

## EASS שיעור 11- עבודה עם GITHUB, המשך STREAMLIT

### CMD VS ENTRYPOINT

הערות לעבודה עם github:

1. קבצים שלא נרצה שיוצגו בגיטהאב יש לשים בקובץ gitignore (כמו .DS\_Store , תיקיית .vscode , cache וכו').  
כדי לכלול בקובץ את כל הקבצים שיש להם סיומת זהה ניתן להשתמש בכוכבית, לדוגמה: \*.cache
- ניצור קובץ gitignore. ונרשום בו בכל שורה את השמות של הקבצים/תיקיות המיותרים.  
ניתן לחפש באינטרנט תבניות מוכנות template של קבצי gitignore להשראה.  
קובץ readme איכותי. קישור להסברים והדגמות:
2. <https://docs.github.com/en/get-started/writing-on-github/getting-started-with-writing-and-formatting-on-github/basic-writing-and-formatting-syntax>
3. בכל העלאת קבצים לגיטהאב ניצור commit לכל שינוי עם שם משמעותי.
4. עבור כל מטלה/פרוייקט ניצור repository נפרד.
5. ישנה תוכנה בשם github desktop שבה ניתן לערוך ולהעתיק repositories.
6. עבודה עם branch: ניצור branch שבו נעלה שינויים, כשנאחד את השינויים עם הbranch הראשי נמחק את הbranch המשני. מצב שבו לא נמחק branches זה כשיש גרסאות שונות אז לכל גרסה יהיה branch.

- בפקודת write ניתן ליצור ולהדפיס טבלה בעזרת pandas dataframe

```
import streamlit as st
st.title("Hello, Streamlit!")
st.write("Here's our first attempt at using Streamlit")

st.markdown(
    """
    # Streamlit is **awesome**.

    That's why we use it for all our projects.

    $ \mathbf{F} = m \mathbf{a}$

    """)
st.latex(r'''
    a + ar + a r^2 + a r^3 + \cdots + a r^{n-1} =
    \sum_{k=0}^{n-1} ar^k =
    a \left(\frac{1-r^n}{1-r}\right)
    ''')

import pandas as pd

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]
}))
```

← → ↺ ⌂ 🌐 localhost:2222 🔍 📄

$k=0$

Here's our first attempt at using data to create a table:

	first column	second column
0	1	10
1	2	20
2	3	30
3	4	40

- שימוש ב-widget כדי לקבל קלט מהמשתמש. דוגמה:

```
import streamlit as st
x = st.slider('x') # this is the widget
st.write(x, 'squared is', x * x)
```

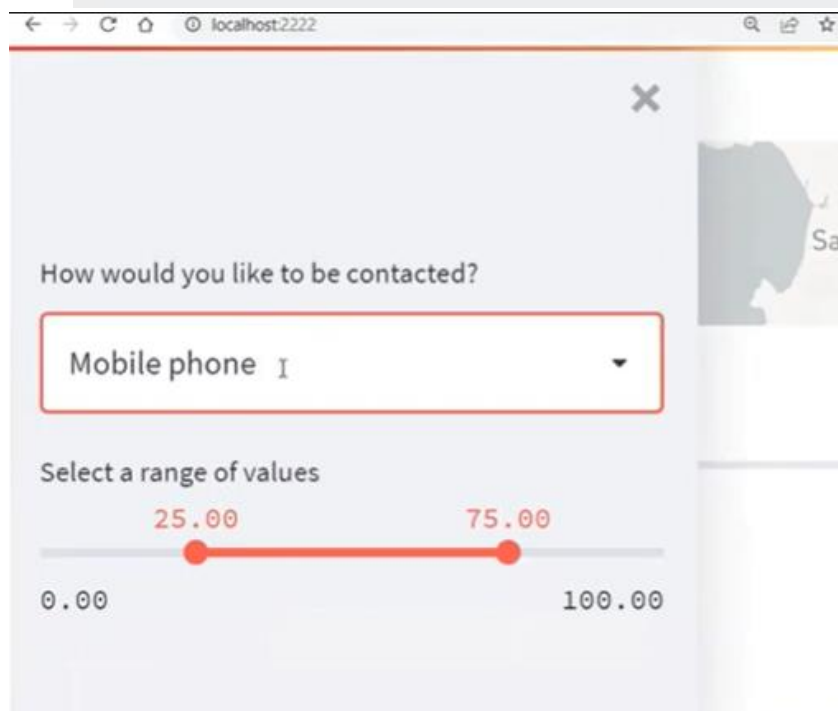


- Layout – הוספת אלמנטים לפריסת המסך כמו תפריט צד, תא שניתן להרחיב ולצמצם אותו. קישור לדוגמאות נוספות: <https://docs.streamlit.io/library/api-reference/layout>

```
import streamlit as st

# Add a selectbox to the sidebar:
add_selectbox = st.sidebar.selectbox(
    'How would you like to be contacted?',
    ('Email', 'Home phone', 'Mobile phone')
)

# Add a slider to the sidebar:
add_slider = st.sidebar.slider(
    'Select a range of values',
    0.0, 100.0, (25.0, 75.0)
)
```

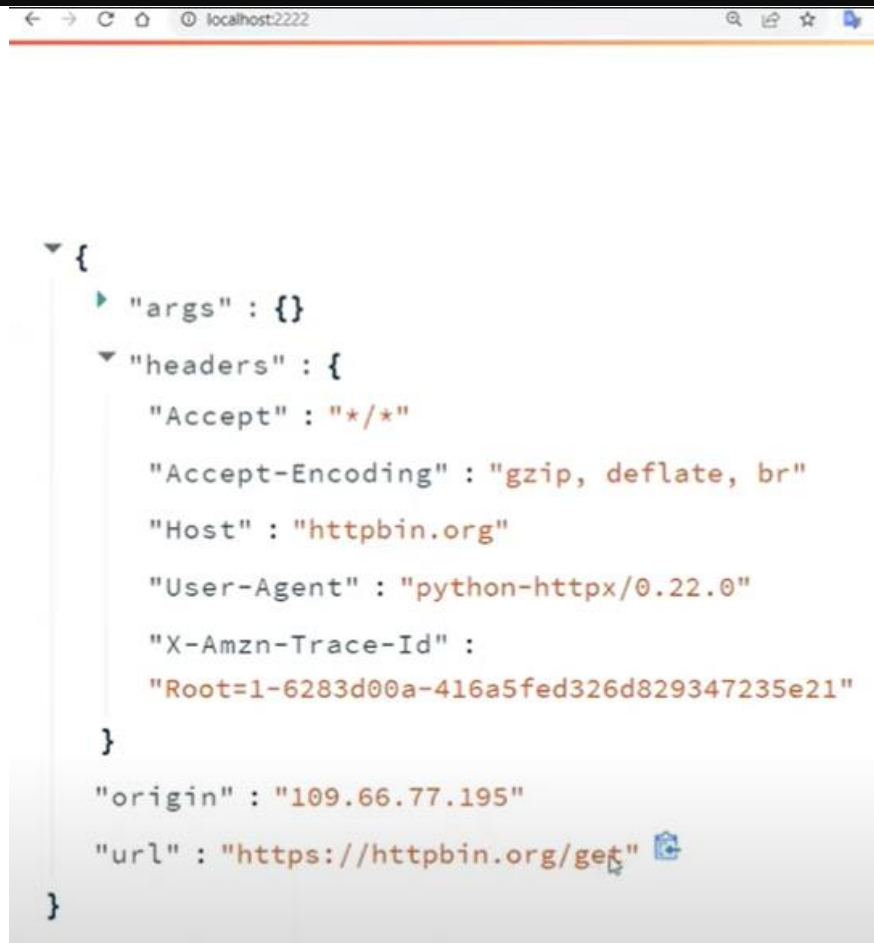


- תקשורת עם ה-backend : נשתמש בבקשות HTTP בעזרת ספריית httpx

```
import streamlit as st
import httpx

backend_endpoint = "https://httpbin.org/get"
r = httpx.get(backend_endpoint)

st.json(r.json())
```



```
{
  "args": {},
  "headers": {
    "Accept": "*/*",
    "Accept-Encoding": "gzip, deflate, br",
    "Host": "httpbin.org",
    "User-Agent": "python-httpx/0.22.0",
    "X-Amzn-Trace-Id": "Root=1-6283d00a-416a5fed326d829347235e21"
  },
  "origin": "109.66.77.195",
  "url": "https://httpbin.org/get"
}
```

- שימוש ב-Streamlit באמצעות Dockerfile:

```
FROM python:3.8
RUN pip install streamlit
COPY . /app
WORKDIR /app
ENTRYPOINT ["streamlit", "run", "app.py"]
```

הבדל בין ENTRYPOINT ל-CMD:

שתי הפקודות הללו מגדירות מה ירוץ כשהקונטינר יתחיל.

ההבדל ביניהן הוא ש-ENTRYPOINT נותן פרמטרים שלא "נדרסים" אוטומטית (אלא "נדרסים" רק אם נשים --entrypoint עם פרמטר חדש בשורת ה-run), לעומת זאת CMD נדרס אוטומטית אם נשים פרמטר (ללא צורך בקידומת).

```
from ubuntu:latest
RUN apt-get update
RUN apt-get install -y iputils-ping

ENTRYPOINT ["/bin/ping"]
CMD ["localhost"]
```

נוסיף את פרמטר google.com שיחליף את localhost אוטומטית:

```
→ html-demo docker run -it aaa google.com
PING google.com (142.251.37.174) 56(84) bytes of data.
64 bytes from 142.251.37.174 (142.251.37.174): icmp_seq=1 ttl=37 time
=43.2 ms
64 bytes from 142.251.37.174 (142.251.37.174): icmp_seq=2 ttl=37 time
=43.6 ms
```

נוסיף את קידומת --entrypoint עם הפרמטר החדש שלה כדי לדרוס את /bin/ping :

```
→ html-demo docker run --entrypoint bash -it aaa
root@4d0d6d9e9f2f:/# exit
exit
```