Streamlit App Deployment

January 2024

Yonas M.

Outline

- Introduction
- Install Streamlit (Windows & Linux)
- Display texts with Streamlit
- Display an image, video or audio file with Streamlit
- Input widgets
- Progress and status bar with Streamlit
- Sidebar and container
- Display graphs with Streamlit
- Display maps with Streamlit
- Changing Themes
- Example Web App: BMI Calculator web app
- Build and deploy a machine learning application locally
- Deploy ML application app on Web

What is Streamlit?

From the data science pipeline, one of the most important steps is ***model deployment***.

We have a lot of options in python for deploying our model. Some popular frameworks are ***Flask*** and ***Django***.

But the issue with using these frameworks is that we should have some knowledge of ***HTML***, ***CSS***, and ***JavaScript***.

Using streamlit you can ***deploy*** any ***machine learning model*** and any ***python project*** with ease and without worrying about the ***frontend***.

Streamlit is very ***user-friendly***.

Why Streamlit?

Streamlit is a free and open-source framework to rapidly build and share machine learning and data science web apps.

It is a Python-based library specifically designed for machine learning engineers.

It is compatible with the majority of Python libraries (e.g. pandas, matplotlib, seaborn, plotly, Keras, PyTorch, SymPy(latex)).

Less code is needed to create amazing web apps.

Data caching simplifies and speeds up computation pipelines.

Python Virtual Environment

A virtual environment is an isolated environment for python projects.

It allows you to create an isolated environment for each python project.

This makes it easier for us to manage packages and dependencies throughout projects, especially where they share the same dependencies.

Installing Streamlit with Promit

Windows Command Promit

```
cd C:\Users\xxx\Documents\stremlit_projects

python -m venv streamlitenv2

streamlitenv2\Scripts\activate
streamlitenv2\Scripts\Activate.ps1

(streamlitenv1) C:\Users\xxx>

pip install streamlit

streamlit hello
python -m streamlit hello


streamlit run app.py
python -m streamlit run app.py
```

Windows Power Shell

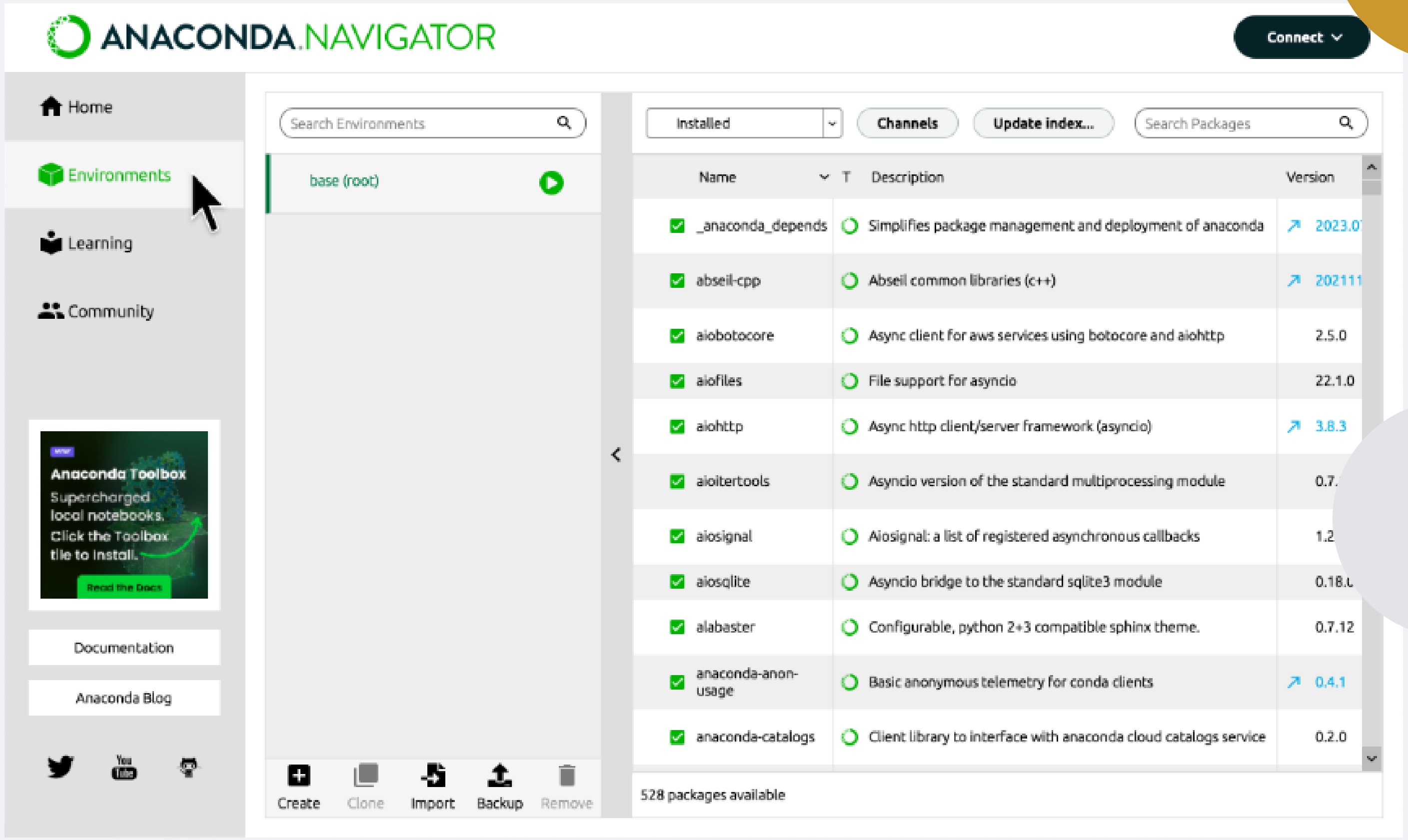
```
cd C:\Users\xxx\Documents\stremlit_projects

python -m venv streamlitenv2


streamlitenv2\Scripts\activate
streamlitenv2\Scripts\Activate.ps1

(streamlitenv1) C:\Users\xxx>

pip install streamlit

streamlit hello
python -m streamlit hello


streamlit run app.py
python -m streamlit run app.py
```

Installing Streamlit: Linux: Ubuntu

```
conda create -n streamlit_env1

conda activate streamlit_env1

pip install streamlit

streamlit hello
python -m streamlit hello

streamlit run app.py
python -m streamlit run app.py
```

Installing Streamlit using Anaconda

Create	Create an environment using Anaconda Navigator
Open	Open Anaconda Navigator (the graphical interface included with Anaconda Distribution).
Click	In the left menu, click "Environments".

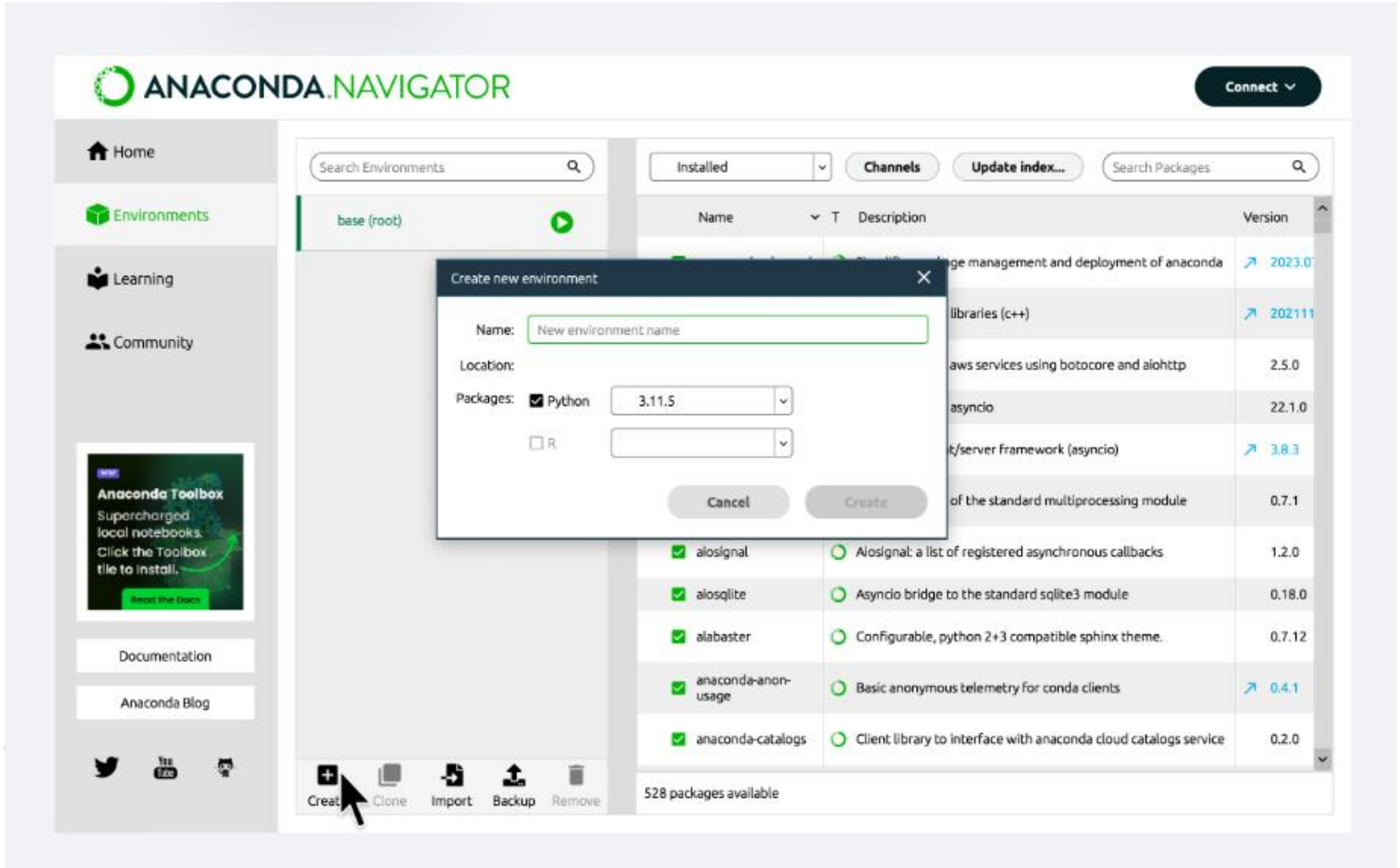


Name	T	Description	Version
_anaconda_depends		Simplifies package management and deployment of anaconda	2023.0
abseil-cpp		Abseil common libraries (c++)	202111
aiobotocore		Async client for aws services using botocore and aiohttp	2.5.0
aiofiles		File support for asyncio	22.1.0
aiohttp		Async http client/server framework (asyncio)	3.8.3
aioitertools		Asyncio version of the standard multiprocessing module	0.7.
aiosignal		Aiosignal: a list of registered asynchronous callbacks	1.2
aiosqlite		Asyncio bridge to the standard sqlite3 module	0.18.
alabaster		Configurable, python 2+3 compatible sphinx theme.	0.7.12
anaconda-anon-usage		Basic anonymous telemetry for conda clients	0.4.1
anaconda-catalogs		Client library to interface with anaconda cloud catalogs service	0.2.0

528 packages available

Installing Streamlit using Anaconda

- At the bottom of your environments list, click "**Create**".



Create new environment

Name: New environment name

Location:

Packages: ☑ Python 3.11.5

☐ R

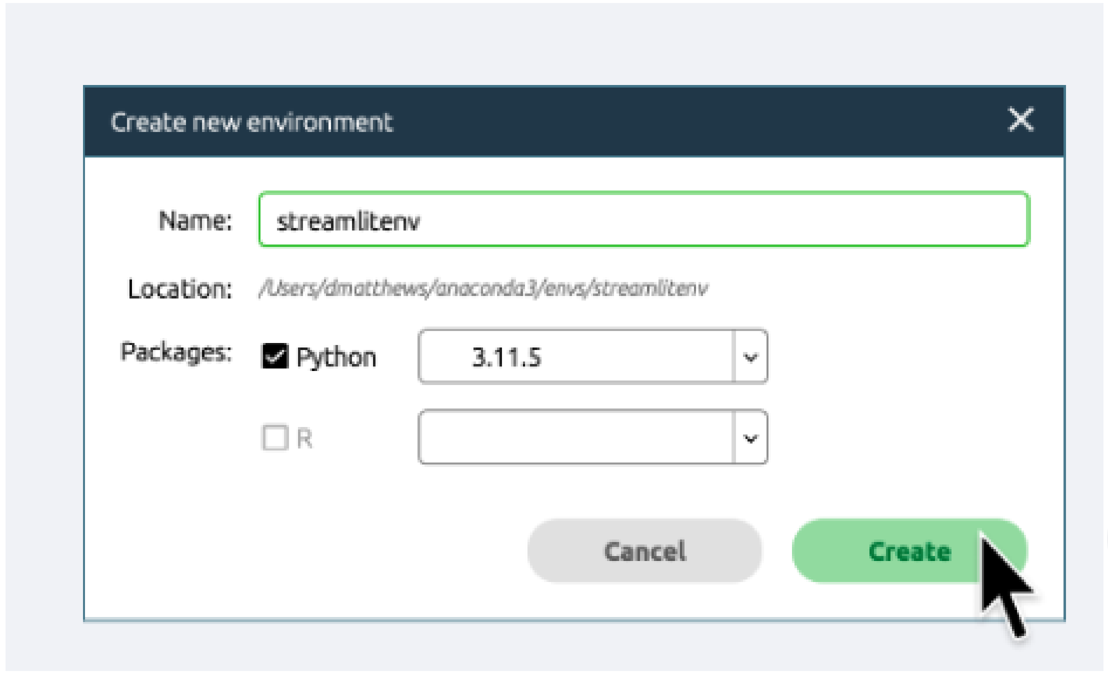
Cancel Create

Name	T	Description	Version
		...ge management and deployment of anaconda	2023.0
		...libraries (c++)	202111
		...aws services using botocore and aiohttp	2.5.0
		...asyncio	22.1.0
		...t/server framework (asyncio)	3.8.3
		...of the standard multiprocessing module	0.7.1
☑ aiosignal		Aiosignal: a list of registered asynchronous callbacks	1.2.0
☑ aiosqlite		Asyncio bridge to the standard sqlite3 module	0.18.0
☑ alabaster		Configurable, python 2+3 compatible sphinx theme.	0.7.12
☑ anaconda-anon-usage		Basic anonymous telemetry for conda clients	0.4.1
☑ anaconda-catalogs		Client library to interface with anaconda cloud catalogs service	0.2.0

528 packages available

Installing Streamlit using Anaconda

- Enter "streamlitenv" for the name of your environment.
- Click "**Create.**"


Create new environment

Name: streamlitenv

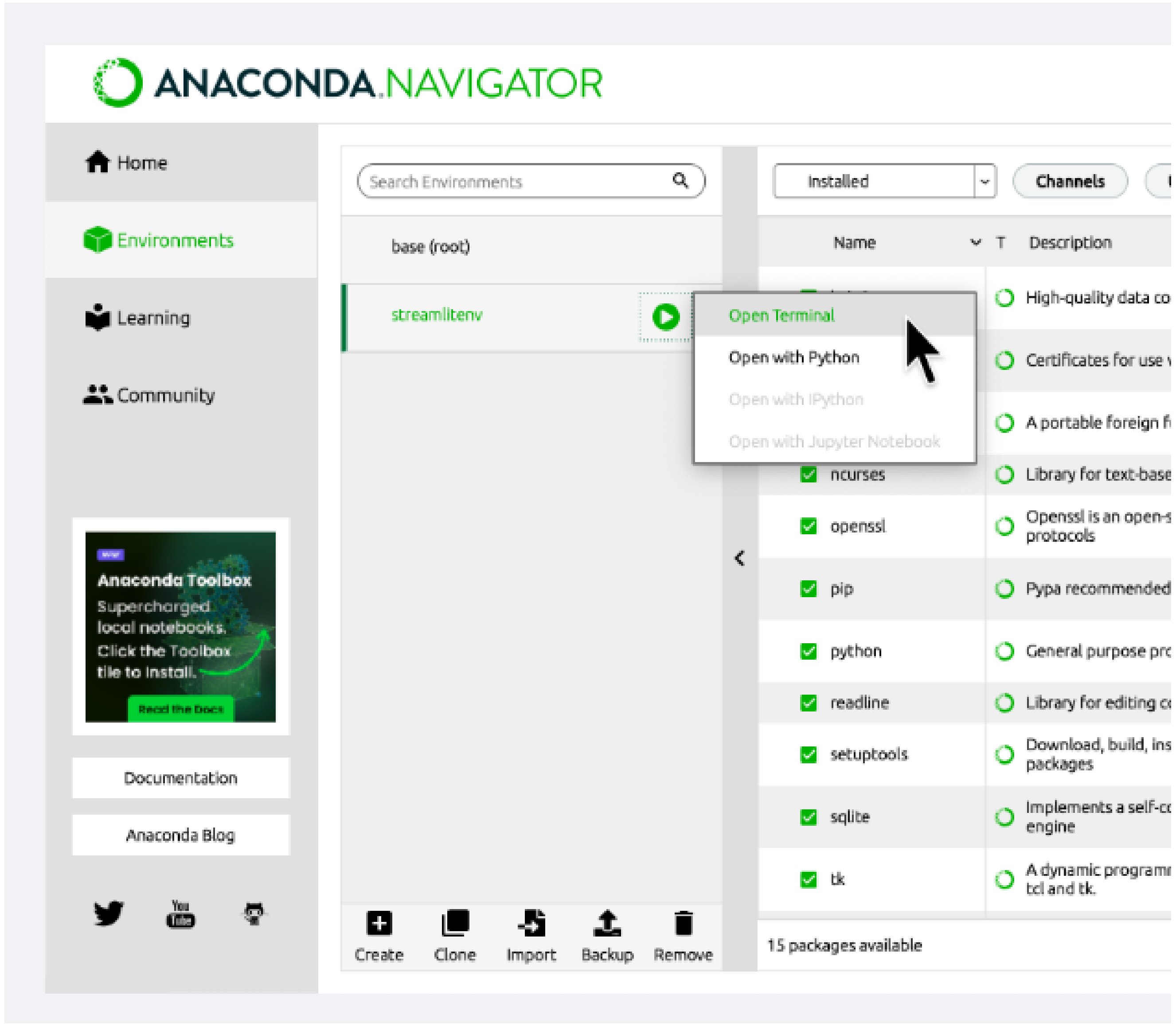
Location: /Users/dmatthews/anaconda3/envs/streamlitenv

Packages: ☑ Python 3.11.5

☐ R

Installing Streamlit using Anaconda

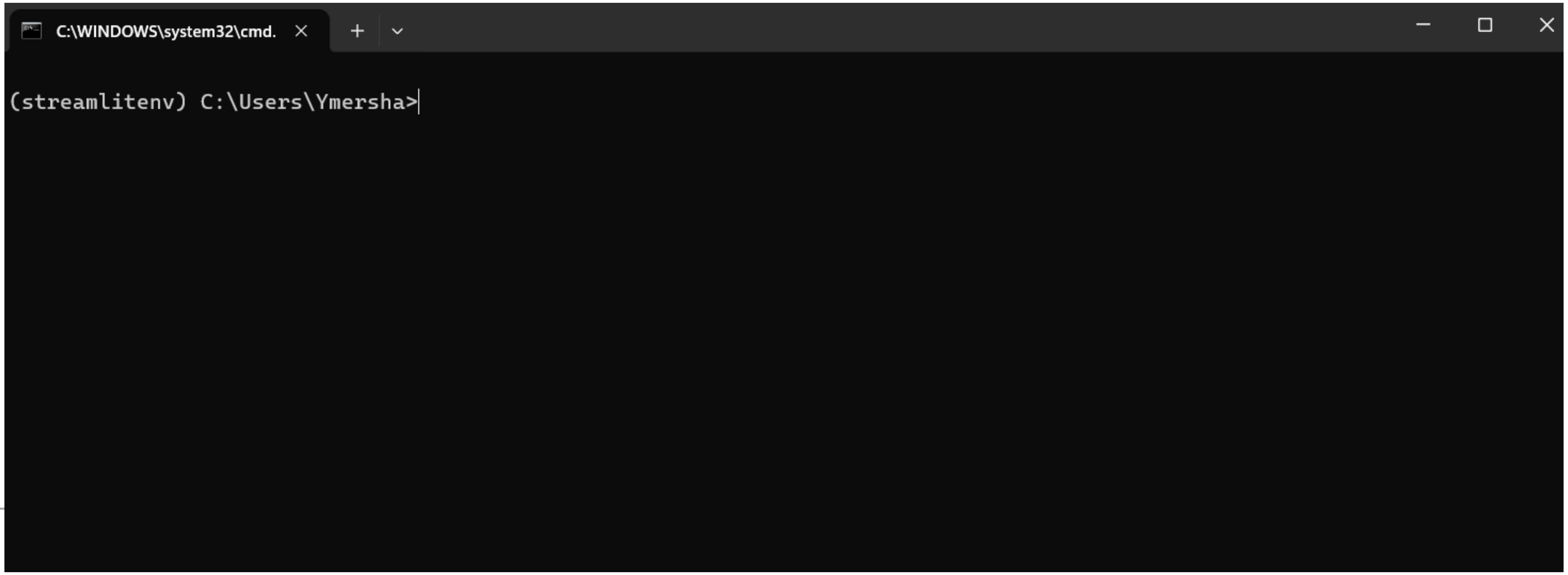
- Click the green play icon (play_circle) next to your environment.
- Click "Open Terminal."



Installing Streamlit using Anaconda

- A terminal will open with your environment activated.

- Your environment's name will appear in parentheses at the beginning of your terminal's prompt to show that it's activated.



```
(streamlitenv) C:\Users\Ymersha>
```

Installing Streamlit using Anaconda

- In your terminal, type:

```
pip install streamlit

streamlit hello
python -m streamlit hello
```

- The Streamlit Hello example app will automatically open in your browser.

- If it doesn't, open your browser and go to the localhost address indicated in your terminal, typically `http://localhost:8501`.

Streamlit: Hello World

```
import streamlit as st

st.write("Hello World")
```

- Download VS Code & install it: https://code.visualstudio.com/download
- Create project folder
- Open **VS Code** Editor with the new project
- Create a Python file named ***app.py*** in your project folder
- Copy the following code into app.py and save it.
- Activate the streamlit environment
- In your terminal, type: ***streamlit run app.py***
- If this doesn't work, use the long-form command: ***python -m streamlit run app.py***
- When you're done, you can stop your app with Ctrl+C in your terminal or just by closing your terminal

Display texts with Streamlit

st.title(): This function allows you to add the title of the app.

st.header(): This function is used to set header of a section.

st.markdown(): This function is used to set a markdown of a section.

st.subheader(): This function is used to set sub-header of a section.

st.caption(): This function is used to write caption.

st.code(): This function is used to set a code.

st.latex(): This function is used to display mathematical expressions formatted as LaTeX.

Markdown

```python
st.markdown("# This is a markdown")

st.markdown("## This is a markdown")

st.markdown("### This is a markdown")

st.markdown("#### This is a markdown")

st.markdown("##### This is a markdown")

st.markdown("###### This is a markdown")
```

Title, Header, Subheader, text, & write

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

st.header("This is a header")

st.subheader("This is a subheader")

st.text("This is a text")

st.write("This is a write")
```

Write Function

- Using write function, we can also display code in ***coding format***.

- This is not possible using st.text("   ").

```
# Write text
st.write("Text with write")

# Writing python inbuilt function range()
st.write(range(10))

st.text(range(10))
```

Success, Info, Warning, Error, & Exception

```
# Success
st.success("Success")

# Information
st.info("Information")

# Warning
st.warning("Warning")

# Error
st.error("Error")

# Exception - This has been added later
exp = ZeroDivisionError("Trying to divide by Zero")
st.exception(exp)
```

Caption, Code, & latex

- st.caption("this is the caption")
- st.code("x=2021")
- st.latex(r''' a+a r^1+a r^2+a r^3 ''')

Display Images

- Using write function, we can also display code in ***coding format***.

- This is not possible using st.text("   ").

```
# import Image from pillow to open images
from PIL import Image

img = Image.open("streamlit.png")

# display image using streamlit
st.image(img, width=200, caption='streamlit logo')
```

Display an image, video or audio file with Streamlit

- st.image("kid.jpg")
- st.audio("Audio.mp3")
- st.video("video.mp4")

Input widgets 1

- Streamlit has various widgets that allow you to interact directly with your app.
- Widgets are the most important user interface components.
- ***st.checkbox()***: This function returns a Boolean value. When the box is checked, it returns a True value, otherwise a False value.
- ***st.button()***: This function is used to display a button widget.
- ***st.radio()***: This function is used to display a radio button widget.
- ***st.selectbox()***: This function is used to display a select widget.
- ***st.multiselect()***: This function is used to display a multiselect widget.
- ***st.select_slider()***: This function is used to display a select slider widget.

- ***st.slider()***: This function is used to display a slider widget.

Checkbox

- A checkbox returns a **boolean value**.
- When the box is checked, it returns a **True** value else returns a **False** value.

```
# title of the checkbox is 'Show/Hide'
# display the text if the checkbox returns True value

if st.checkbox("Show/Hide"):
   st.text("Showing the widget")
```

Button

- **st.button()** returns a **boolean value**.
- It returns a **True** value when clicked else returns **False**.

```python
# Create a simple button that does nothing
st.button("Click me for no reason")

# Create a button, that when clicked, shows a text
if(st.button("About")):
    st.text("This button is clicked")
```

Radio Button

```
status = st.radio("Select Gender: ", ('Male', 'Female'))

if (status == 'Male'):
    st.success("Male")
else:
    st.success("Female")
```

```
status1 = st.radio("Select Gender: ", ('xxx', 'yyy', 'zzz'))

st.write("Your status is: ", status1)
```

Selection Box & Multi-Selectbox

```
hobby = st.selectbox("Hobbies: ", ['Dancing', 'Reading', 'Sports'])

st.write("Your hobby is: ", hobby)
```

```
hobbies = st.multiselect("Hobbies: ",  ['Dancing', 'Reading', 'Sports'])

st.write("You selected", len(hobbies), 'hobbies')
```

Slider & Select_Slider

```
level = st.slider("Select the level", 1, 5)

# .format() is used to print value of a variable at a
specific position

st.text('Selected: {}'.format(level))
```

```
st.select_slider('Pick a mark', ['Bad', 'Good', 'Excellent'])
```

Input widgets 2

- ***st.number_input()***: This function is used to display a numeric input widget.
- ***st.text_input()***: This function is used to display a text input widget.
- ***st.date_input()***: This function is used to display a date input widget to choose a date.
- ***st.time_input()***: This function is used to display a time input widget to choose a time.
- ***st.text_area()***: This function is used to display a text input widget with more than a line text.
- ***st.file_uploader()***: This function is used to display a file uploader widget.
- ***st.color_picker()***: This function is used to display color picker widget to choose a color.

Input widgets 2

```
st.number_input('Pick a number', 0,10)

st.text_input('Email address')

st.date_input('Travelling date')

st.time_input('School time')

st.text_area('Description')

st.file_uploader('Upload a photo')

st.color_picker('Choose your favorite color')
```

Progress and status bar with Streamlit

- ***st.balloons()***: This function is used to display balloons for celebration.
- ***st.progress()***: This function is used to display a progressbar.
- ***st.spinner()***: This function is used to display a temporary waiting message during execution.

```
st.balloons()

st.subheader("progress bar")
st.progress(10)

import time
st.subheader("with the execution")
with st.spinner('Wait for it...'):
    time.sleep(10)
```

Sidebar and Container

- Sidebar and container are used to organize your app on your page .

st.sidebar() will make element pinned to the left, allowing users to focus on the content in your app.

```
st.sidebar.title("This is side bar ")
st.sidebar.button("Click ")
st.sidebar.radio("Pick Gender", ["Male", "Feamle"] )
```

st.container() is used to create an invisible container where you can put elements in order to create a useful arrangement and hierarchy.

```
container = st.container()
container.write("This is written inside the container")
st.write("This is written inside the container")
```

Display graphs with Streamlit: pyplot

```
import matplotlib.pyplot as plt
import numpy as np

rand=np.random.normal(1, 2, size=20)

fig, ax = plt.subplots()

ax.hist(rand, bins=15)

st.pyplot(fig)
```

random.normal(): generates an array of 20 random numbers with a mean of 1 & a standard deviation of 2.

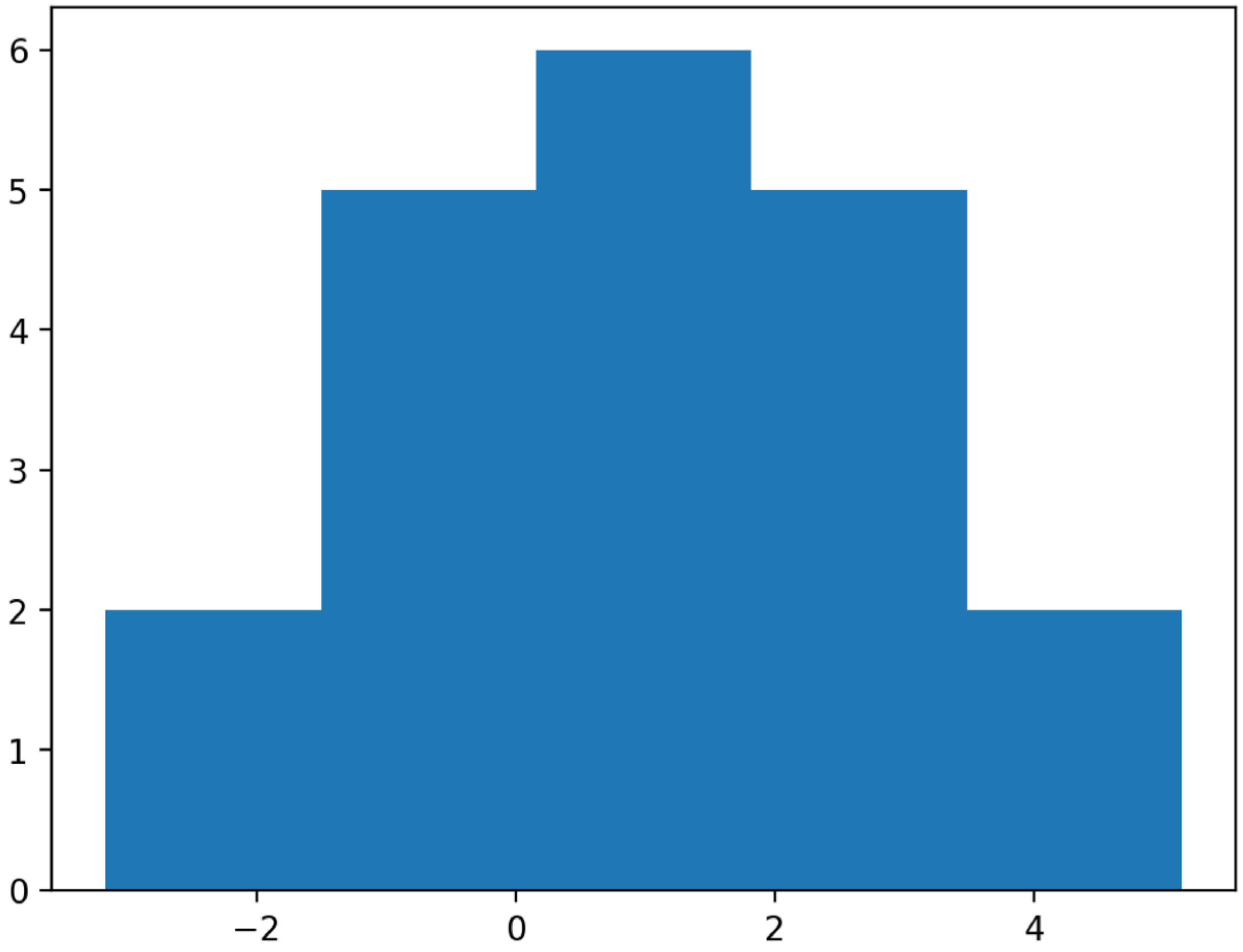
hist(): creates a histogram of the random numbers with 15 bins.

Fig: object is stores the histogram result

ax: object is used to manipulate the plot

st.pyplot() from Streamlit is used to display the histogram in the Streamlit app.

Plot with pyplot



Line and Bar Chart

```
import pandas as pd

df= pd.DataFrame(np.random.randn(10,
2),  columns=['x', 'y'])

st.line_chart(df)

st.bar_chart(df)
```

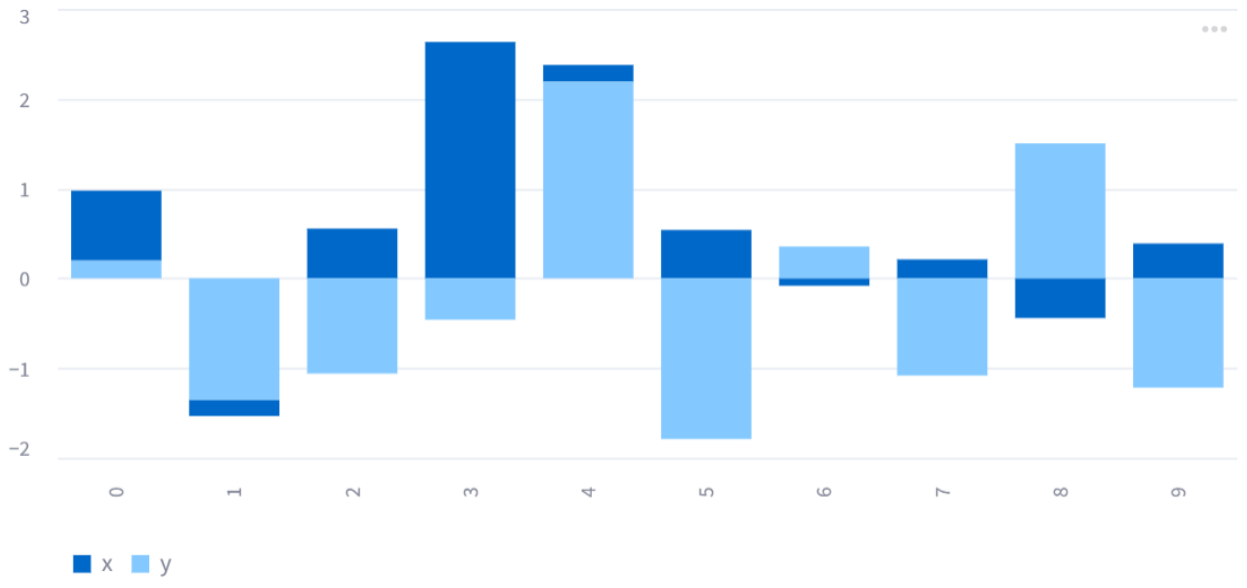
np.random.randn(): creates a Pandas DataFrame with 10 rows and 2 columns, where the values are randomly generated

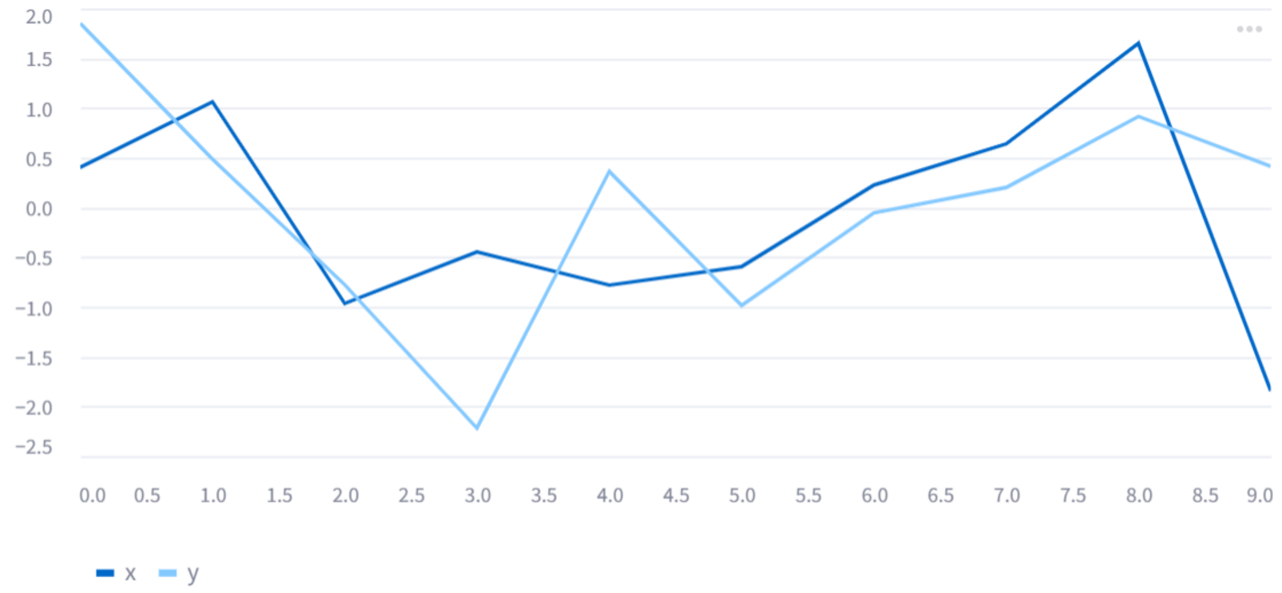
The columns are labeled 'x' and 'y'.

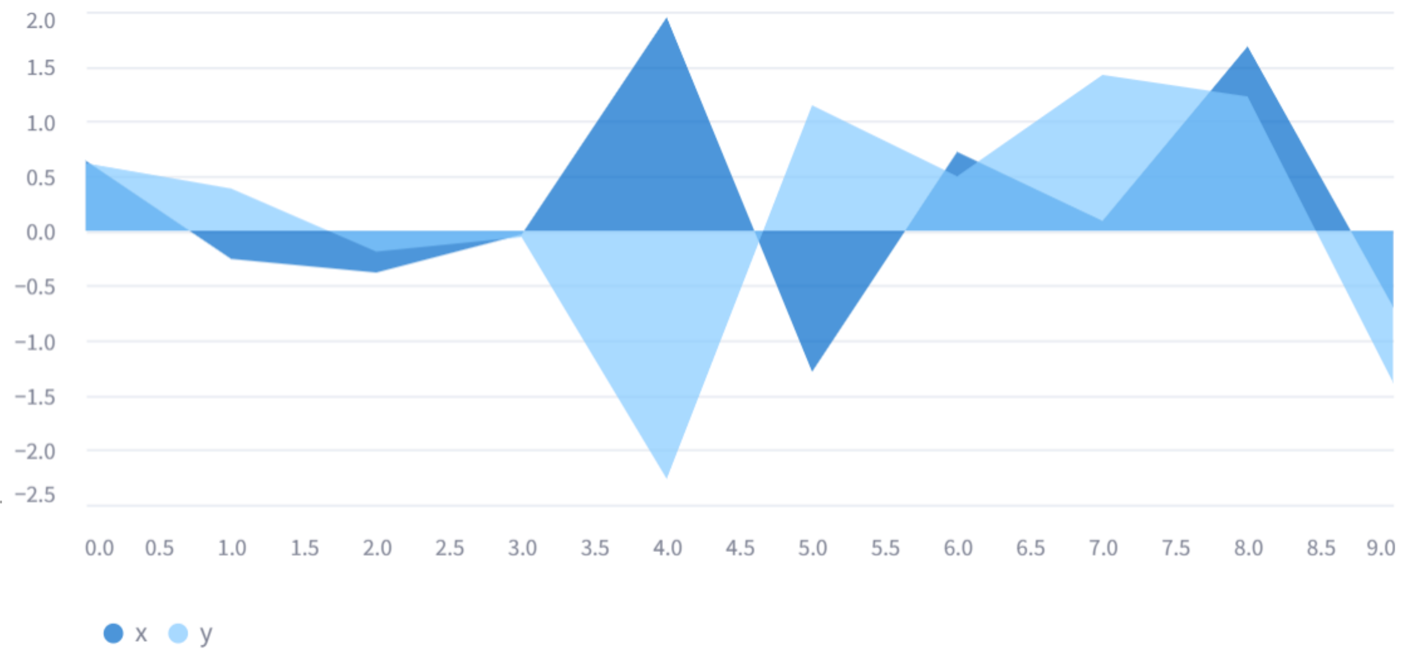
st.line_chart(): from the streamlit library to display a line chart

`st.bar_chart(df):` from the streamlit library to display a bar chart

Bar, Line, Area Chart

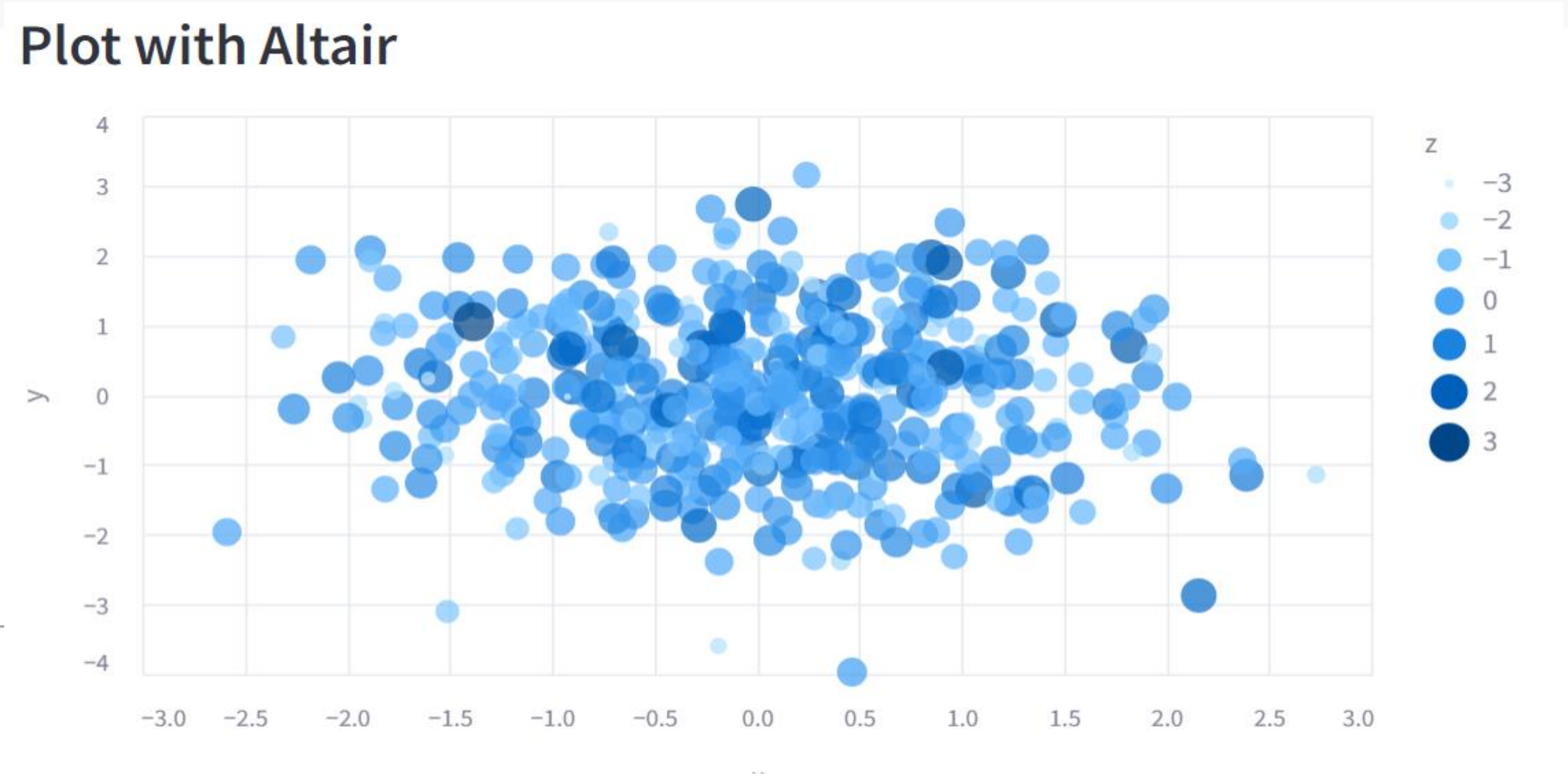






Plot with Altair

```
import altair as alt

df= pd.DataFrame(np.random.randn(500, 3),   columns=['x','y','z'])
c = alt.Chart(df).mark_circle().encode(   x='x', y='y' , size='z', color='z',
tooltip=['x', 'y', 'z'])
st.altair_chart(c, use_container_width=True)
```


Plot with Altair

Display maps with Streamlit

- ***st.map()***: This function is used to display maps in the app.

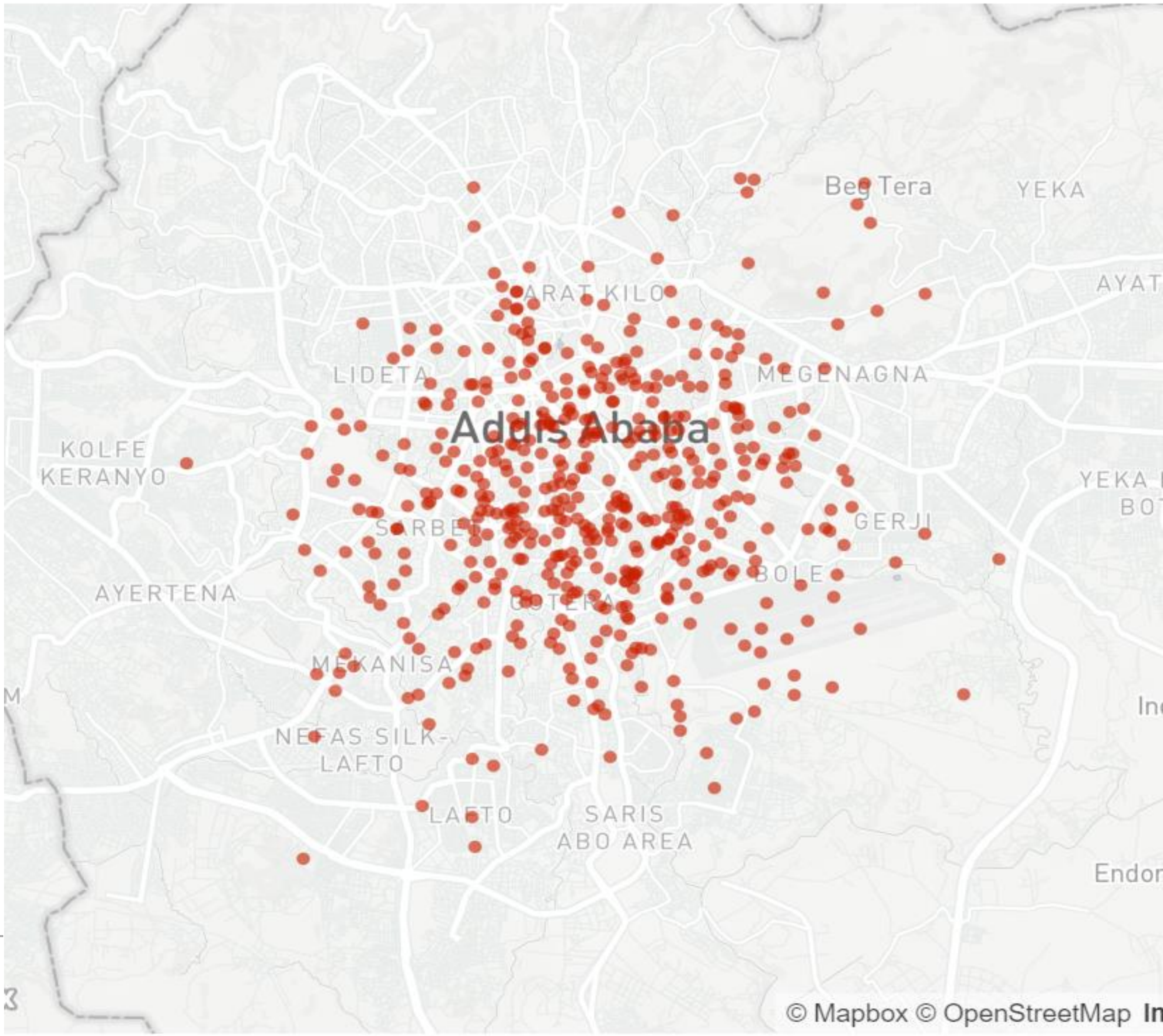
- Create a pandas DataFrame with 500 rows and 2 columns, where the values are randomly generated.

- The values are then divided by [50, 50] and added to the coordinates [9.00, 38.763] to create a set of latitude and longitude coordinates.

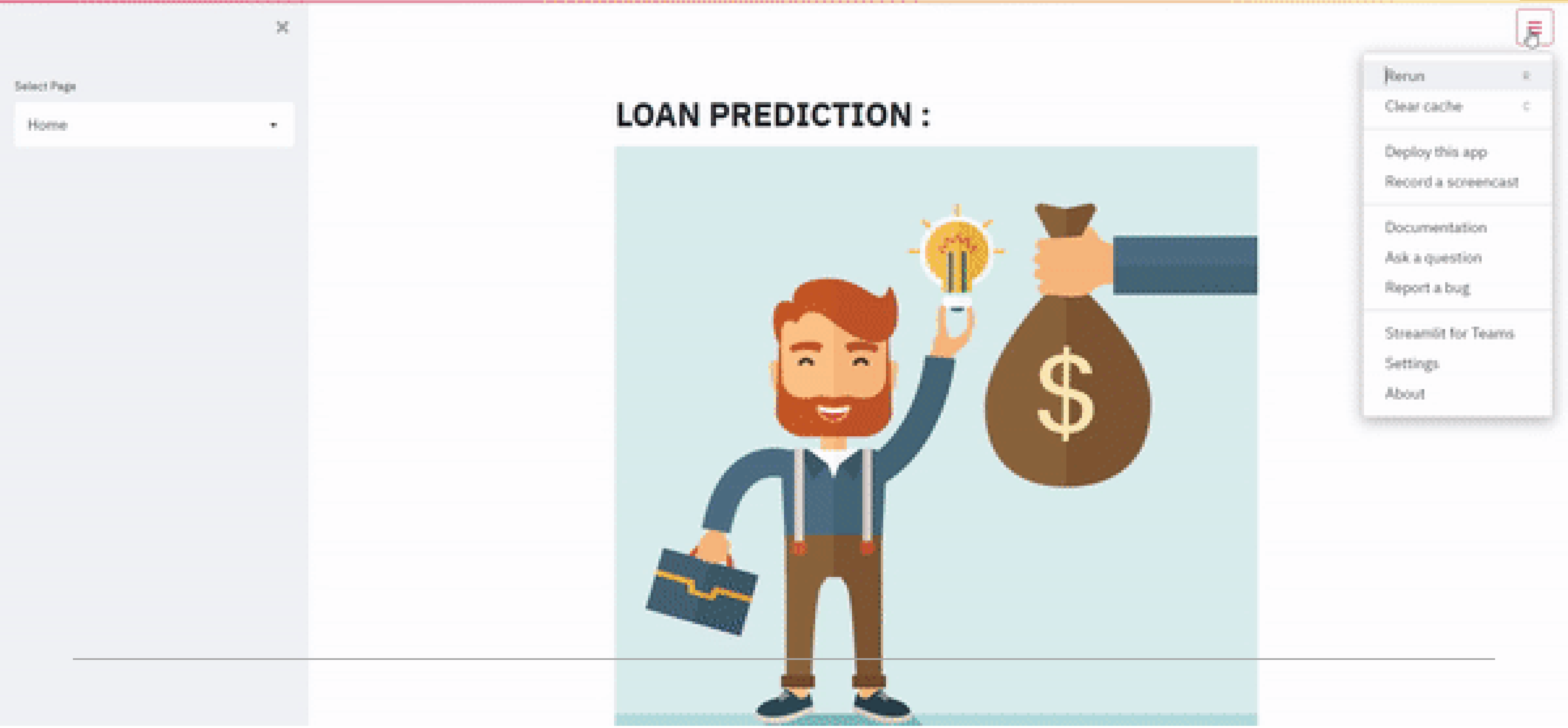
- The columns are labeled 'lat' and 'lon'.

- **st.map** function from the streamlit library is used to display the coordinates on a map.

Display Maps with Streamlit



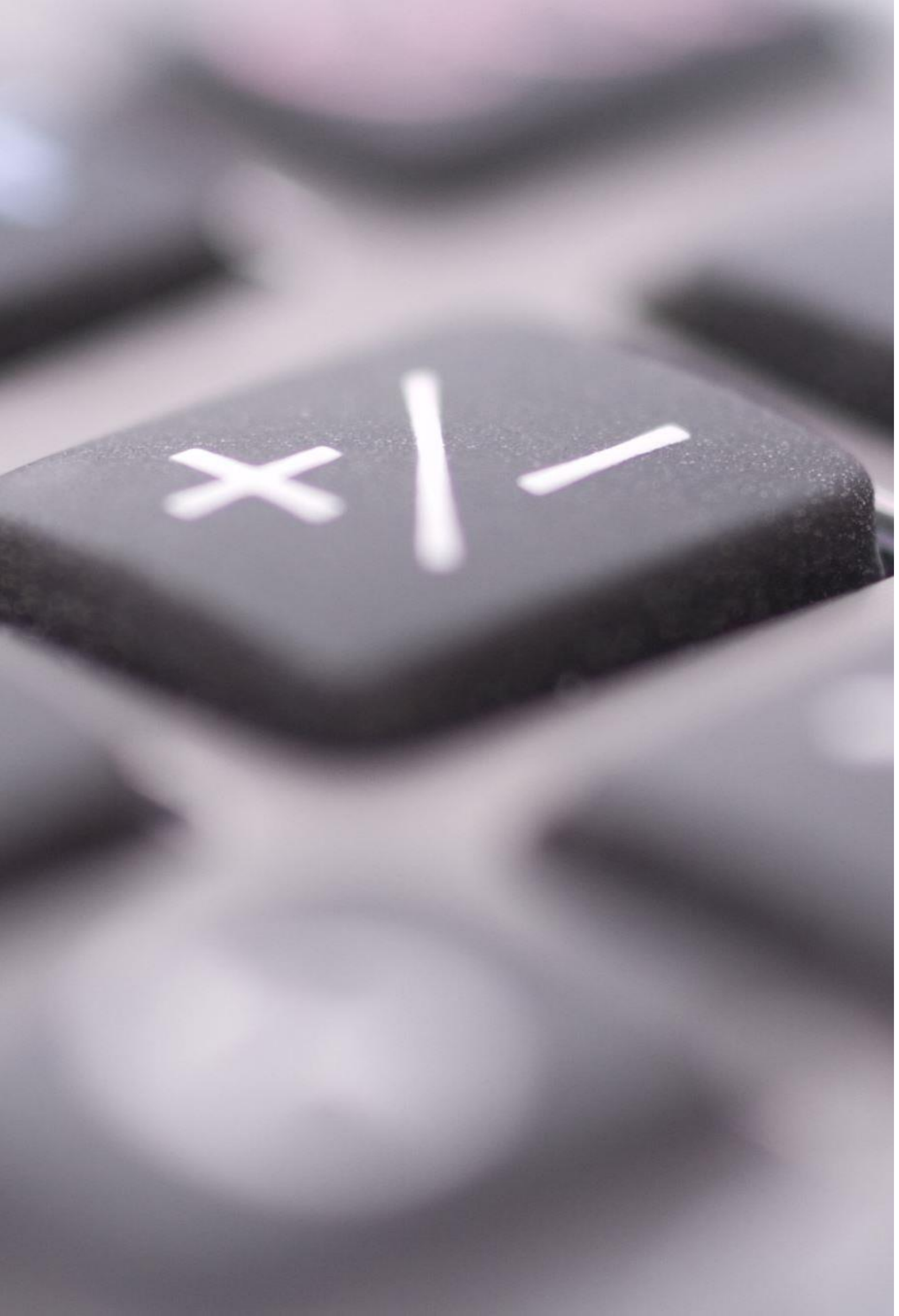
Change Themes



Example: BMI Calculator web app

- Let us recollect everything that we learn above and create a BMI Calculator web app.
- To calculate **BMI** Index the **weight is required in Kgs** and **height is in meters**

$$bmi = weight/height^2$$



BMI Calculator web app

- Import The Streamlit Library
- Give Title To The App
- Take Weight Input
- Take Height Input
- Calculate The BMI
- Print The BMI INDEX
- Interpret BMI Index

Example 1: BMI Calculator web app

```python
# Import the streamlit library
import streamlit as st

# Give a title to the app
#st.title('Welcome to BMI Calculator')

# blue, green, orange, red, violet, gray/grey, rainbow.
st.title(':blue[Welcome to BMI Calculator] 🧮 ')

# TAKE WEIGHT INPUT in kgs
weight = st.number_input("Enter your weight (in kgs)")

# TAKE HEIGHT INPUT
# radio button to choose height format
status = st.radio('Select your height format ', ('cms',
'meters', 'feet'))

# take height input in centimeters

if(status == 'cms'):

    height = st.number_input('Centimeters')

    try:
        bmi = weight / ((height/100)**2)
    except:
        st.text("Enter some value of height")

elif(status == 'meters'):
    # take height input in meters
    height = st.number_input('Meters')

    try:
        bmi = weight / (height ** 2)
    except:
        st.text("Enter some value of height")

else:
    # take height input in feet
    height = st.number_input('Feet')

    # 1 meter = 3.28 feet
    try:
        bmi = weight / (((height/3.28))**2)

    except:
        st.text("Enter some value of height")

# check if the button is pressed or not
if(st.button('Calculate BMI')):

    # print the BMI INDEX
    st.text("Your BMI Index is {}.".format(bmi))
```

BMI Calculator web app

```
if(st.button('Interprate BMI Result')):
    if(bmi < 16):
        st.error("You are Extremely Underweight")
    elif(bmi >= 16 and bmi < 18.5):
        st.warning("You are Underweight")
    elif(bmi >= 18.5 and bmi < 25):
        st.success("Healthy")
    elif(bmi >= 25 and bmi < 30):
        st.warning("Overweight")
    elif(bmi >= 30):
        st.error("Extremely Overweight")
```

Deploy a Machine Learning Model Iris Species Classifier

TRAIN RANDOM FOREST ML MODEL

- RANDOM_FOREST_CLASSIFIER_MODEL.IPYNB

DEPLOY THE MODEL USING STREAMLIT

Deploy a Machine Learning Model Iris Species Classifier

```
# Importing Libraries

# import Streamlit library as st
import streamlit as st
# import the pickle module, which is used for serializing and deserializing Python objects.
import pickle

# Setting Title
# st.title("Iris Flower Prediction")
st.title(':blue[Flower Species Classifier ML Model]')

# Loading the Classifier Model

# opens the  'classifier.pkl' in binary read mode ('rb')
pickle_in = open('classifier.pkl', 'rb')

# ~~load the machine learning model (classifier) using pickle.load()~~
classifier = pickle.load(pickle_in)
```

Deploy a Machine Learning Model Iris Species Classifier

```
# Defining Prediction Function
def prediction(sepal_length, sepal_width, petal_length,
petal_width):
    prediction = classifier.predict(
        [[sepal_length, sepal_width, petal_length, petal_width]])
    return prediction

# Taking User Input

sepal_length = st.number_input('Sepal Length')
sepal_width = st.number_input('Sepal Width')
petal_length = st.number_input('Petal Length')
petal_width = st.number_input('Petal Width')
```

Deploy a Machine Learning Model Iris Species Classifier

```
# Making Prediction on Button Click

# Empty string result is initialized

result =""

if st.button("Predict"):
    result = prediction(sepal_length, sepal_width, petal_length, petal_width)
    # st.success('The output is {}'.format(result))
    if(result == 0):
        st.success("Iris-setosa")
    elif(result == 1 ):
        st.success("Iris-versicolor")
    elif(result == 2):
        st.success("Iris-virginica")
```

How to deploy a Streamlit App

Deploying an application is the process of ***copying***, ***configuring***, and ***enabling*** a specific application to a specific base URL.

Deployment is the mechanism through which applications are delivered from developers to users.

Once the deployment process has finished, the application becomes publicly accessible on the base URL.

The server carries out this two-step process by first **staging** the application, and then **activating** it after successful staging.

Create a **GitHub Account:** https://github.com/

Create a **new repository** on your GitHub where you need to put your app code and dependencies.

How to deploy a Streamlit App

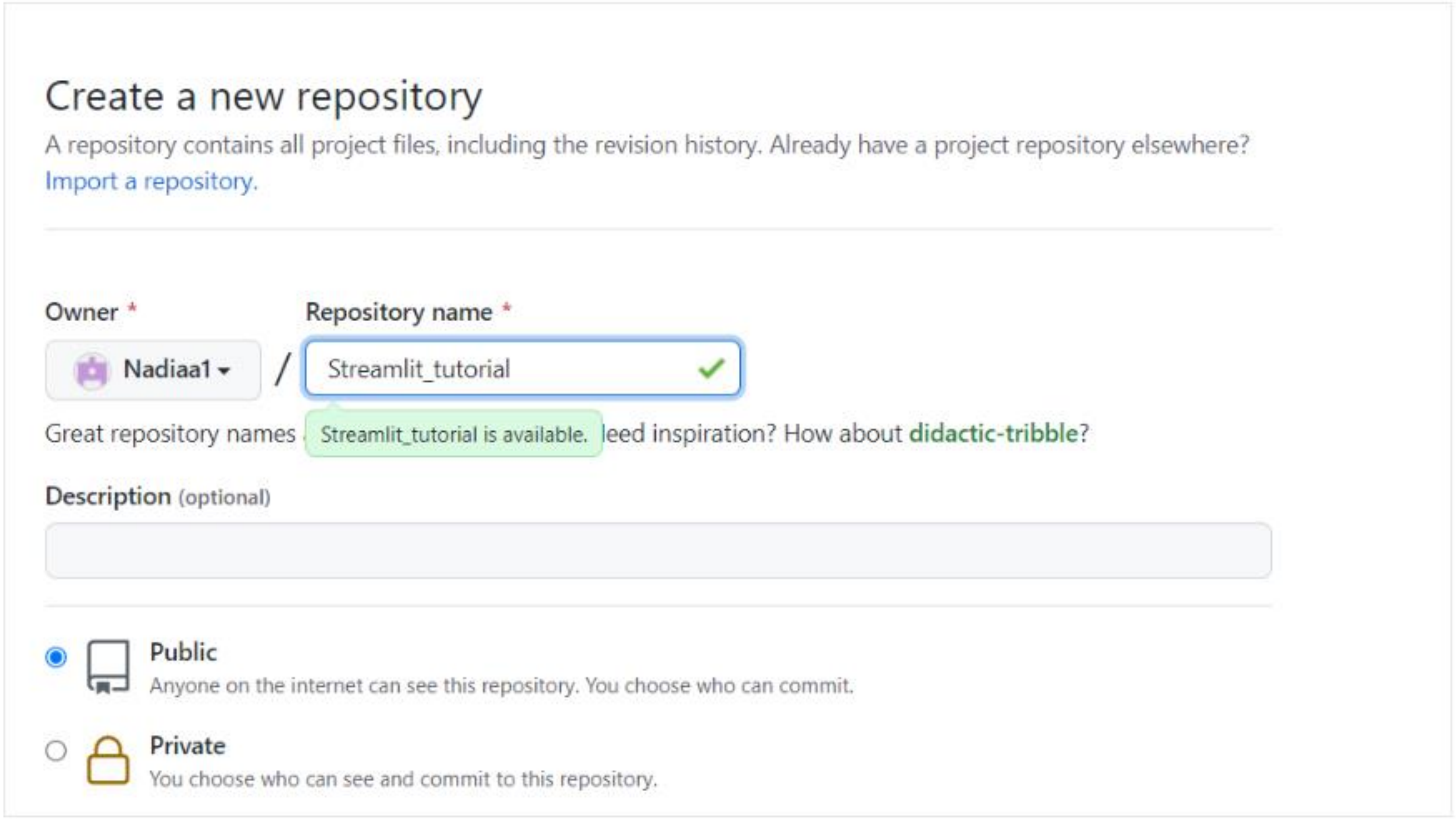
Upload your codes and data to the newly created repository

Create a new file named requirements where you have to put the libraries you used in your app.

Go to this website and link your GItHub account with the streamlit cloud: https://share.streamlit.io/

On the streamlit cloud: New App, then Deploy

Create New GitHub Repository



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
Import a repository.

Owner *

Nadiaa1

/

Repository name *

Streamlit_tutorial ✓

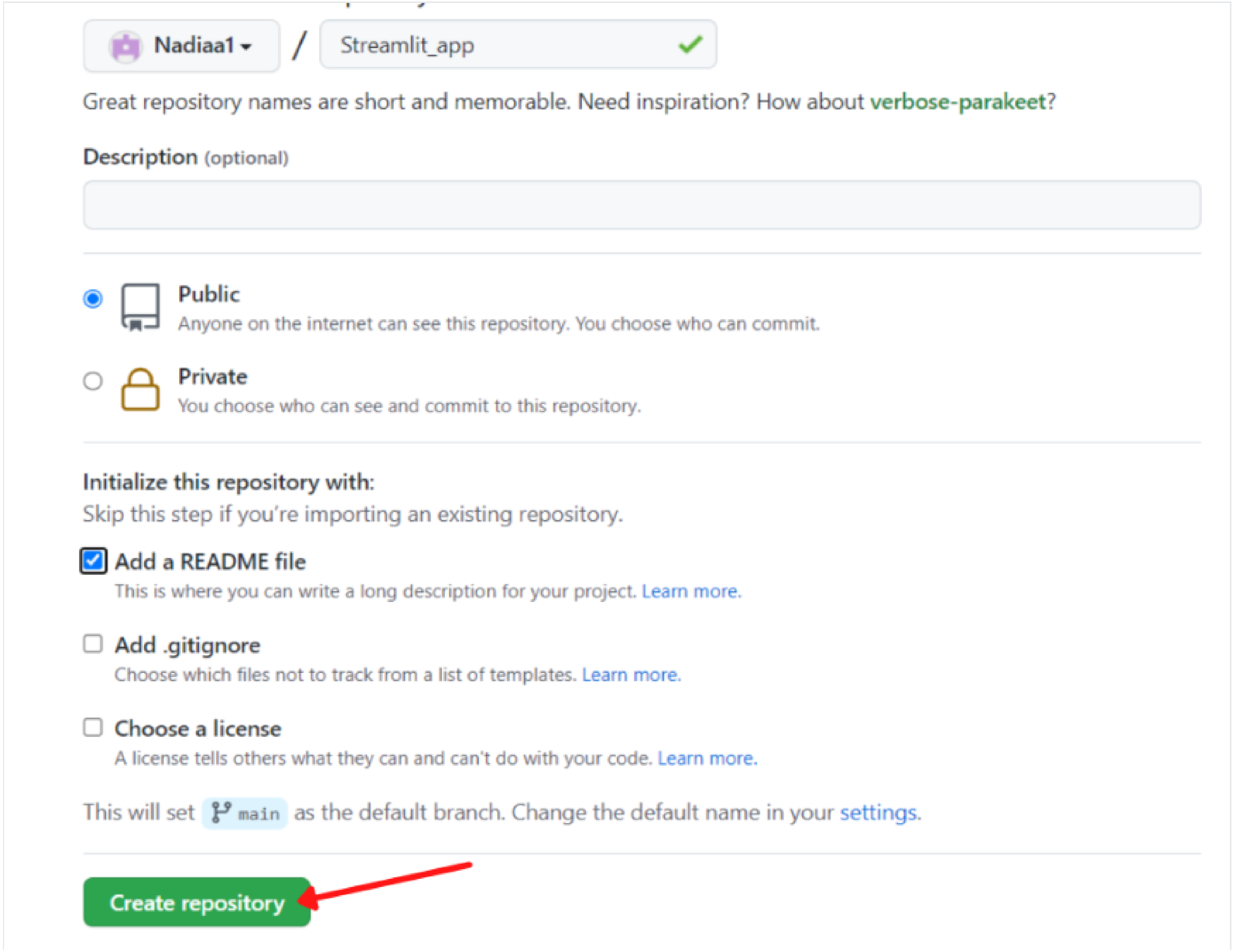
Great repository names [Streamlit_tutorial is available.] leed inspiration? How about **didactic-tribble**?

Description (optional)

[✓ Public]
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Create New GitHub Repository



Nadiaa1 / Streamlit_app ✓

Great repository names are short and memorable. Need inspiration? How about verbose-parakeet?

Description (optional)

[✓ Public]
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

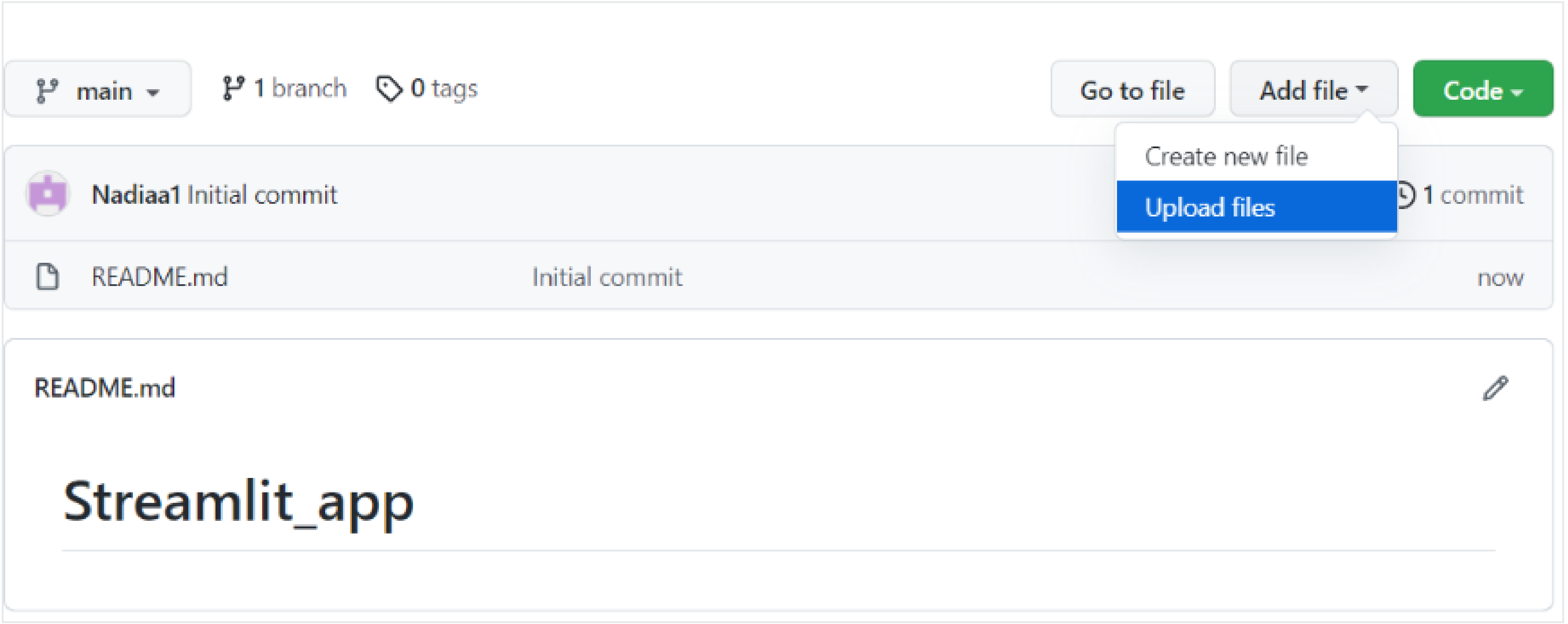
[✓] Add a README file
This is where you can write a long description for your project. Learn more.

[ ] Add .gitignore
Choose which files not to track from a list of templates. Learn more.

[ ] Choose a license
A license tells others what they can and can't do with your code. Learn more.

This will set `main` as the default branch. Change the default name in your settings.

Upload files to GitHub Repository

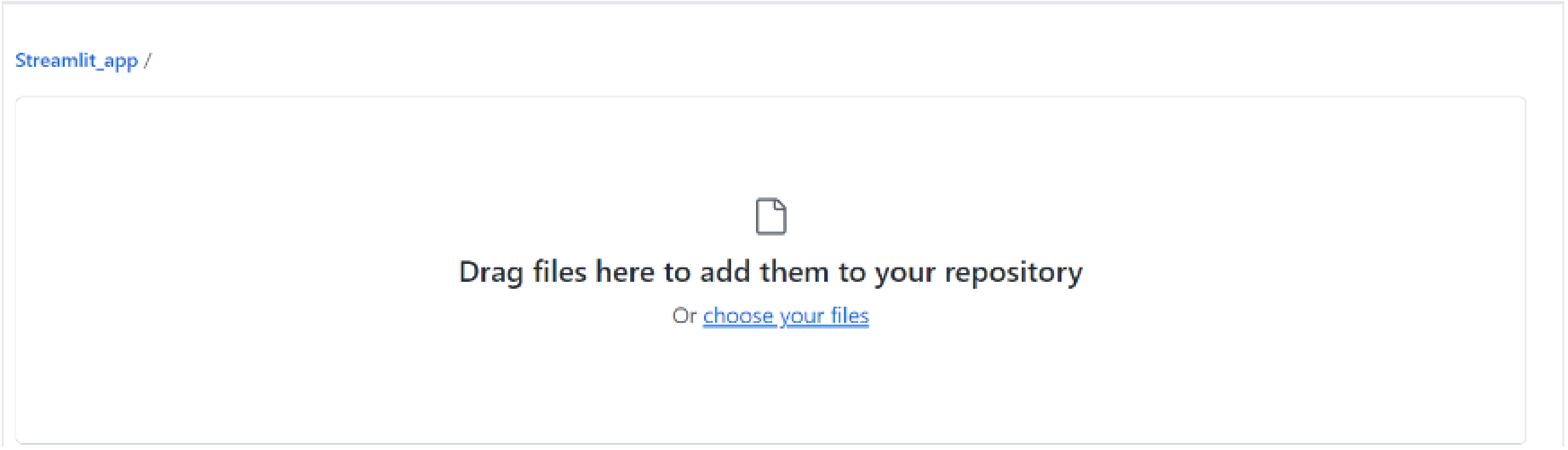


Nadiaa1 Initial commit

README.md	Initial commit	now

README.md

Streamlit_app

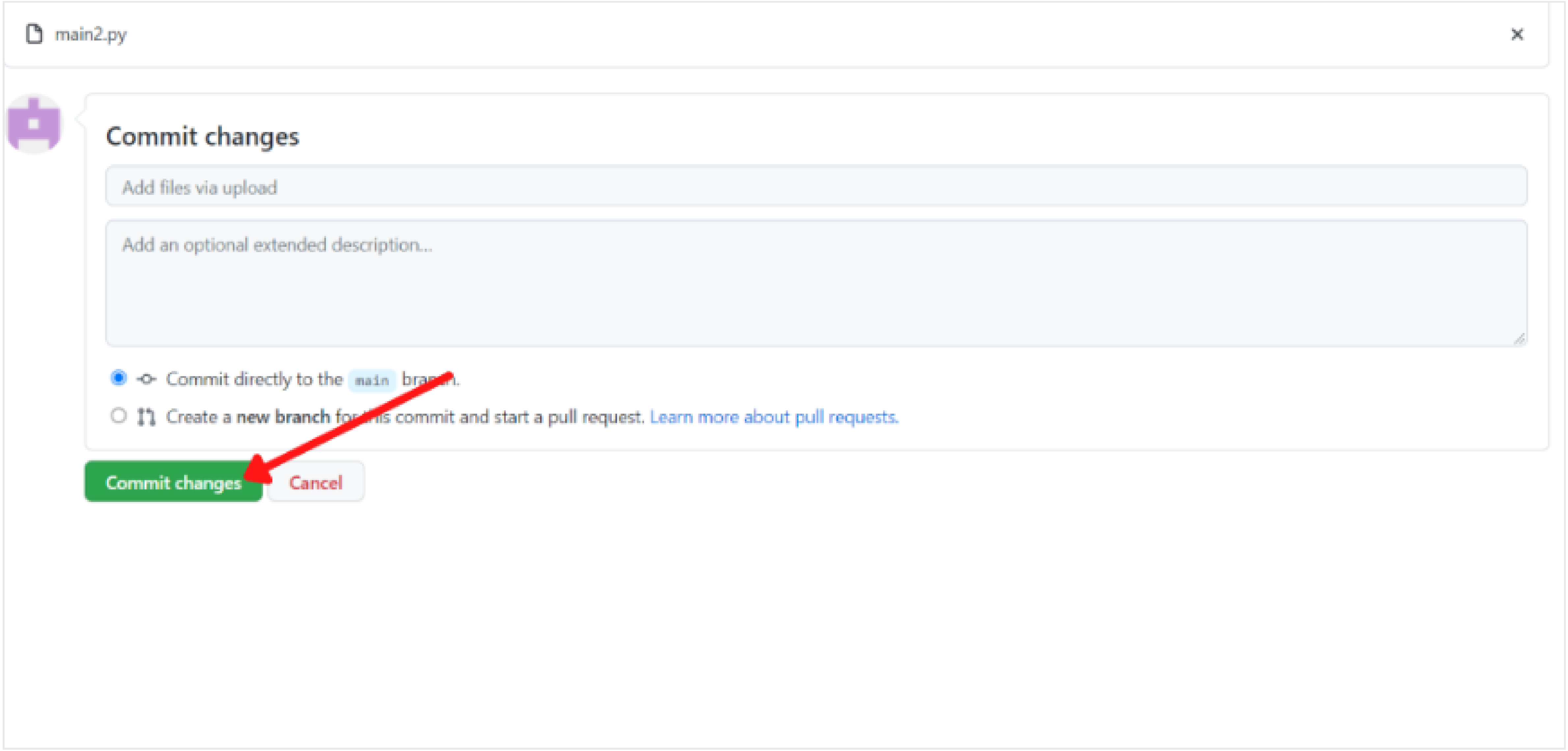


Streamlit_app /

Drag files here to add them to your repository

Or choose your files

Commit Chages



main2.py

Commit changes

Add files via upload

Add an optional extended description...

[✓ Commit directly to the `main` branch.]

Create a **new branch** for this commit and start a pull request. Learn more about pull requests.

Create requirements file

- A ***requirement file*** is a type of file that usually stores information about all the libraries, modules, and packages specific to the project used while developing a particular project.

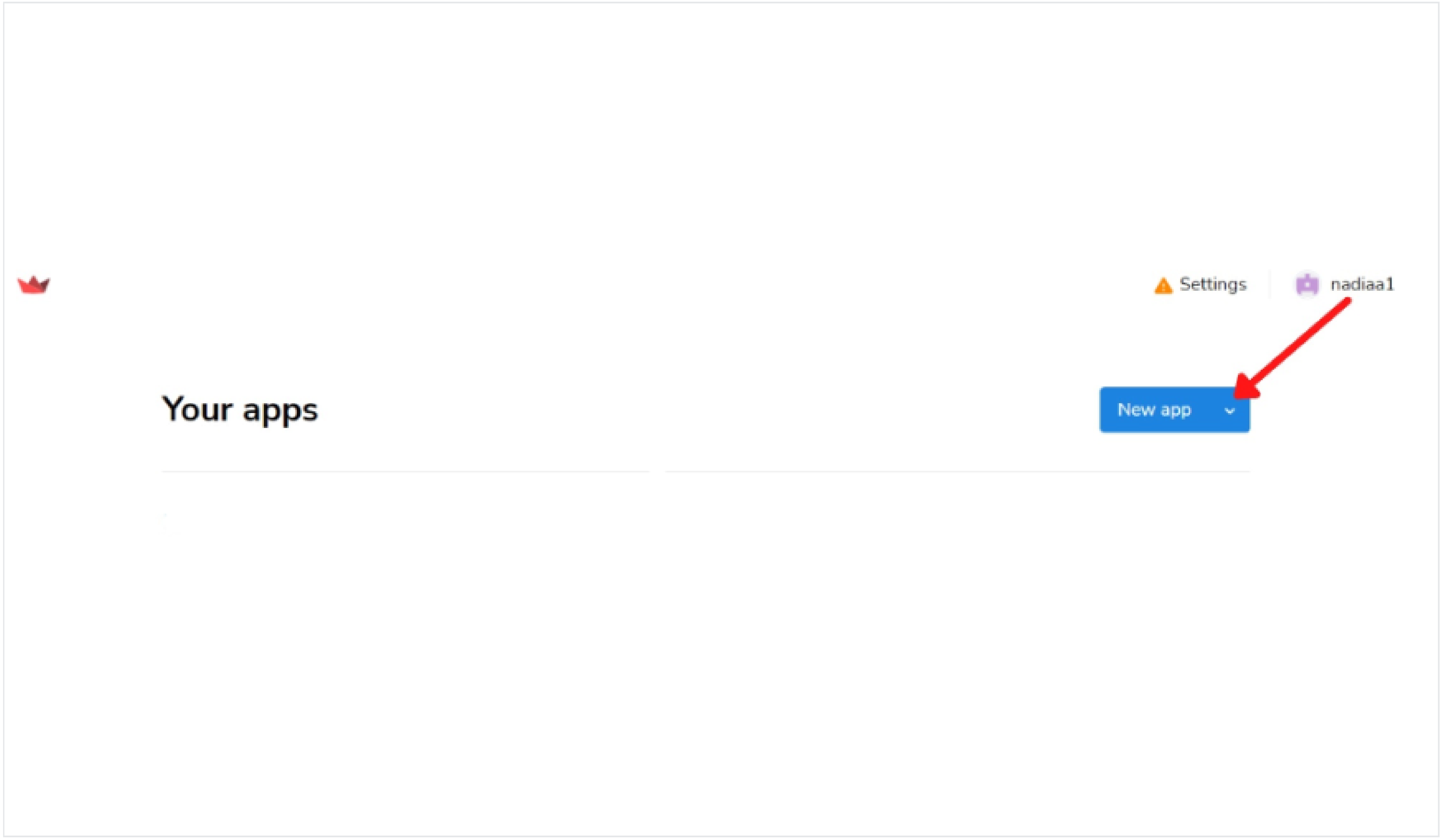
  - pip install freeze

  - pip freeze > requirements.txt (not recommended)
  - pip freeze | grep -i panda >> requirements.txt (better) [works only in Lunix]

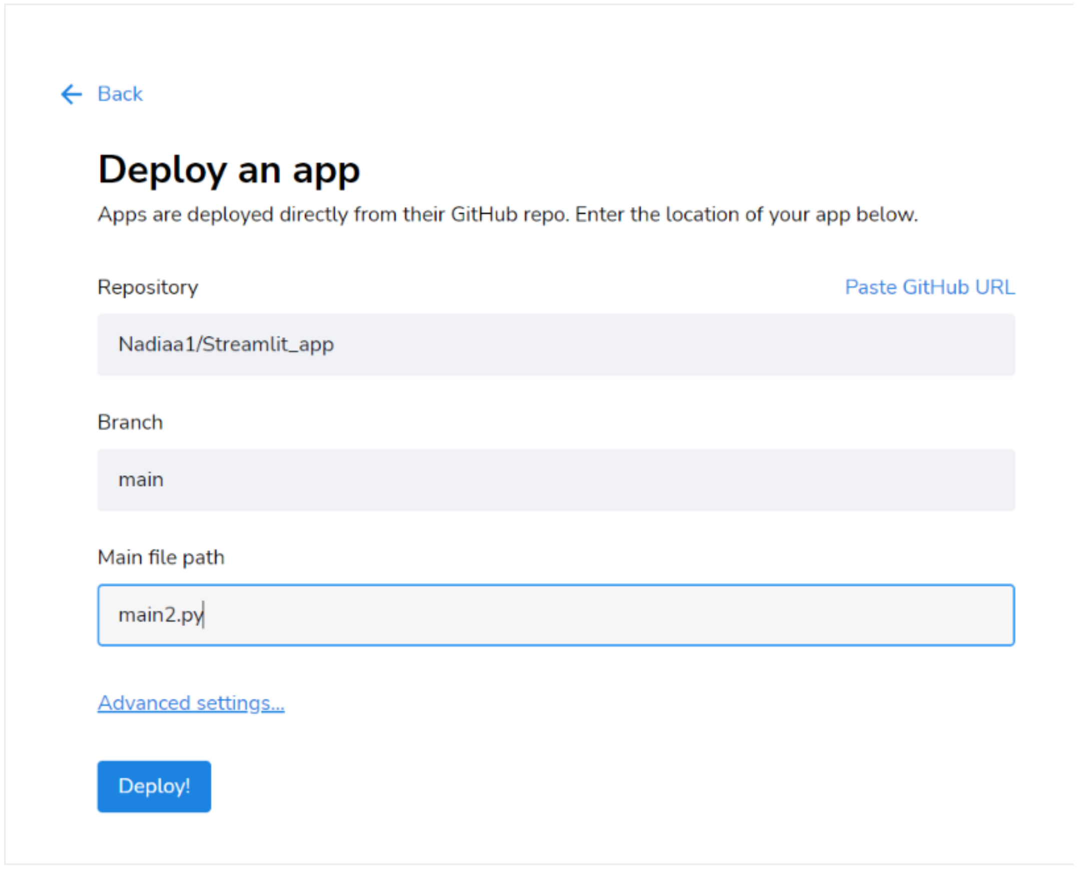
  - pip install pipreqs

  - pipreqs
  - pipreqs [Path]

Deploying your app



Deploying your app



← Back

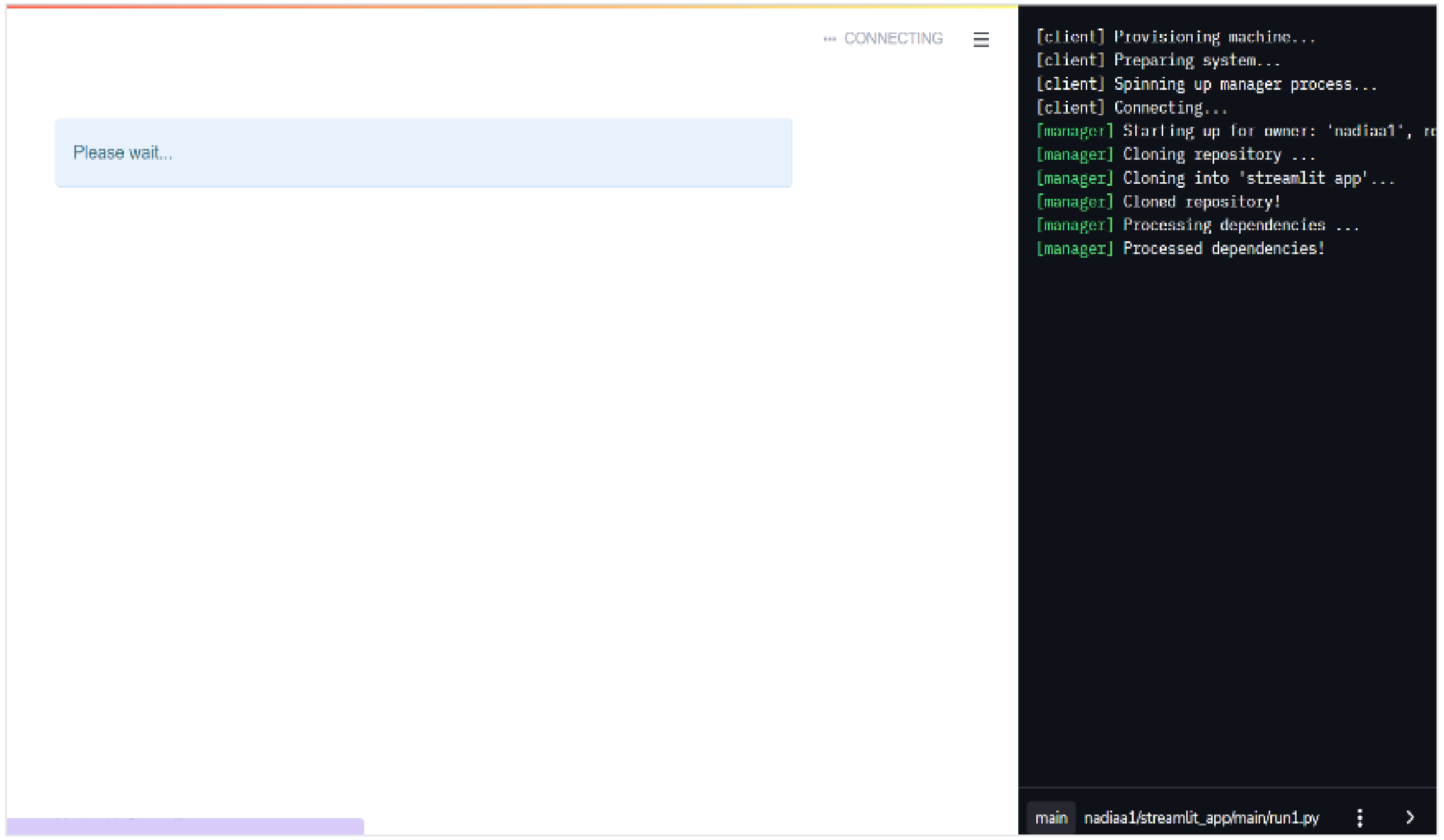
Deploy an app

Apps are deployed directly from their GitHub repo. Enter the location of your app below.

Repository: Nadiaa1/Streamlit_app

Branch: main

Main file path: main2.py

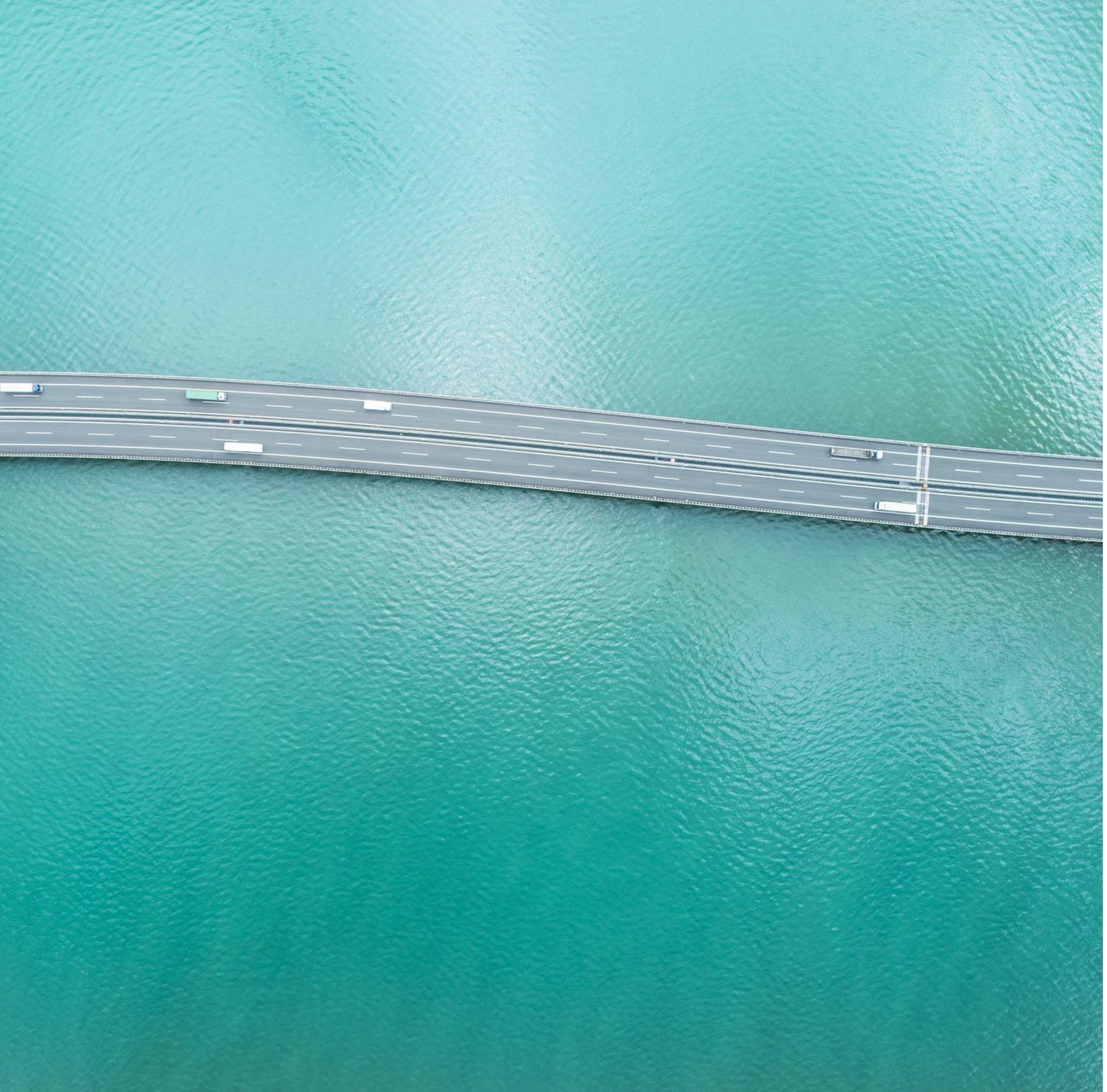


Please wait...

```
[client] Provisioning machine...
[client] Preparing system...
[client] Spinning up manager process...
[client] Connecting...
[manager] Starting up for owner: 'nadiaa1', re
[manager] Cloning repository ...
[manager] Cloning into 'streamlit app'...
[manager] Cloned repository!
[manager] Processing dependencies ...
[manager] Processed dependencies!
```

Links

- Github repository for the ML model: https://github.com/YonSci/ML_classifier
- ML Model deployed on Streamlit Cloud: https://mlclassifier-jntfcz3ekfn8jf6czjvdxu.streamlit.app/
- Resources/Materials: https://github.com/YonSci/Streamlit_Resource
- Email: yonas.mersha14@gmail.com



Thank You

&

Happy web app deployment