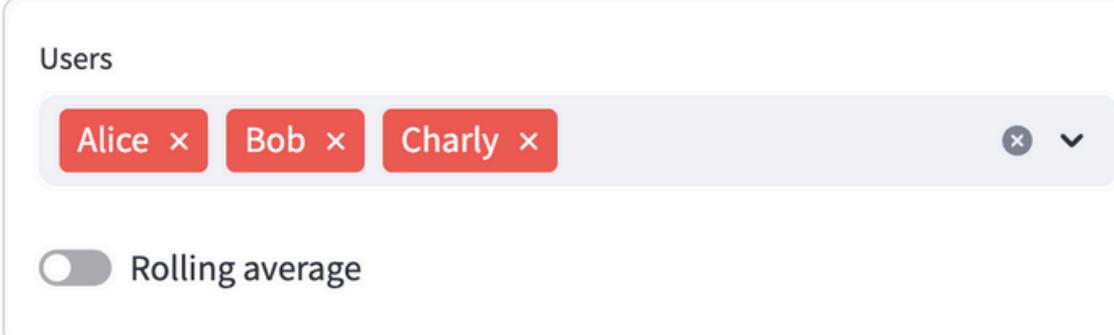
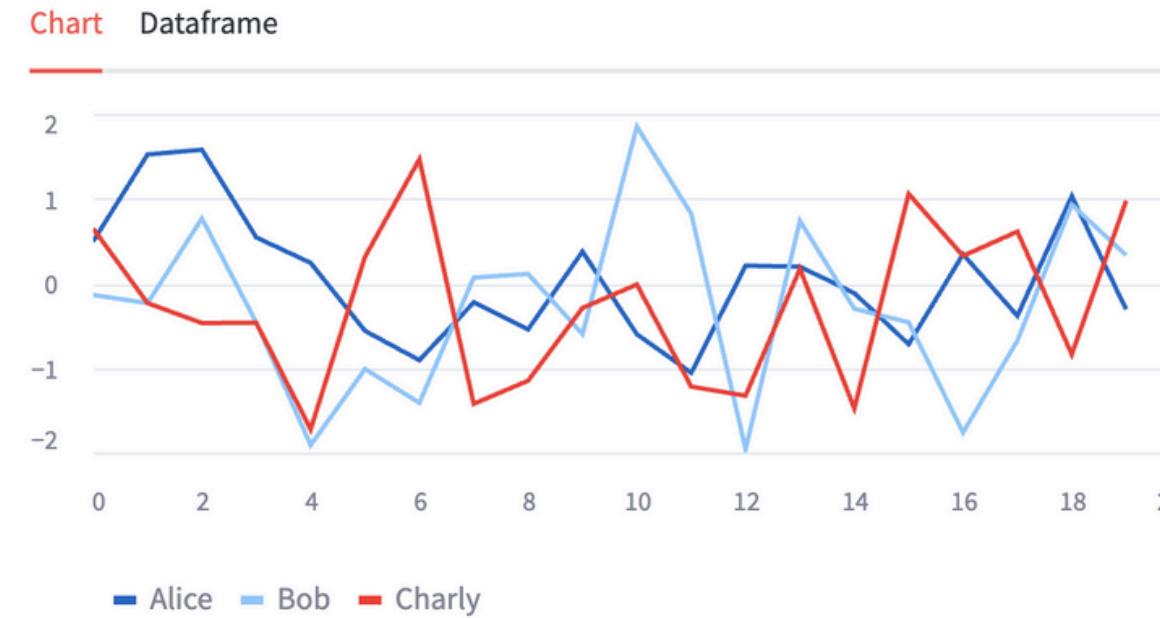
tab1, tab2 = st.tabs(["Chart", "Dataframe"])
tab1.line_chart(data, height=250)
tab2.dataframe(data, height=250, use_container_width=True)

```



Share

Streamlit supports a wide range of data visualizations, including [Plotly](#), [Altair](#), and [Bokeh charts](#).  And with over 20 input widgets, you can easily make your data interactive!

`tab1, tab2 = st.tabs(["Chart", "Dataframe"])`

👉 Creates two tabs:

1 “Chart” → Shows a graph.

2 “Dataframe” → Shows the raw table.

Chart Dataframe

	Alice	Bob	Charly
0	0.4967	-0.1383	0.647
1	1.523	-0.2342	-0.234
2	1.5792	0.7674	-0.469
3	0.5426	-0.4634	-0.465
4	0.242	-1.9133	-1.724
5	-0.5623	-1.0128	0.314

```
import streamlit as st
import pandas as pd
import numpy as np

st.write("Streamlit supports a wide range of data visualizations, i")

all_users = ["Alice", "Bob", "Charly"]
with st.container(border=True):
    users = st.multiselect("Users", all_users, default=all_users)
    rolling_average = st.toggle("Rolling average")

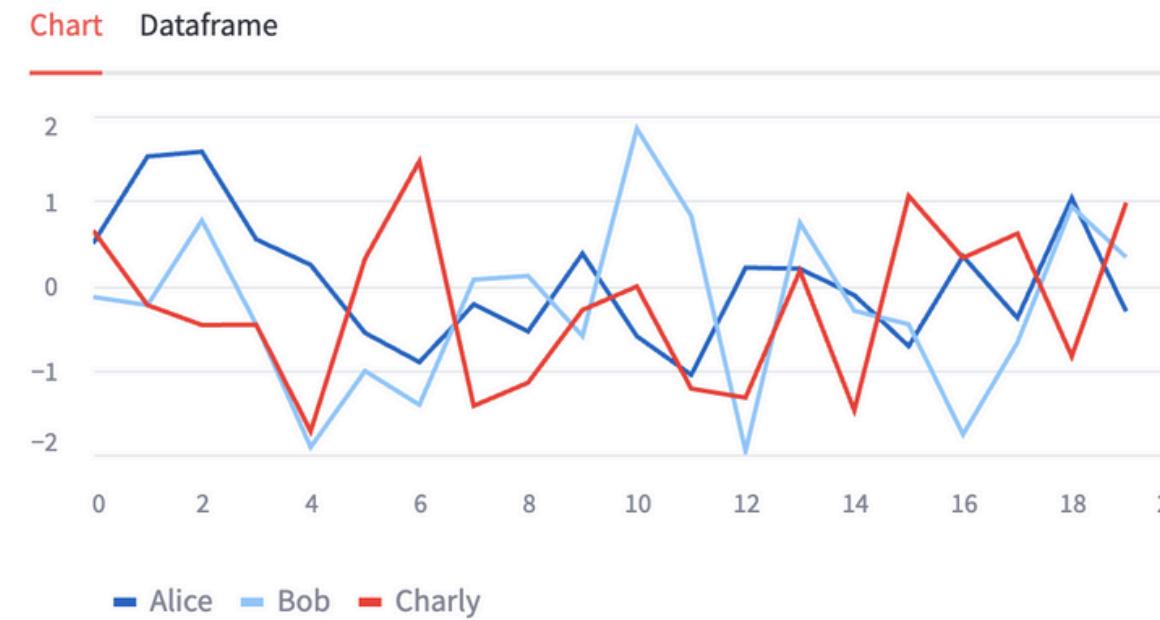
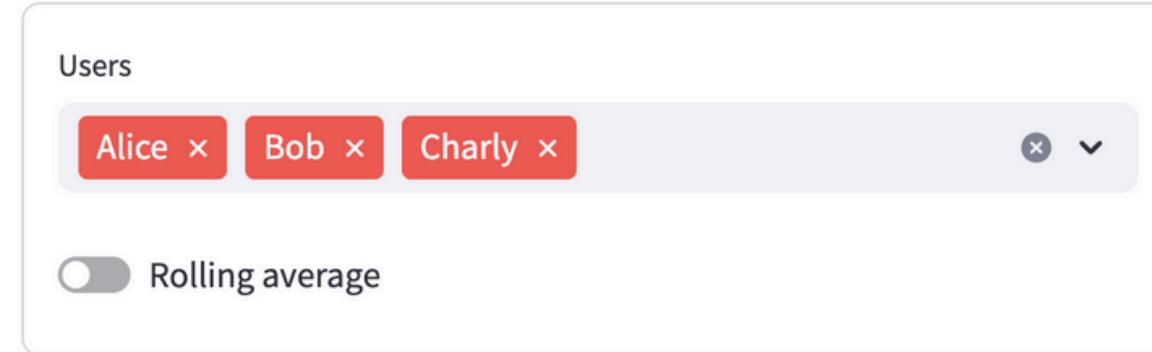
np.random.seed(42)
data = pd.DataFrame(np.random.randn(20, len(users)), columns=users)
if rolling_average:
    data = data.rolling(7).mean().dropna()

tab1, tab2 = st.tabs(["Chart", "Dataframe"])
tab1.line_chart(data, height=250)
tab2.dataframe(data, height=250, use_container_width=True)
```



Share

Streamlit supports a wide range of data visualizations, including [Plotly](#), [Altair](#), and [Bokeh charts](#). And with over 20 input widgets, you can easily make your data interactive!



tab1.line_chart(data, height=250)

👉 Displays a line chart of the data.

✓ Analogy:

Like an Excel chart 📈, but in Python.

```
import streamlit as st
import pandas as pd
import numpy as np

st.write("Streamlit supports a wide range of data visualizations, including charts, dataframes, and tables. You can even use machine learning models to make predictions!")

all_users = ["Alice", "Bob", "Charly"]
with st.container(border=True):
    users = st.multiselect("Users", all_users, default=all_users)
    rolling_average = st.toggle("Rolling average")

np.random.seed(42)
data = pd.DataFrame(np.random.randn(20, len(users)), columns=users)
if rolling_average:
    data = data.rolling(7).mean().dropna()

tab1, tab2 = st.tabs(["Chart", "Dataframe"])
tab1.line_chart(data, height=250)
tab2.dataframe(data, height=250, use_container_width=True)
```



Streamlit supports a wide range of data visualizations, including [Plotly](#), [Altair](#), and [Bokeh charts](#). And with over 20 input widgets, you can easily make your data interactive!

The screenshot shows a Streamlit application interface. On the left, there's a sidebar titled "Users" with three selected items: Alice, Bob, and Charly. Below it is a toggle switch labeled "Rolling average". The main area has two tabs at the top: "Chart" (which is currently active) and "Dataframe". The "Chart" tab displays a line chart of the data. The "Dataframe" tab displays a table of the same data. The table has columns for Alice, Bob, and Charly, and rows indexed from 0 to 5. The data values are as follows:

	Alice	Bob	Charly
0	0.4967	-0.1383	0.6477
1	1.523	-0.2342	-0.2341
2	1.5792	0.7674	-0.4695
3	0.5426	-0.4634	-0.4657
4	0.242	-1.9133	-1.7249
5	-0.5623	-1.0128	0.3142

`tab2.dataframe(data, height=250, use_container_width=True)`

👉 Displays the data table.

✓ Analogy:

Like an Excel spreadsheet but interactive.

Install Streamlit

There are multiple ways to set up your development environment and install Streamlit. Read below to understand these options. Developing locally with Python installed on your own computer is the most common scenario.

Summary for experts

1. Set up your Python development environment.
2. Run:

```
pip install streamlit
```

3. Validate the installation by running our Hello app:

```
streamlit hello
```

4. Jump to our [Basic concepts](#).

Step 1: Open Terminal

Windows:

Press Win + S, search for “Anaconda Prompt” or
“Command Prompt” (cmd).
Click Open.

Mac:

Press Cmd + Space, search for “Terminal”.
Click Open.

Step 2: Create a New Project Folder

Next, we will create a folder to store our Streamlit project.

📌 Run these commands in the terminal:

```
cd ~ # Go to your home directory  
mkdir nanomovie # Create a new folder  
cd nanomovie # Enter the folder
```



Now, you are inside your project folder!

```
03:10:38 with freax in ~ via ● v20.18.1 via ⬤ base  
[→ cd ~  
  
03:10:50 with freax in ~ via ● v20.18.1 via ⬤ base  
[→ mkdir nanomovie  
  
03:10:59 with freax in ~ via ● v20.18.1 via ⬤ base  
[→ cd nanomovie  
  
03:11:04 with freax in ~/nanomovie via ● v20.18.1 via ⬤ base  
[→
```

Step 3: Create & Activate a Conda Virtual Environment

It is recommended to use a Conda virtual environment to manage dependencies.

```
T 03:11:04 with freax in ~/nanomovie via ● v20.18.1 via Ⓜbase
[→ conda create -n nanomovie python=3.10 -y
```

conda create -n nanomovie → Creates a new environment called "nanomovie".
python=3.10 → Installs Python 3.10 in this environment.
-y → Automatically confirms the installation.

```
03:13:13 with freax in ~/nanomovie via ● v20.18.1 via Ⓜbase took 40.5s
[→ conda activate nanomovie

03:13:17 with freax in ~/nanomovie via ● v20.18.1 via Ⓜnanomovie ...
```

Activate the environment

Step 4: Install Streamlit

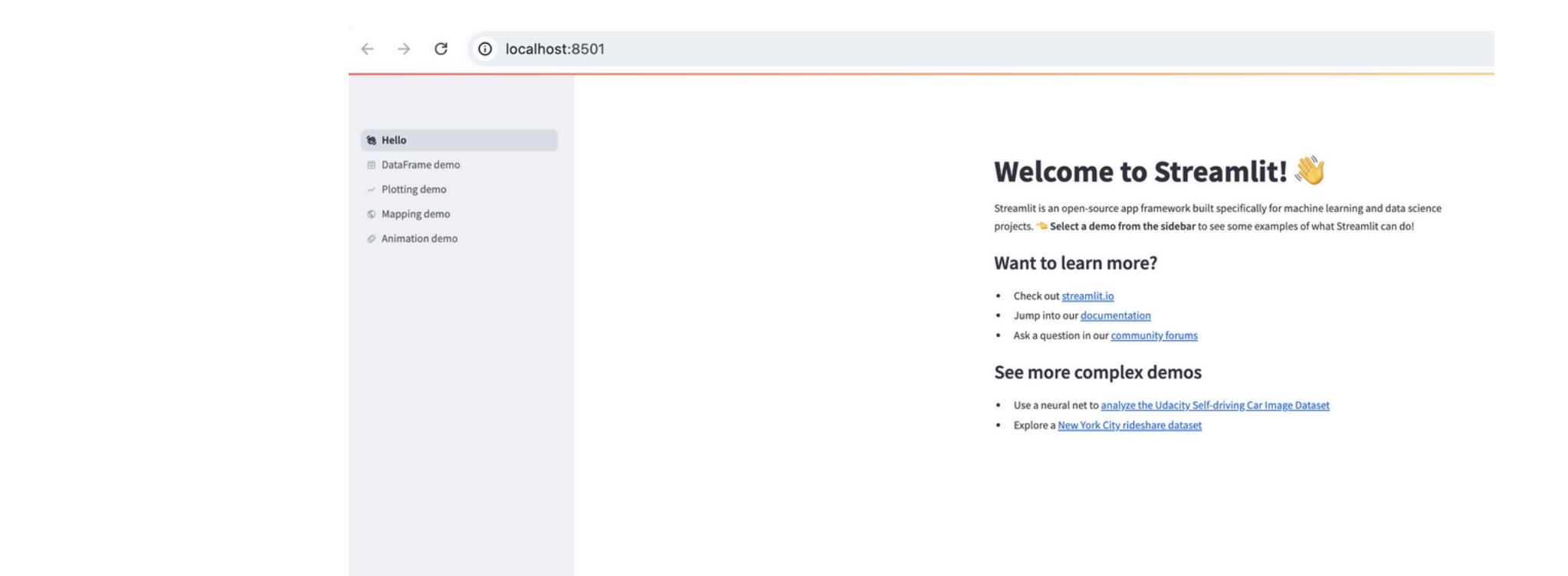
Now, let's install Streamlit inside this environment.

```
03:13:17 with freax in ~/nanomovie via ● v20.18.1 via Ⓜ nanomovie
→ pip install streamlit
```

Step 5:

Validate the installation by running Hello app

```
03:14:34 with freax in ~/nanomovie via ● v20.18.1 via Ⓜ nanomovie took 22.7s
→ streamlit hello
```



Success!

THANK YOU



See you next time!

