1 Hour Journey with a first WebApp with Streamlit

# Version Information

Satya Komatineni, 1/25/24

Version 1

Version 2 is expected in the next month or so.

Contents

[Version Information 1](#_Toc157100103)

[Goal: A quick python webapp 1](#_Toc157100104)

[Prerequisites 2](#_Toc157100105)

[First things first 2](#_Toc157100106)

[How it works at its core 2](#_Toc157100107)

[Here is what happens with Streamlit: Next Level 2](#_Toc157100108)

[Your first webapp 2](#_Toc157100109)

[How long it took me: 1 hour for a basic page. 3](#_Toc157100110)

[The English Instructions file in the repo 3](#_Toc157100111)

[What ChatGPT gave me 4](#_Toc157100112)

[Limitations 5](#_Toc157100113)

[In the Next Version of the document 5](#_Toc157100114)

[Key References 5](#_Toc157100115)

[Credits 5](#_Toc157100116)

# Goal: A quick python webapp

I have been (for a week) experimenting with Hugging face and LangChain for utilizing LLMs.

One need is to expose the results as a web page.

Streamlit I was introduced as a quick tool to do this.

I have spent an hour to do this. I was worried about how long this initial foray would be as already swamped with a plethora of tools.

Hope this document helps you, experienced (web) programmer, as well, to quickly get there.

Good luck.

# Prerequisites

1. Python
2. Vscode
3. A need for creating a simple webapp with python as the backend

# First things first

The website is at: <https://docs.streamlit.io/>

The installation is here: <https://docs.streamlit.io/get-started/installation>

Fastest command line is here: <https://docs.streamlit.io/get-started/installation/command-line>

Note:

1. I have basic python with vscode. I don’t have anaconda distribution.
2. I don’t have virtual environments (Another story, why I don’t)
3. Assumption: You know how to work with Vscode, pip install etc.

# How it works at its core

1. You will have a regular py file like hello.py.
2. Then you run: Streamlit run hello.py.
3. This will open a webserver and browser on your local box.
4. That’s it.

# Here is what happens with Streamlit: Next Level

1. It is just a python lib in your command line apps.
2. pip install it, just another lib (like all other python stuff the dependencies galore)
3. You are ready to go.
4. If you run Streamlit on a command line, it creates a webapp on your local box.
5. It uses the python file you specify as its home page.
6. That .py file has write-statements in it like hello world.
7. That is your first webapp!

# Your first webapp

First

1. pip install streamlit

A simple python file as your home page

1. Hello.py

2. import streamlit as st

3. st.write(“Hello world”)

You run this as

1. streamlit run hello.py

# How long it took me: 1 hour for a basic page.

1. Went to their web page.
2. Read getting started.
3. Wrote what I want in English.
4. Gave it to ChatGPT. (will show you this spec in a minute)
5. I copied the code from ChatGPT.
6. A web form with its output back to the page is the requirements doc.

# The English Instructions file in the repo

I fed the following file to ChatGPT.

1. Filename: streamlit-ui-requirements.txt

2. A file to guide ChatGPT based development.

3. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

4. Requirement 1: for ChatGPT: Create a webpage with a from submission

5. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

6. What I want:

7.

8. 1. Create a function

9. 2. it should prompt the user on command line with a prompt

10. 3. Gather the prompt from user

11. 4. If the prompt is empty exit the program

12. 5. if the prompt is "q" or "quit" exactly exit the program

13. 6. call a function to process the prompt. Call this function "processPrompt"

14. 7. Put all this in a while loop

15. 8. Call the function "executePromptLoop"

16.

17. Assume:

18. 1. To write code modularly

# What ChatGPT gave me

I made very minor changes

1. import streamlit as st

2. #

3. #\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

4. # Key elements

5. # 1. A form with a submit

6. # 2. Process text input

7. # 3. Write back what is entered so far

8. # 4. Rememebers with each web refresh the state

9. #\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

10. #

11. def writeIntro():

13. st.title("Demo page")

17. st.write("hello there")

18.

19.

20. def initialize\_state():

21. writeIntro()

22.

23. """Initialize the state variable."""

24. if 'totalResponseText' not in st.session\_state:

25. st.session\_state.totalResponseText = []

26.

27. def writeOutput():

28. # Display the accumulated text

29. st.write("Accumulated Text:")

30. for line in st.session\_state.totalResponseText:

31. st.write(line)

32.

33.

34. def process\_input(input\_text):

35. """Append the input text to the totalResponseText state variable."""

36. st.session\_state.totalResponseText.append(input\_text)

37.

38. def main():

39. """Main function to render the Streamlit page."""

40. initialize\_state()

41.

42. # Create a form

43. with st.form(key='input\_form'):

44. text\_input = st.text\_input("Enter your text")

45. submit\_button = st.form\_submit\_button("Submit")

46.

47. # Process the form submission

48. if submit\_button and text\_input:

49. process\_input(text\_input)

50.

51. writeOutput()

52.

53. #Kick it off

54. main()

55.

# Limitations

1. I don’t know yet at all how limited Streamlit capabilities are in this space compared to full blown webapps.
2. I have not researched any other competing tools.

# In the Next Version of the document

1. Use it for a more comprehensive web application.
2. Use it with LangChain and HuggingFace LLMs.
3. Update this document.

# Key References

1. The website is at: <https://docs.streamlit.io/>
2. The installation is here: <https://docs.streamlit.io/get-started/installation>
3. Fastest command line is here: <https://docs.streamlit.io/get-started/installation/command-line>

# Credits

1. ChatGPT
2. Streamlet Team