Featured

Getting started

Hello, world

Simple web scraper

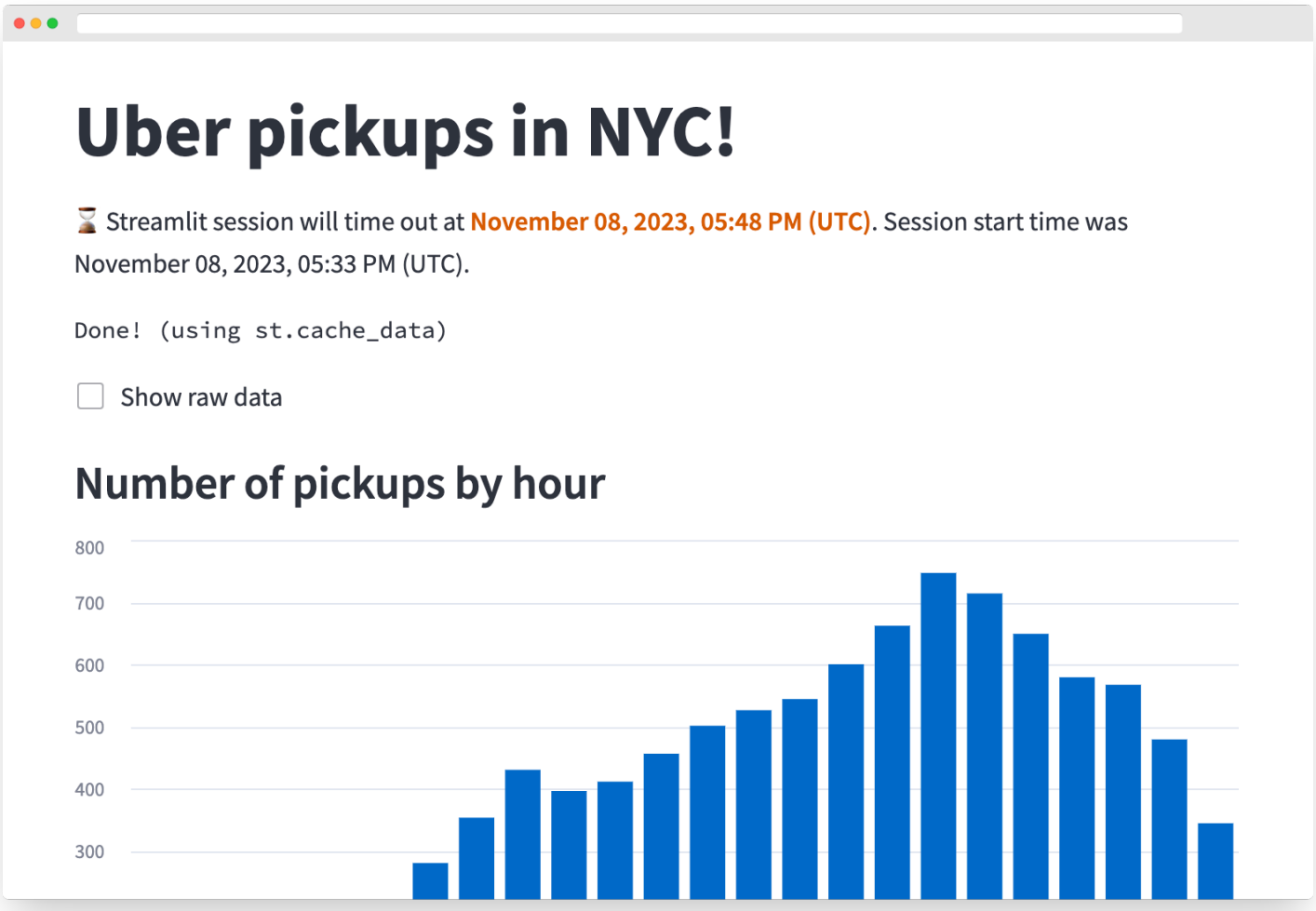
Serving web endpoints

Large language models (LLMs)

Run and share Streamlit apps

View on GitHub

This example shows you how to run a Streamlit app with `modal serve`, and then deploy it as a serverless web app.



This example is structured as two files:

1. This module, which defines the Modal objects (name the script `serve_streamlit.py` locally).
2. `app.py`, which is any Streamlit script to be mounted into the Modal function (download script).

```
import shlex
import subprocess
from pathlib import Path

import modal
```

Define container dependencies

The `app.py` script imports three third-party packages, so we include these in the example's image definition.

```
image = modal.Image.debian_slim(python_version="3.11").pip_install(
    "streamlit~=1.35.0", "numpy~=1.26.4", "pandas~=2.2.2"
)

app = modal.App(name="example-modal-streamlit", image=image)
```

Mounting the `app.py` script

We can just mount the `app.py` script inside the container at a pre-defined path using a Modal `Mount`.

```
streamlit_script_local_path = Path(__file__).parent / "app.py"
streamlit_script_remote_path = Path("/root/app.py")

if not streamlit_script_local_path.exists():
    raise RuntimeError(
        "app.py not found! Place the script with your streamlit app in the same directory."
    )

streamlit_script_mount = modal.Mount.from_local_file(
    streamlit_script_local_path,
    streamlit_script_remote_path,
)
```

Spawning the Streamlit server

Inside the container, we will run the Streamlit server in a background subprocess using `subprocess.Popen`. We also expose port 8000 using the `@web_server` decorator.

```
@app.function(
    allow_concurrent_inputs=100,
    mounts=[streamlit_script_mount],
)
@modal.web_server(8000)
def run():
    target = shlex.quote(str(streamlit_script_remote_path))
    cmd = f"streamlit run {target} --server.port 8000 --server.enableCORS=false --server.e
    subprocess.Popen(cmd, shell=True)
```

Iterate and Deploy

While you're iterating on your screamlit app, you can run it "ephemerally" with `modal serve`. This will run a local process that watches your files and updates the app if anything changes.

```
modal serve serve_streamlit.py
```

Once you're happy with your changes, you can deploy your application with

```
modal deploy serve_streamlit.py
```

If successful, this will print a URL for your app, that you can navigate to from your browser 🎉 .