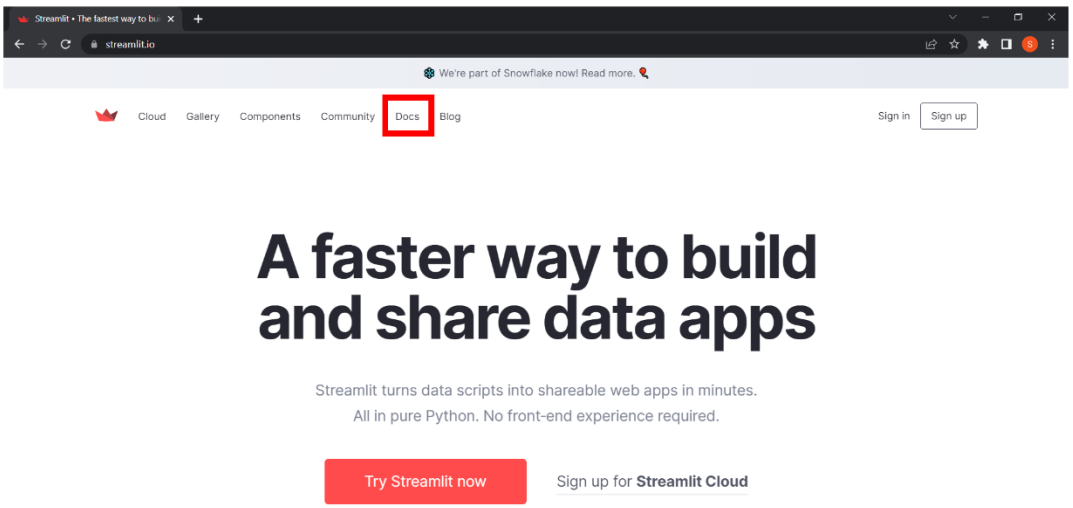
ไลบรารี Streamlit

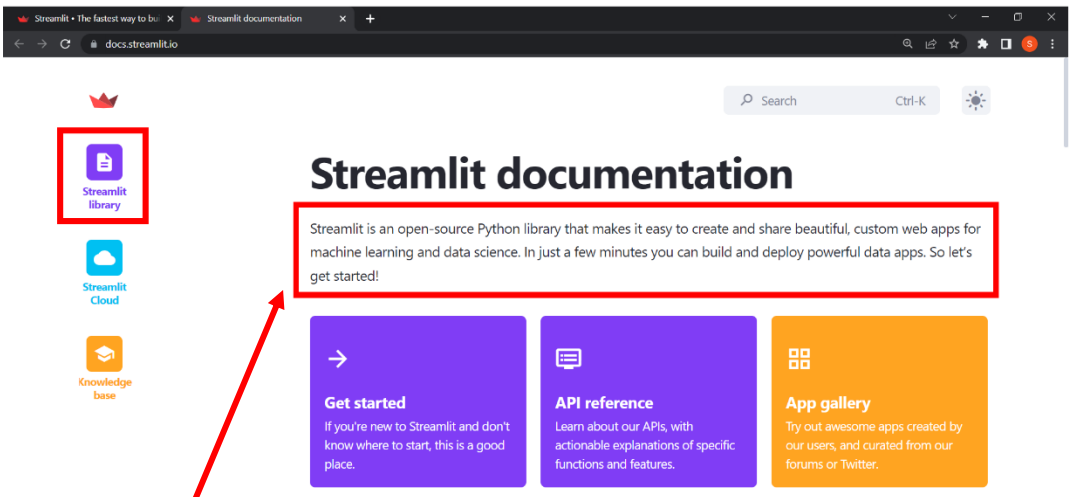
เว็บไซต์ https://streamlit.io
เมื่อเข้ามาที่หน้าเว็บไซต์ ให้กดที่เมนู Docs



Streamlit เป็นเครื่องมือสำหรับสร้างเว็บแอปพลิเคชัน (web applications) ในภาษา Python อย่างง่าย โดยมีการสร้างอินเทอร์เฟซผู้ใช้แบบสตรีมไลน์

Streamlit

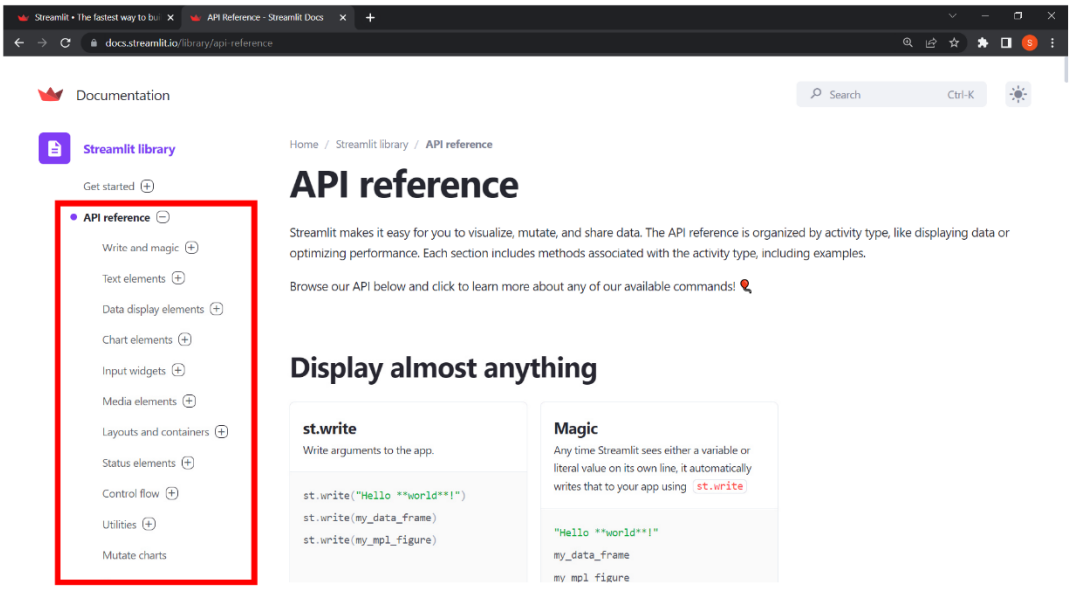
ที่หน้า Streamlit Documentation ให้กดที่เมนู Streamlit library



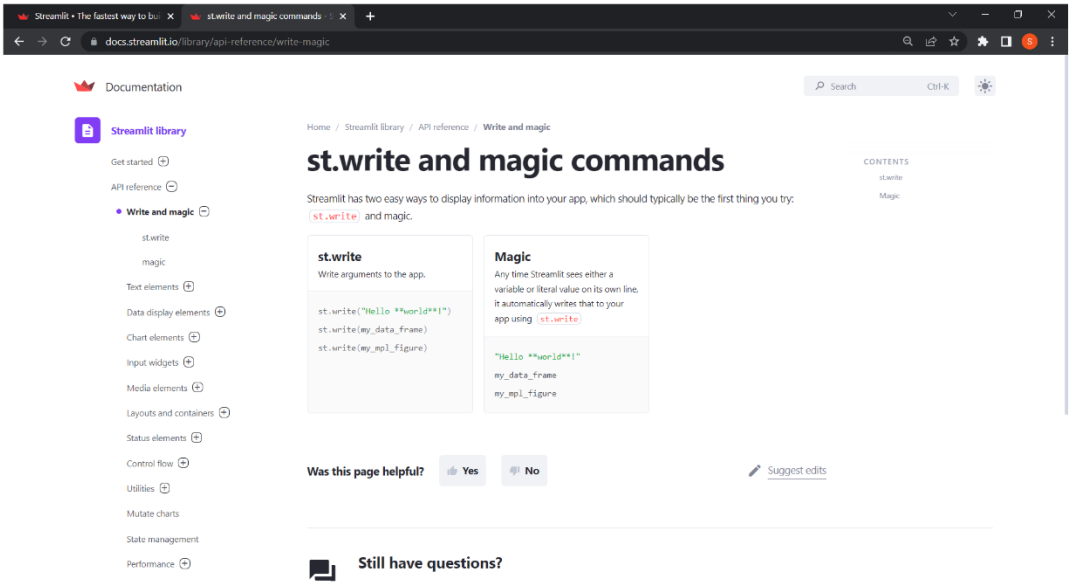
Streamlit เป็นไลบรารี Python โอเพนซอร์สที่ทำให้ง่ายต่อการสร้างและแบ่งปันเว็บแอปพลิเคชันที่สวยงามและที่ปรับแต่งได้ตามต้องการ

Streamlit

ที่ API reference แสดงคำสั่งต่าง ๆ สำหรับใช้ในการสร้าง Web Application

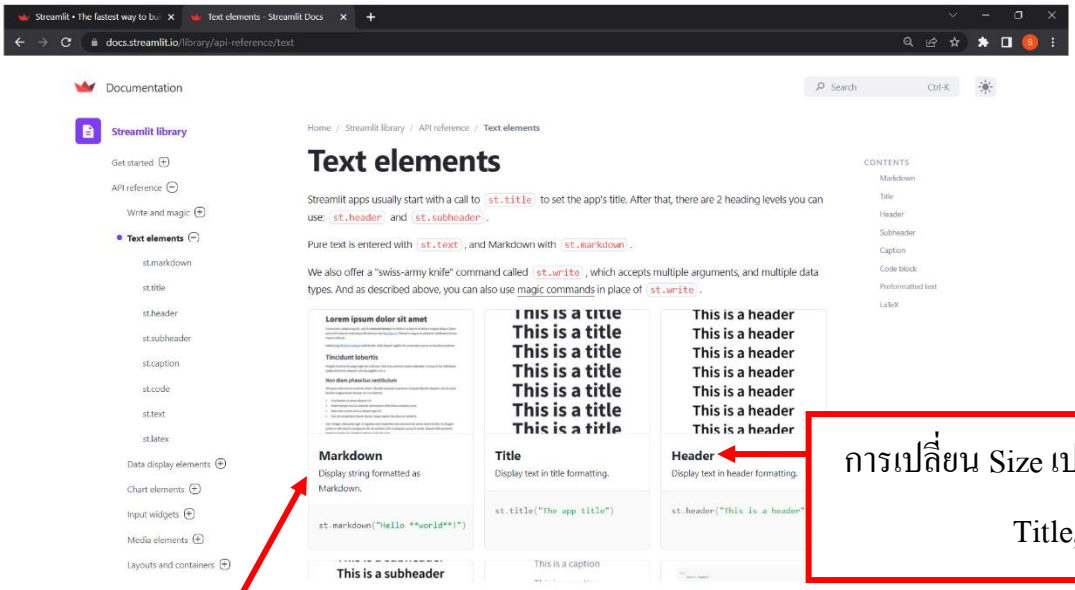


Write and magic



การแสดงผลข้อความเป็น Streamlit.write ก็จะเป็นการนำค่าจากตัวแปรขึ้นจอที่หน้าเว็บไซต์

Text elements

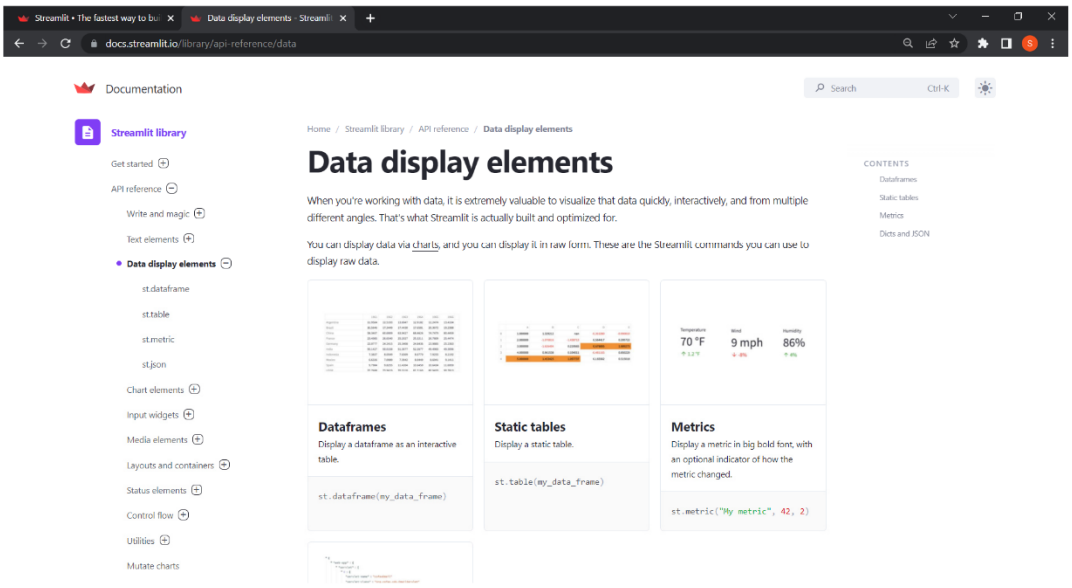


การเปลี่ยน Size เปลี่ยนขนาดตัวหนังสือ

Title, Header

การเปลี่ยนสี, เปลี่ยนการจัดการหน้า ชิดซ้าย ชิดขวา อยู่ตรงกลางจอ จะใช้คำสั่งที่ชื่อ **(Markdown)**

Data display elements



การแสดงผลหน้าจอ: ก็จะมีการแสดงผลแบบตาราง แสดงผลค่าสถิติ แสดงผลปบ Metrics

Chart elements

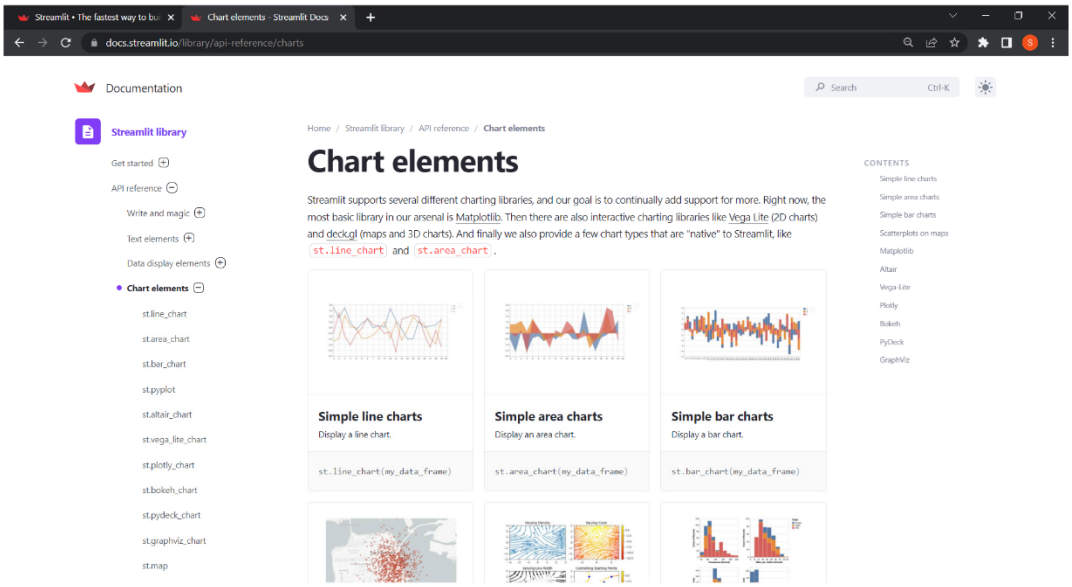
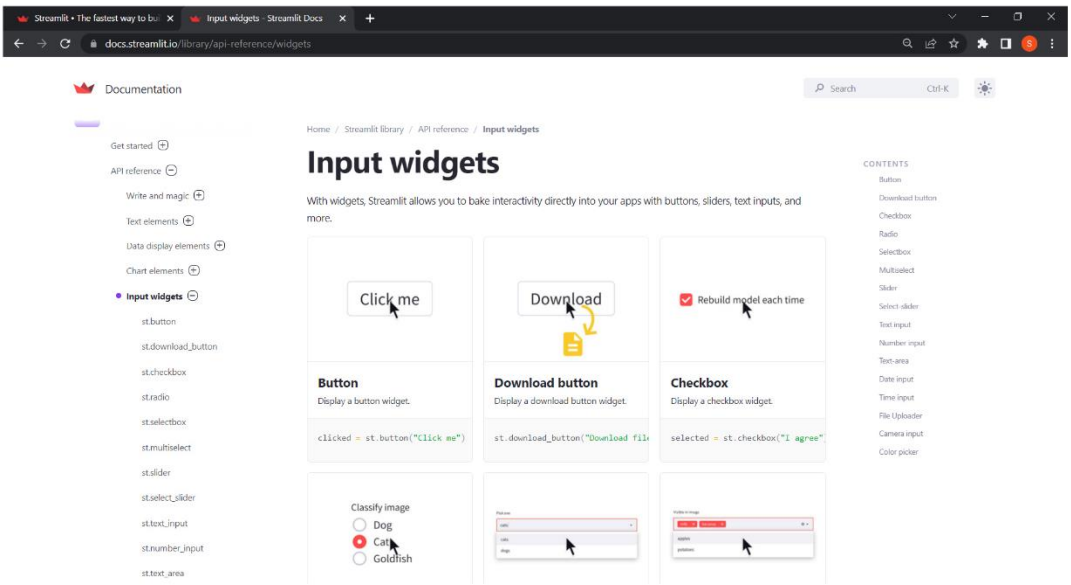


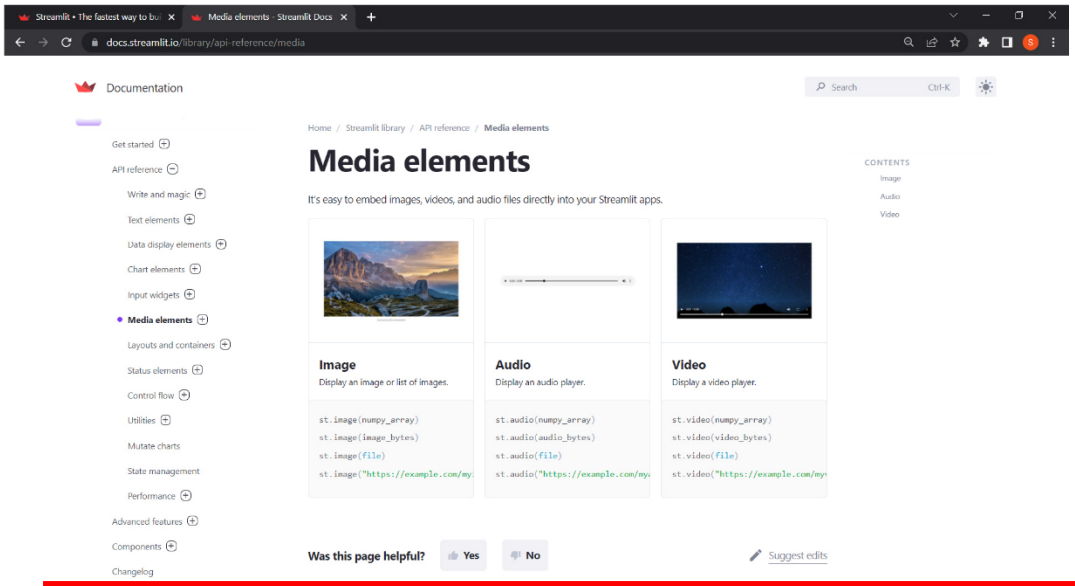
Chart สามารถ Port Graph ได้หลายกราฟ หลายแบบ เราสามารถ Port Graph ที่หน้าเว็บไซต์ได้หลายแบบ

Input widgets



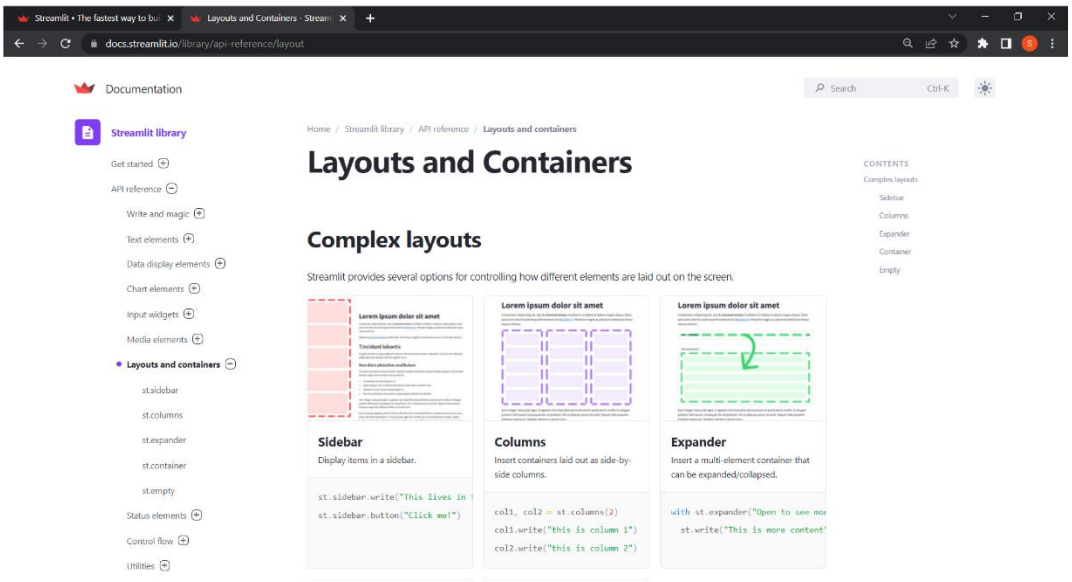
Input = ก็จะมีการรับค่าจากการกดปุ่ม, Checkbox (ช่องทำเครื่องหมาย), radio button (ปุ่มตัวเลือก), Read box (กล่องอ่าน), Multi List box (กล่องรายการหลายรายการ)

Media elements



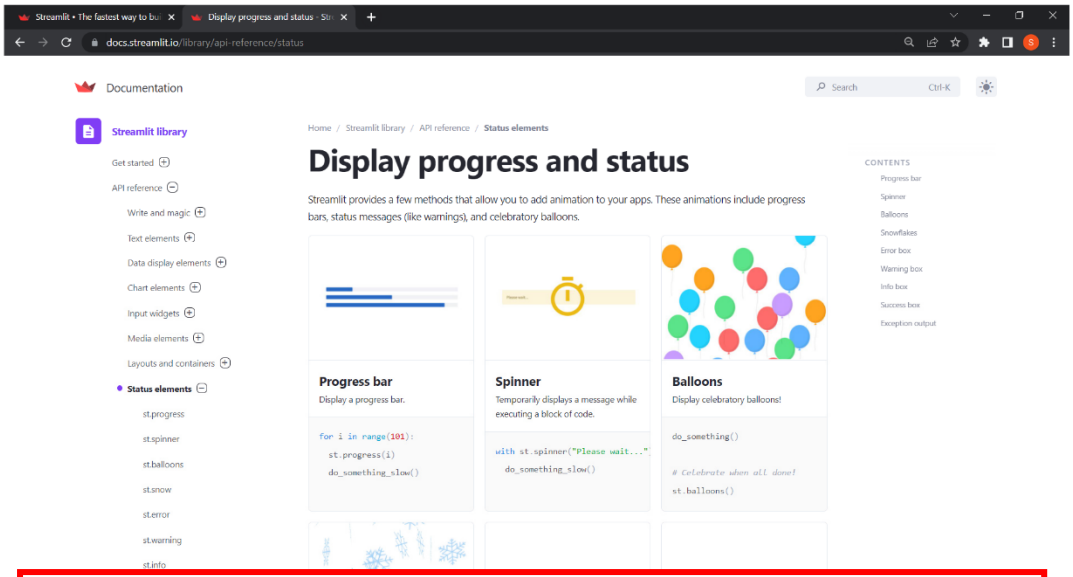
จัดการเกี่ยวกับภาพที่เราจะใช้กัน คือ **Image** รวมถึง Audio และ Video ด้วยในการประมวลผลเสร็จแล้วไปขึ้นหน้าจอ

Layouts and Containers



การจัดการคอลัมน์ = การจัดนำข้อมูลมาจัดเป็นคอลัมน์ 3 คอลัมน์ 4คอลัมน์ ก็ทำได้

Display progress and status

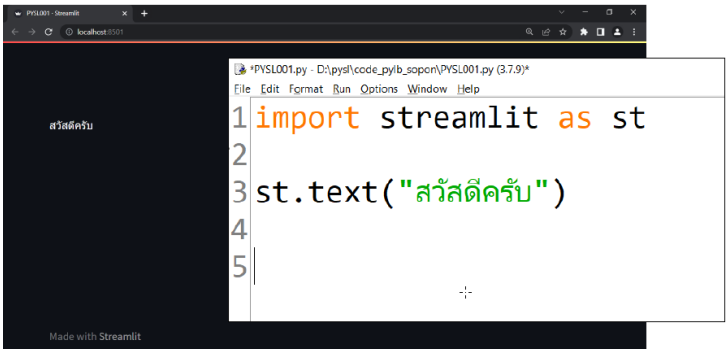


Progress กรณีที่เราประมวลผลอยู่ก็จะมีแถบเลื่อนให้ หรือจะโชว์เป็นลักษณะนาฬิกา หรือจะแสดงความยินดีเป็นลูกโป่ง หรือบอลลูน

การเรียกใช้ฟังก์ชัน และคำสั่งต่างๆ ของ Library Streamlit

ตัวอย่างที่ 1 การแสดงข้อความ ด้วย st.text (PYSL001.py)

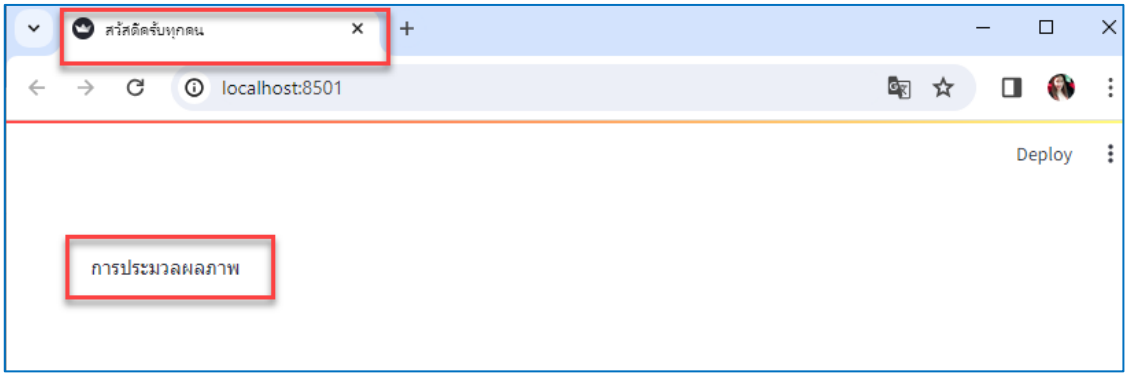
```
PYSL001.py
1   import streamlit as st
2
3   st.text("สวัสดีครับ/สวัสดีค่ะ")
```



- คำสั่งนี้ใช้ฟังก์ชัน text ของ Streamlit เพื่อแสดงข้อความธรรมดาบนหน้าเว็บแอปพลิเคชัน
- import streamlit as st : การนำเข้าของไลบรารี Streamlit และให้เรียกใช้งานฟังก์ชันต่างๆ ภายใต้ชื่อ st.

ตัวอย่างที่ 2 การกำหนด title ของหน้าเว็บเพจ

```
PYSL002.py
1   import streamlit as st
2
3   st.set_page_config(page_title="สวัสดีครับทุกคน")
4
5   st.text("การประมวลผลภาพ")
6
```

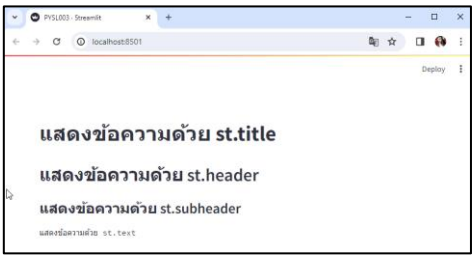


- ฟังก์ชัน set_page_config กำหนดการตั้งค่าชื่อเพจหรือหัวข้อหน้าเว็บในแท็บของเบราว์เซอร์

ตัวอย่างที่ 3 การแสดงข้อความ กำหนดขนาดข้อความ) ด้วย st.title , st.header และ st.subheader

PYSL003.py

```python
import streamlit as st

st.title("แสดงข้อความด้วย st.title")
st.header("แสดงข้อความด้วย st.header")
st.subheader("แสดงข้อความด้วย st.subheader")
st.text("แสดงข้อความด้วย st.text")
```

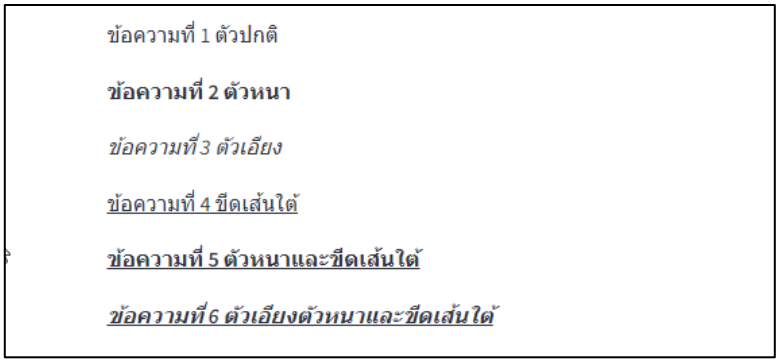


- หัวข้อใหญ่ / หัวข้อรอง / หัวข้อย่อย / ข้อความธรรมดา

ตัวอย่างที่ 4 การทำตัวอักษรให้เป็น ตัวหนา ตัวเอียง และขีดเส้นใต้ ด้วย st.markdown

PYSL007.py > ...

```python
import streamlit as st

text1 = "ข้อความที่ 1 ตัวปกติ"
st.markdown(f'<p>{text1}</p>', unsafe_allow_html=True)

text2 = "ข้อความที่ 2 ตัวหนา"
st.markdown(f'<b>{text2}</b>', unsafe_allow_html=True)

text3 = "ข้อความที่ 3 ตัวเอียง"
st.markdown(f'<i>{text3}</i>', unsafe_allow_html=True)

text4 = "ข้อความที่ 4 ขีดเส้นใต้"
st.markdown(f'<u>{text4}</u>', unsafe_allow_html=True)

text5 = "ข้อความที่ 5 ตัวหนาและขีดเส้นใต้"
st.markdown(f'<u><b>{text5}</b></u>', unsafe_allow_html=True)

text6 = "ข้อความที่ 6 ตัวเอียงตัวหนาและขีดเส้นใต้"
st.markdown(f'<u><b><i>{text6}</i></b></u>', unsafe_allow_html=True)
```



❖ import streamlit as st : การนำเข้าของไลบรารี Streamlit และให้เรียกใช้งานฟังก์ชันต่างๆ ภายใต้ชื่อ st.

f คือ f-string หรือ "Formatted String Literals" ซึ่งเป็นวิธีการจัดรูปแบบสตริงใน Python

❖ ฟังก์ชัน st.markdown คือ ใช้เพื่อแสดงข้อความที่มีการกำหนดรูปแบบด้วย HTML/CSS

unsafe_allow_html=True อนุญาตให้ใช้ HTML ภายใน st.markdown

ข้อความที่ 1: แสดง "ข้อความที่ 1 ตัวปกติ" ในรูปแบบปกติ.

ข้อความที่ 2: แสดง "ข้อความที่ 2 ตัวหนา" ในรูปแบบตัวหนา. HTML ... ใช้สำหรับตัวหนา.

ข้อความที่ 3: แสดง "ข้อความที่ 3 ตัวเอียง" ในรูปแบบตัวเอียง. HTML ... ใช้สำหรับตัวเอียง.

ข้อความที่ 4: แสดง "ข้อความที่ 4 ขีดเส้นใต้" ในรูปแบบขีดเส้นใต้. HTML <u>...</u> ใช้สำหรับขีดเส้นใต้.

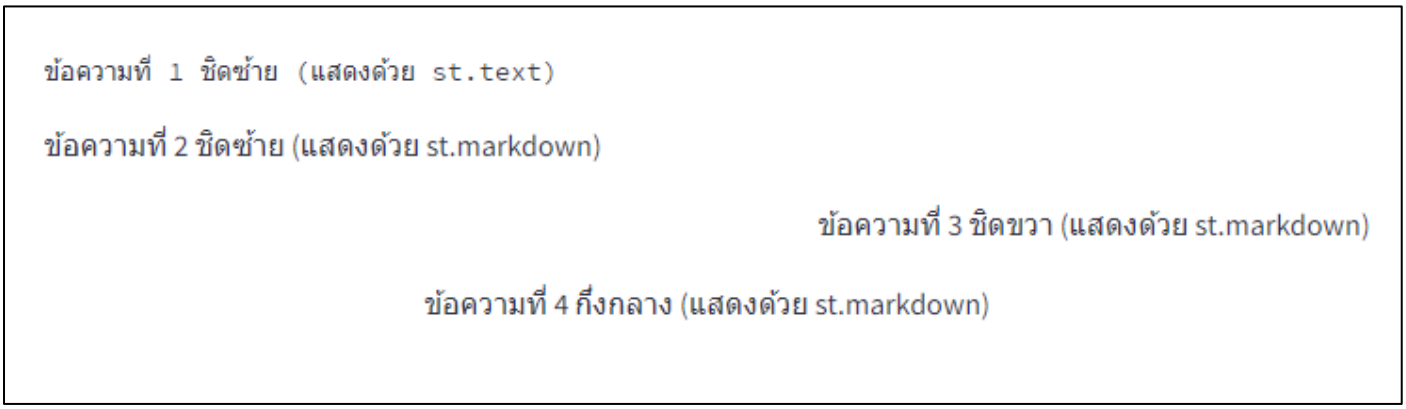
ข้อความที่ 5: แสดง "ข้อความที่ 5 ตัวหนาและขีดเส้นใต้" ในรูปแบบทั้งตัวหนาและขีดเส้นใต้.

ข้อความที่ 6: แสดง "ข้อความที่ 6 ตัวเอียงตัวหนาและขีดเส้นใต้" ในรูปแบบที่รวมตัวเอียง, ตัวหนา และขีดเส้นใต้.

โค้ดนี้ใช้ HTML เพื่อกำหนดรูปแบบของข้อความที่แสดงบนเว็บ. มันช่วยให้สามารถควบคุมรูปแบบข้อความได้มากกว่าที่ st.text ทำได้.

<u>ตัวอย่างที่ 5</u> การทำให้ตัวอักษรอยู่ชิดซ้าย ชิดขวา และกึ่งกลาง ด้วย st.markdown แสดงข้อความที่มีการกำหนดรูปแบบด้วย HTML/CSS

```python
import streamlit as st

text1 = "ข้อความที่ 1 ชิดซ้าย (แสดงด้วย st.text)"
st.text(text1)

text2 = "ข้อความที่ 2 ชิดซ้าย (แสดงด้วย st.markdown)"
st.markdown(f'<p style="text-align:left">{text2}</p>', unsafe_allow_html=True)

text3 = "ข้อความที่ 3 ชิดขวา (แสดงด้วย st.markdown)"
st.markdown(f'<p style="text-align:right">{text3}</p>', unsafe_allow_html=True)

text4 = "ข้อความที่ 4 กึ่งกลาง (แสดงด้วย st.markdown)"
st.markdown(f'<p style="text-align:center">{text4}</p>', unsafe_allow_html=True)
```



- import streamlit as st : การนำเข้าของไลบรารี Streamlit และให้เรียกใช้งานฟังก์ชันต่างๆ ภายใต้ชื่อ st.
- ใช้ฟังก์ชัน text เพื่อแสดง text1 บนหน้าเว็บ st.text มักจะแสดงข้อความโดยการชิดซ้ายตามค่าเริ่มต้น

แสดงข้อความชิดซ้ายด้วย st.text:

- text1 = "ข้อความที่ 1 ชิดซ้าย (แสดงด้วย st.text)": st.text จะแสดงข้อความในรูปแบบชิดซ้ายเป็น ฟังก์ชัน text เพื่อแสดง text1 บนหน้าเว็บ st.text มักจะแสดงข้อความโดยการชิดซ้ายตามค่าเริ่มต้น

แสดงข้อความชิดซ้ายด้วย HTML ใน st.markdown:

- text2 = "ข้อความที่ 2 ชิดซ้าย (แสดงด้วย st.markdown)": กำหนดข้อความให้กับตัวแปร text2.
- st.markdown(f' {text2} ', unsafe_allow_html=True): แสดงข้อความ text2 ในรูปแบบ HTML โดยใช้ฟังก์ชัน st.markdown. unsafe_allow_html=True อนุญาตให้ใช้ HTML ใน Markdown.

แสดงข้อความชิดขวาด้วย HTML ใน st.markdown:

- text3 = "ข้อความที่ 3 ชิดขวา (แสดงด้วย st.markdown)": กำหนดข้อความให้กับตัวแปร text3.
- st.markdown(f' {text3} ', unsafe_allow_html=True): แสดงข้อความ text3 ในรูปแบบ HTML โดยใช้ st.markdown และตั้งค่าการจัดตำแหน่งข้อความเป็นชิดขวา.

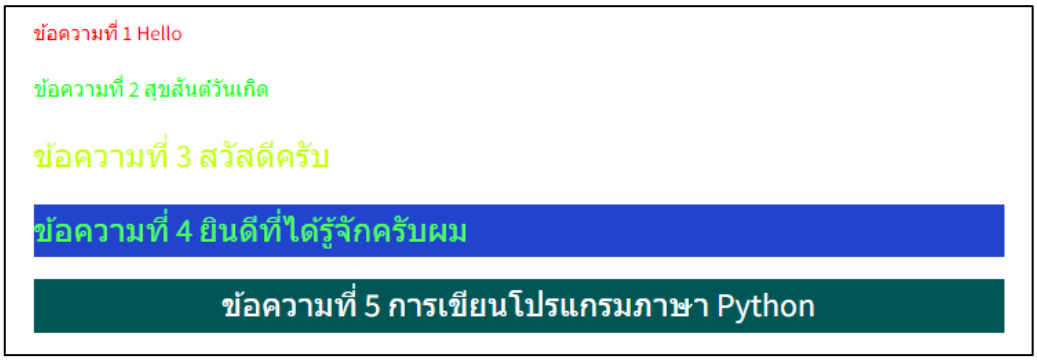
แสดงข้อความกึ่งกลางด้วย HTML ใน st.markdown:

- text4 = "ข้อความที่ 4 กึ่งกลาง (แสดงด้วย st.markdown)": กำหนดข้อความให้กับตัวแปร text4.
- st.markdown(f' {text4} ', unsafe_allow_html=True): แสดงข้อความ text4 ในรูปแบบ HTML โดยใช้ st.markdown และตั้งค่าการจัดตำแหน่งข้อความเป็นกึ่งกลาง.
- style="text-align ช่วยกำหนดการจัดเรียงข้อความภายในแท็ก
- unsafe_allow_html=True: อนุญาตให้ใช้ HTML ในฟังก์ชัน st.markdown

<u>ตัวอย่างที่ 6</u> การเปลี่ยนสีและขนาดของตัวอักษร ด้วย st.markdown (ใช้เพื่อแสดงข้อความที่มีการกำหนดรูปแบบด้วย HTML/CSS)

PYSL009.py > ...

```python
import streamlit as st

text1 = "ข้อความที่ 1 Hello"
st.markdown(f'<p style="color:#ff0000;font-size:15px;">{text1}</p>', unsafe_allow_html=True)

text2 = "ข้อความที่ 2 สุขสันต์วันเกิด"
st.markdown(f'<p style="color:#00ff00;font-size:100%;">{text2}</p>', unsafe_allow_html=True)

text3 = "ข้อความที่ 3 สวัสดีครับ"
st.markdown(f'<p style="color:#ccff00;font-size:150%;">{text3}</p>', unsafe_allow_html=True)

text4 = "ข้อความที่ 4 ยินดีที่ได้รู้จักครับผม"
st.markdown(f'<p style="background-color:#2244cc;color:#55ff55;font-size:24px;">{text4}</p>', unsafe_allow_html=True)

text5 = "ข้อความที่ 5 การเขียนโปรแกรมภาษา Python"
st.markdown(f'<p style="background-color:#005555;color:#ffffff;font-size:150%;text-align:center">{text5}</p>', unsafe_allow_html=True)

```



- import streamlit as st : การนำเข้าของไลบรารี Streamlit และให้เรียกใช้งานฟังก์ชันต่างๆ ภายใต้ชื่อ st.
- color, font-size, background-color, และ text-align ใน style ช่วยให้สามารถกำหนดสี, ขนาดฟอนต์, สีพื้นหลัง, และการจัดวางข้อความได้ตามต้องการในรูปแบบของ HTML
- unsafe_allow_html=True: อนุญาตให้ใช้ HTML ในฟังก์ชัน st.markdown

ตัวอย่างที่ 7 การแสดงข้อความแบบ 2 ข้อความ จากค่าในตัวแปร

```python
import streamlit as st
from time import sleep

text_out1 = st.empty()
text_out2 = st.empty()

x = ["แอปเปิ้ล","มะละกอ","กล้วย","ส้ม"]
y = ["ราดหน้า","ข้าวผัด","ไข่เจียว","หมูทอด"]

for i in range(0,4): # i = 0 1 2 3
    text_out1.text("แสดงชื่อผลไม้ : " + x[i])
    text_out2.text("แสดงชื่ออาหาร : " + y[i])
    sleep(2)

st.text("แสดงข้อมูลครบถ้วนแล้ว")
```

```
แสดงชื่อผลไม้ : ส้ม

แสดงชื่ออาหาร : หมูทอด

แสดงข้อมูลครบถ้วนแล้ว
```

- โค้ดนี้ใช้ Streamlit และโมดูล time ของ Python เพื่อแสดงข้อความที่อัปเดตตามลำดับในหน้าเว็บแอป
- import streamlit as st: นำเข้าไลบรารี Streamlit. และให้เรียกใช้งานฟังก์ชันต่างๆ ภายใต้ชื่อ st.
- from time import sleep: นำเข้าฟังก์ชัน sleep จากโมดูล time
- text_out1 = st.empty(): สร้างอ็อบเจกต์ว่างที่จะใช้สำหรับแสดงข้อความที่ 1.
- text_out2 = st.empty(): สร้างอ็อบเจกต์ว่างที่จะใช้สำหรับแสดงข้อความที่ 2.
- x = ["แอปเปิ้ล","มะละกอ","กล้วย","ส้ม"] : สร้างรายการ x ที่ประกอบด้วยชื่อผลไม้.
- y = ["ราดหน้า","ข้าวผัด","ไข่เจียว","หมูทอด"] : สร้างรายการ y ที่ประกอบด้วยชื่ออาหาร.
- for i in range(0,4) : สร้างลูปที่ทำงานสี่รอบ, โดย i จะมีค่าตั้งแต่ 0 ถึง 3.
- text_out1.text("แสดงชื่อผลไม้ : " + x[i]): อัปเดตข้อความใน text_out1 ด้วยชื่อผลไม้จากรายการ x.
- text_out2.text("แสดงชื่ออาหาร : " + y[i]): อัปเดตข้อความใน text_out2 ด้วยชื่ออาหารจากรายการ y.

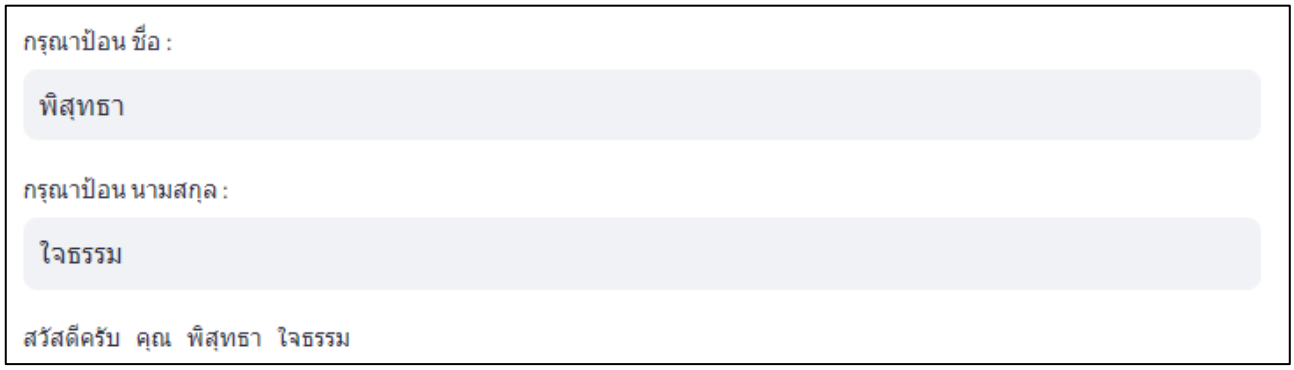
❖ sleep(2): หน่วงเวลา 2 วินาทีก่อนที่จะทำรอบถัดไปของลูป

แสดงข้อความท้ายสุด:

❖ st.text("แสดงข้อมูลครบถ้วนแล้ว")

<u>ตัวอย่างที่ 8</u> การรับข้อความอินพุตด้วย st.text_input

```
PYSL020.py > ...
1   import streamlit as st
2
3   #name = st.text_input("กรุณาป้อนชื่อ : ","พิมพ์ตรงนี้นะครับ")
4   name = st.text_input("กรุณาป้อน ชื่อ : ")
5   surname = st.text_input("กรุณาป้อน นามสกุล : ")
6   st.text("สวัสดีครับ คุณ " + name + ' ' + surname)
7
8
```



❖ import streamlit as st: นำเข้าไลบรารี Streamlit. และให้เรียกใช้งานฟังก์ชันต่างๆ ภายใต้ชื่อ st.

สร้างฟิลด์ป้อนข้อมูลต่างๆ:

❖ name = st.text_input("กรุณาป้อน ชื่อ : ") : สร้างฟิลด์ป้อนข้อมูล (text input) ที่ให้ผู้ใช้ป้อนชื่อ ค่าที่ผู้ใช้ป้อนจะถูกเก็บในตัวแปร name.

❖ surname = st.text_input("กรุณาป้อน นามสกุล : "): สร้างฟิลด์ป้อนข้อมูลอีกอันหนึ่งที่ให้ผู้ใช้ป้อนนามสกุล. ค่าที่ผู้ใช้ป้อนจะถูกเก็บในตัวแปร surname.

❖ st.text("สวัสดีครับ คุณ " + name + ' ' + surname): คำสั่งนี้ใช้เพื่อแสดงข้อความที่รวมชื่อและนามสกุลที่ผู้ใช้ป้อนไว้

*** โค้ดนี้มีประโยชน์ในการสร้างฟอร์มหรือแบบสอบถามง่ายๆ บนเว็บแอปพลิเคชัน Streamlit ซึ่งผู้ใช้สามารถป้อนข้อมูลและได้รับการตอบกลับแบบโต้ตอบบนหน้าเดียวกัน.

ตัวอย่างที่ 9 การส่งข้อความทาง LINE Notify

```python
import streamlit as st
import line_api

st.header("ส่งข้อความทาง LINE Notify")
mytoken = st.text_input("ป้อน LINE Token : ")
text_send = st.text_input("ป้อนข้อความที่ต้องการส่ง : ")

if st.button("ส่งข้อความ"):
    try:
        line_api.lineNotify(text_send,mytoken)
        st.text("ส่งข้อความของคุณแล้ว")
    except:
        st.text("ไม่สามารถส่งข้อความได้")
```

- import streamlit as st: นำเข้าไลบรารี Streamlit
- import line_api: นำเข้าไลบรารีที่สมมติว่าชื่อ line_api, ซึ่งคาดว่าให้ความสามารถในการโต้ตอบกับ LINE Notify.

สร้างหัวข้อหลักบนหน้าเว็บ:

- st.header("ส่งข้อความทาง LINE Notify"): ใช้ฟังก์ชัน header เพื่อแสดงหัวข้อ "ส่งข้อความทาง LINE Notify" บนหน้าเว็บ.

รับข้อมูลจากผู้ใช้:

- mytoken = st.text_input("ป้อน LINE Token : "): สร้างช่องใส่ข้อมูลสำหรับผู้ใช้ใส่ LINE Token.
- text_send = st.text_input("ป้อนข้อความที่ต้องการส่ง : "): สร้างช่องใส่ข้อมูลสำหรับผู้ใช้ใส่ข้อความที่ต้องการส่ง.

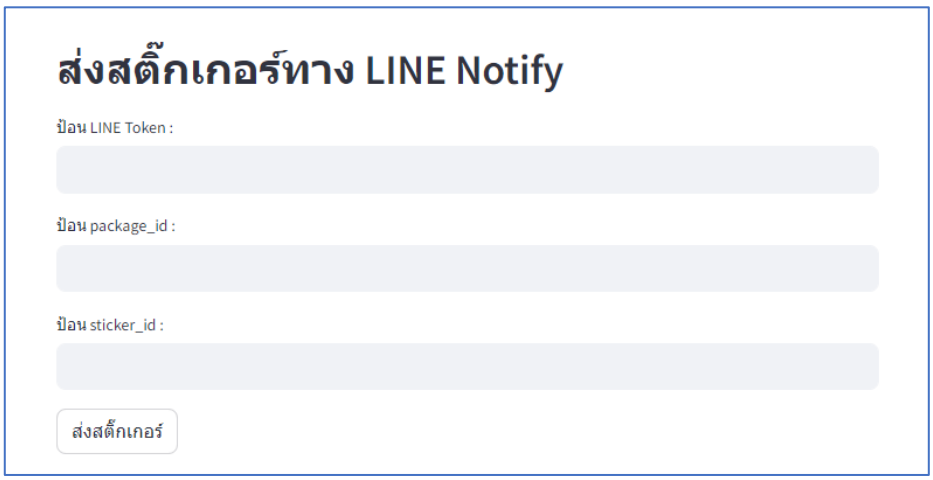
ปุ่มกดสำหรับส่งข้อความและตรวจสอบการดำเนินการ:

- if st.button("ส่งข้อความ"): ตรวจสอบว่าปุ่ม "ส่งข้อความ" ถูกกดหรือไม่.
- try: บล็อก try ใช้สำหรับเรียกใช้งานที่อาจเกิดข้อผิดพลาด.
- line_api.lineNotify(text_send,mytoken): เรียกใช้ฟังก์ชัน lineNotify จากไลบรารี line_api พร้อมข้อความที่ต้องการส่งและ token.
- st.text("ส่งข้อความของคุณแล้ว"): ถ้าไม่มีข้อผิดพลาด, แสดงข้อความนี้บนหน้าเว็บ.
- except: ถ้าเกิดข้อผิดพลาดในบล็อก try, บล็อก except จะดำเนินการ.
- st.text("ไม่สามารถส่งข้อความได้"): แสดงข้อความนี้บนหน้าเว็บหากไม่สามารถส่งข้อความได้.

*** ผลลัพธ์คือเว็บแอปที่ผู้ใช้สามารถป้อน LINE Token และข้อความที่ต้องการส่ง และเมื่อกดปุ่ม "ส่งข้อความ", จะมีการพยายามส่งข้อความไปยัง LINE ผ่าน LINE Notify. ถ้าการส่งสำเร็จ, จะแสดงข้อความยืนยันบนหน้าเว็บ.

ตัวอย่างที่ 10 การส่งสติ๊กเกอร์ทาง LINE Notify

```python
import streamlit as st
import line_api

st.header("ส่งสติ๊กเกอร์ทาง LINE Notify")
mytoken = st.text_input("ป้อน LINE Token : ")
package_id = st.text_input("ป้อน package_id : ")
sticker_id = st.text_input("ป้อน sticker_id : ")

if st.button("ส่งสติ๊กเกอร์"):
    try:
        line_api.notifySticker(int(sticker_id),int(package_id),mytoken)
        st.text("ส่งสติ๊กเกอร์ของคุณแล้ว")
    except:
        st.text("ไม่สามารถส่งสติ๊กเกอร์ได้")

```



- import streamlit as st: นำเข้าไลบรารี Streamlit
- import line_api: นำเข้าไลบรารีที่สมมติว่าชื่อ line_api. สมมติว่าไลบรารีนี้มีฟังก์ชันที่ใช้สำหรับส่งสติกเกอร์ผ่าน LINE Notify.

สร้างหัวข้อบนหน้าเว็บ:

- st.header("ส่งสติ๊กเกอร์ทาง LINE Notify"): ใช้ header เพื่อแสดงหัวข้อ "ส่งสติ๊กเกอร์ทาง LINE Notify" บนหน้าเว็บ.

รับข้อมูลจากผู้ใช้:

- mytoken = st.text_input("ป้อน LINE Token : "): สร้างช่องใส่ข้อมูลสำหรับผู้ใช้ใส่ LINE Token.
- package_id = st.text_input("ป้อน package_id : "): ช่องใส่ข้อมูลสำหรับผู้ใช้ใส่ package_id ของสติกเกอร์.
- sticker_id = st.text_input("ป้อน sticker_id : "): ช่องใส่ข้อมูลสำหรับผู้ใช้ใส่ sticker_id ของสติกเกอร์.

ปุ่มกดสำหรับส่งสติกเกอร์และตรวจสอบการดำเนินการ:

- if st.button("ส่งสติ๊กเกอร์"): ตรวจสอบว่าปุ่ม "ส่งสติ๊กเกอร์" ถูกกดหรือไม่.
- try: บล็อก try ใช้สำหรับเรียกใช้งานที่อาจเกิดข้อผิดพลาด.
- line_api.notifySticker(int(sticker_id), int(package_id), mytoken): เรียกใช้ฟังก์ชัน notifySticker จากไลบรารี line_api ด้วย sticker_id, package_id ที่แปลงเป็นจำนวนเต็ม, และ token.
- st.text("ส่งสติ๊กเกอร์ของคุณแล้ว"): ถ้าไม่มีข้อผิดพลาด, แสดงข้อความนี้บนหน้าเว็บ.
- except: ถ้าเกิดข้อผิดพลาดในบล็อก try, บล็อก except จะดำเนินการ.
- st.text("ไม่สามารถส่งสติ๊กเกอร์ได้"): แสดงข้อความนี้บนหน้าเว็บหากไม่สามารถส่งสติกเกอร์ได้.

*** ผลลัพธ์คือเว็บแอปที่ให้ผู้ใช้ป้อน LINE Token และรายละเอียดของสติกเกอร์ (package_id และ sticker_id) และสามารถส่งสติกเกอร์ผ่าน LINE Notify โดยกดปุ่ม "ส่งสติ๊กเกอร์".

ตัวอย่างที่1 การแสดงภาพนิ่งด้วย St.image

PYSL042.py - D:\VDO WepApp\code_pycvwebapp2\PYSL042.py (3.7.9)

File Edit Format Run Options Window Help

```
import streamlit as st
import cv2

img = cv2.imread("zebra.jpg")

st.image(img, caption='ม้าลาย',channels="BGR") #OpenCV BGR  RGB

```

ม้าลาย

ตัวอย่างที่2 การแสดงภาพนิ่งพร้อมด้วยความกว้างและความสูง

PYSL043.py - D:\VDO WepApp\code_pycvwebapp2\PYSL043.py (3.7.9)

File Edit Format Run Options Window Help

```
import streamlit as st
import cv2

img = cv2.imread("bird.jpg")
(h,w,c) = img.shape

st.image(img, caption="นก",channels="BGR")
st.text("ภาพนี กว้าง " + str(w) + " สูง " + str(h))

```

นก

ภาพนี้ กว้าง 400 สูง 267

ตัวอย่างที่3 การแสดงภาพนิ่งแบบ 3 คอลัมน์ด้วยst.columns

PYSL044.py - D:\VDO WepApp\code_pycvwebapp2\PYSL044.py (3.7.9)

File Edit Format Run Options Window Help

```
import streamlit as st
import cv2

img1 = cv2.imread("zebra.jpg")
img2 = cv2.imread("duck.jpg")
img3 = cv2.imread("bird.jpg")

col1, col2, col3 = st.columns(3)

col1.image(img1, caption='ม้าลาย',channels="BGR")
col2.image(img2, caption='เป็ด',channels="BGR")
col3.image(img3, caption='นก',channels="BGR")

```

ม้าลาย

เป็ด

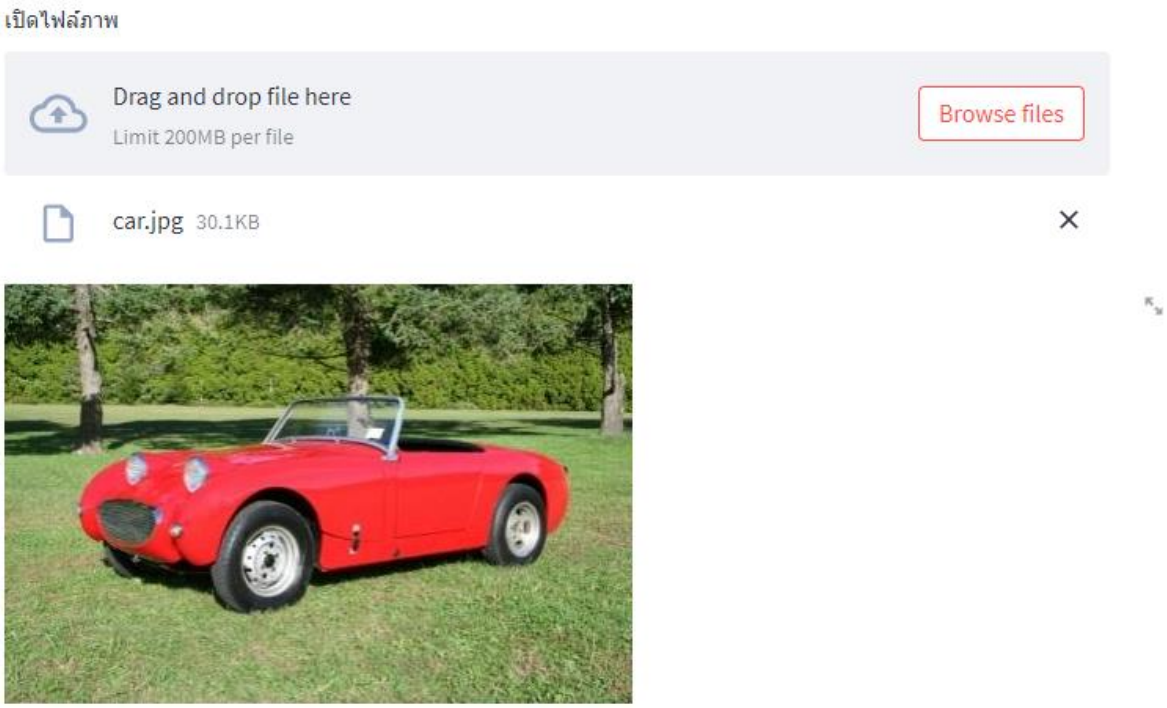
นก

ตัวอย่างที่4 การเปิดไฟล์ภาพนิ่งด้วย st.file_uploader

PYSL045.py - D:\VDO WepApp\code_pycvwebapp2\PYSL045.py (3.7.9)

File Edit Format Run Options Window Help

```
import streamlit as st
import cv2

img_file = st.file_uploader("เปิดไฟล์ภาพ")

if img_file is not None:
    st.image(img_file,channels="BGR")

```

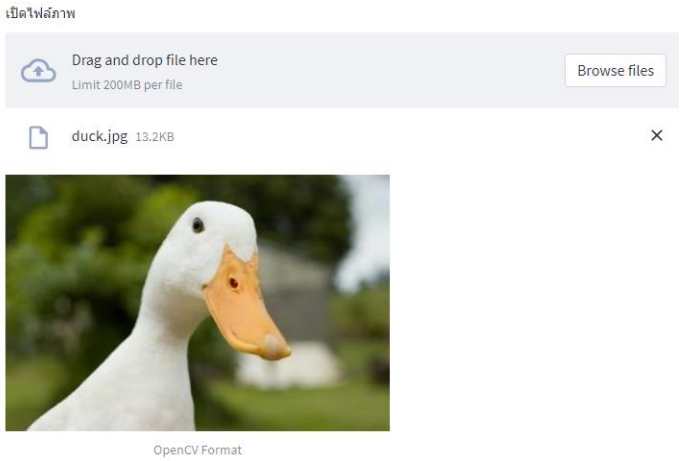


ตัวอย่างที่5 การเปิดไฟล์ภาพนิ่งด้วย st.file_uploader แล้วแปลงให้อยู่ในรูปแบบของ OpenCV

PYSL046.py - D:\VDO WepApp\code_pycvwebapp2\PYSL046.py (3.7.9)

File Edit Format Run Options Window Help

```
1  import streamlit as st
2  import cv2
3  import numpy as np
4
5  img_file = st.file_uploader("เปิดไฟล์ภาพ")
6
7  if img_file is not None:
8      file_bytes = np.asarray(bytearray(img_file.read()), dtype=np.uint8)
9      img = cv2.imdecode(file_bytes, cv2.IMREAD_COLOR)
10     st.image(img ,caption="OpenCV Format",channels="BGR")
11
```

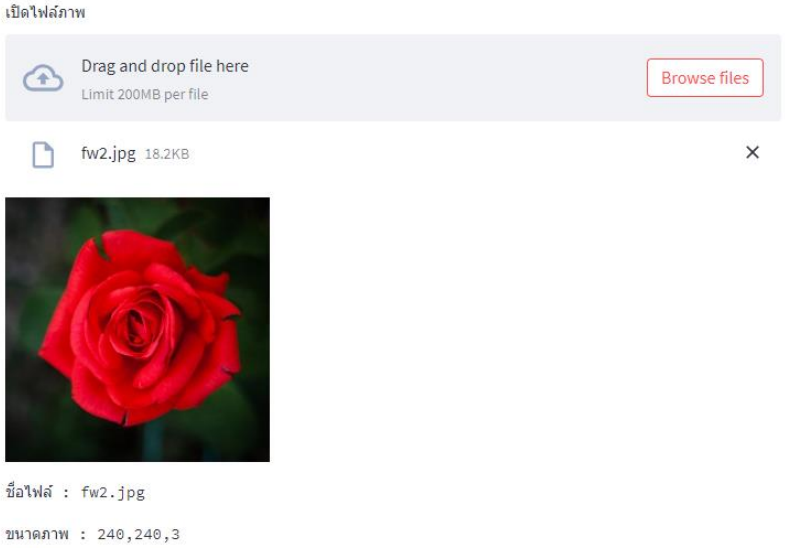


ตัวอย่างที่6 การเปิดไฟล์ภาพนิ่งด้วย st.file_uploader พร้อมแสดงชื่อไฟล์และขนาดภาพ

PYSL047.py - D:\VDO WepApp\code_pycvwebapp2\PYSL047.py (3.7.9)

File Edit Format Run Options Window Help

```python
import streamlit as st
import cv2
import numpy as np

img_file = st.file_uploader("เปิดไฟล์ภาพ")

if img_file is not None:
    print(img_file)
    file_bytes = np.asarray(bytearray(img_file.read()), dtype=np.uint8)
    img = cv2.imdecode(file_bytes, cv2.IMREAD_COLOR)
    (h,w,c) = img.shape
    st.image(img ,channels="BGR")
    st.text("ชื่อไฟล์ : " + img_file.name)
    st.text("ขนาดภาพ : " + str(w) + "," + str(h) + "," + str(c) )
```

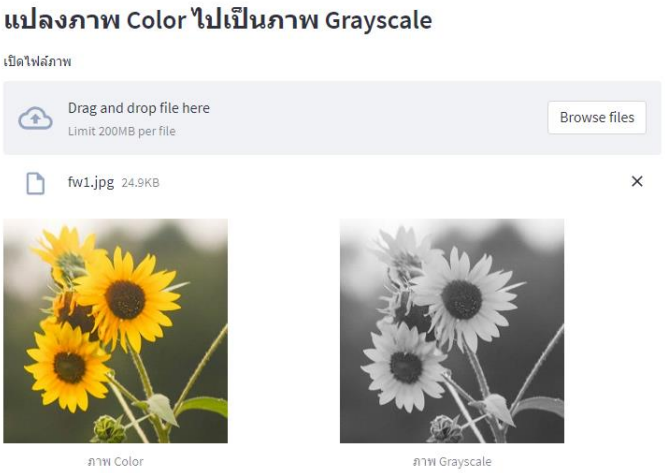


ตัวอย่างที่7 การแปลงภาพ Color ไปเป็นภาพ Grayscale

PYSL048.py - D:\VDO WepApp\code_pycvwebapp2\PYSL048.py (3.7.9)

File Edit Format Run Options Window Help

```
import streamlit as st
import cv2
import numpy as np

st.subheader("แปลงภาพ Color ไปเป็นภาพ Grayscale")
img_file = st.file_uploader("เปิดไฟล์ภาพ")

if img_file is not None:
    file_bytes = np.asarray(bytearray(img_file.read()), dtype=np.uint8)
    img = cv2.imdecode(file_bytes, cv2.IMREAD_COLOR)
    #-----------------------------------------------
    img_out = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    #-----------------------------------------------
    col1, col2 = st.columns(2)
    col1.image(img, caption='ภาพ Color',channels="BGR")
    col2.image(img_out, caption='ภาพ Grayscale')
```



ตัวอย่างที่8 การแปลงไปเป็นภาพ Binary

PYSL049.py - D:\VDO WepApp\code_pycvwebapp2\PYSL049.py (3.7.9)

File Edit Format Run Options Window Help

```
import streamlit as st
import cv2
import numpy as np

st.subheader("แปลงภาพไปเป็น Binary")
img_file = st.file_uploader("เปิดไฟล์ภาพ")

if img_file is not None:
    file_bytes = np.asarray(bytearray(img_file.read()), dtype=np.uint8)
    img = cv2.imdecode(file_bytes, cv2.IMREAD_COLOR)
    #-----------------------------------------------
    img_gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    ret,BW = cv2.threshold(img_gray,120,255,cv2.THRESH_BINARY_INV)
    #-----------------------------------------------
    col1,col2,col3 = st.columns(3)
    col1.image(img, caption='ภาพ Color',channels="BGR")
    col2.image(img_gray, caption='ภาพ Grayscale')
    col3.image(BW, caption='ภาพ Binary')
```

แปลงภาพไปเป็น Binary

เปิดไฟล์ภาพ

Drag and drop file here

Limit 200MB per file

Browse files

padthai.jpg 35.4KB

ภาพ Color

ภาพ Grayscale

ภาพ Binary

ตัวอย่างที่9 การใส่ข้อความลงบนภาพ

PYSL052.py - D:\VDO WepApp\code_pycvwebapp2\PYSL052.py (3.7.9)

File Edit Format Run Options Window Help

```
import streamlit as st
import cv2
import numpy as np

img_file = st.file_uploader("เปิดไฟล์ภาพ")

if img_file is not None:
    file_bytes = np.asarray(bytearray(img_file.read()), dtype=np.uint8)
    img = cv2.imdecode(file_bytes, cv2.IMREAD_COLOR)
    #-----------------------------------------------------
    img_out = img.copy()
    text = "Hello"
    font = cv2.FONT_HERSHEY_SIMPLEX
    cv2.putText(img_out,text,(20,100),font,3.0,(0,255,0),3)
    #-----------------------------------------------------
    col1, col2 = st.columns(2)
    col1.image(img, caption='ภาพ Input',channels="BGR")
    col2.image(img_out, caption='ภาพ Output',channels="BGR")

```

เปิดไฟล์ภาพ

Drag and drop file here

Limit 200MB per file

Browse files

elephant.jpg 35.0KB

ภาพ Input

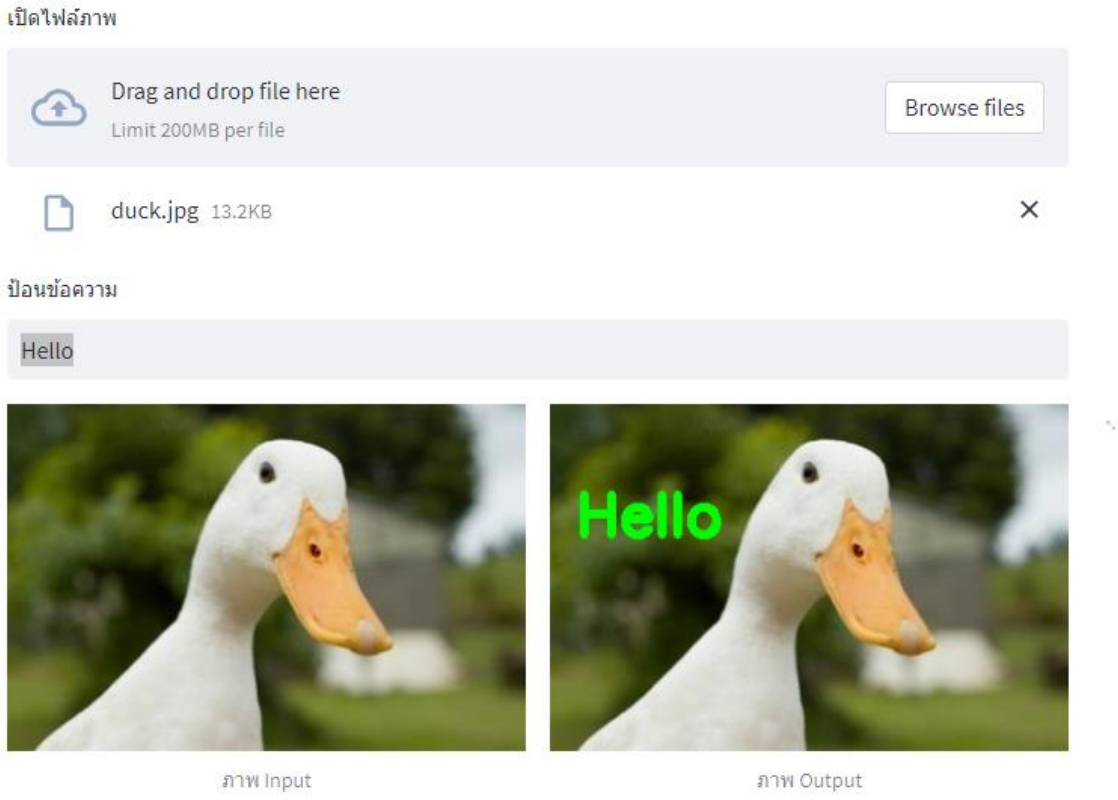
ภาพ Output

ตัวอย่างที่10 การใส่ข้อความลงบนภาพแบบป้อนข้อความได้

PYSL053.py - D:\VDO WepApp\code_pycvwebapp2\PYSL053.py (3.7.9)

File Edit Format Run Options Window Help

```
import streamlit as st
import cv2
import numpy as np

img_file = st.file_uploader("เปิดไฟล์ภาพ")

def put_text(img,text):
    #-----------------------------------------------------
    img_out = img.copy()
    font = cv2.FONT_HERSHEY_SIMPLEX
    cv2.putText(img_out,text,(20,100),font,1.5,(0,255,0),5)
    #-----------------------------------------------------
    col1, col2 = st.columns(2)
    col1.image(img, caption='ภาพ Input',channels="BGR")
    col2.image(img_out, caption='ภาพ Output',channels="BGR")

if img_file is not None:
    file_bytes = np.asarray(bytearray(img_file.read()), dtype=np.uint8)
    img = cv2.imdecode(file_bytes, cv2.IMREAD_COLOR)
    text_in = st.text_input("ป้อนข้อความ","Hello")
    put_text(img,text_in)
```

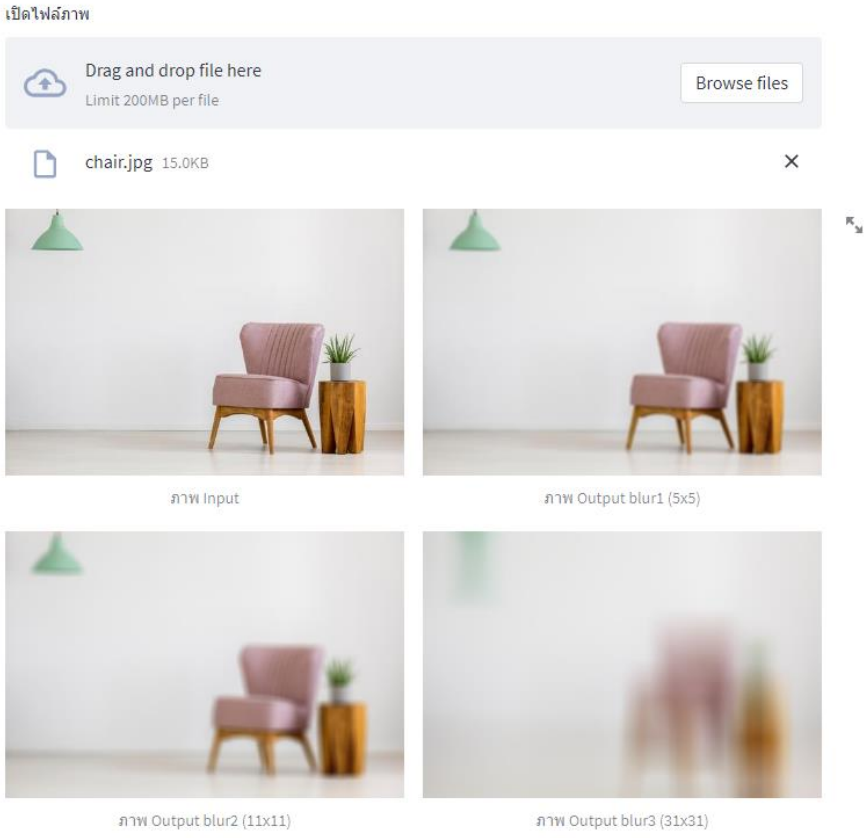


ตัวอย่างที่11 การเบลอภาพด้วย Average Filter

PYSL054.py - D:\VDO WepApp\code_pycvwebapp2\PYSL054.py (3.7.9)

File Edit Format Run Options Window Help

```
import streamlit as st
import cv2
import numpy as np

img_file = st.file_uploader("เปิดไฟล์ภาพ")

if img_file is not None:
    file_bytes = np.asarray(bytearray(img_file.read()), dtype=np.uint8)
    img = cv2.imdecode(file_bytes, cv2.IMREAD_COLOR)
    #-----------------------------------------------
    img_blur1 = cv2.blur(img,(5,5))
    img_blur2 = cv2.blur(img,(11,11))
    img_blur3 = cv2.blur(img,(31,101))
    #-----------------------------------------------
    col1, col2 = st.columns(2)
    col1.image(img, caption='ภาพ Input',channels="BGR")
    col2.image(img_blur1, caption='ภาพ Output blur1 (5x5)',channels="BGR")

    col1, col2 = st.columns(2)
    col1.image(img_blur2, caption='ภาพ Output blur2 (11x11)',channels="BGR")
    col2.image(img_blur3, caption='ภาพ Output blur3 (31x31)',channels="BGR")
```



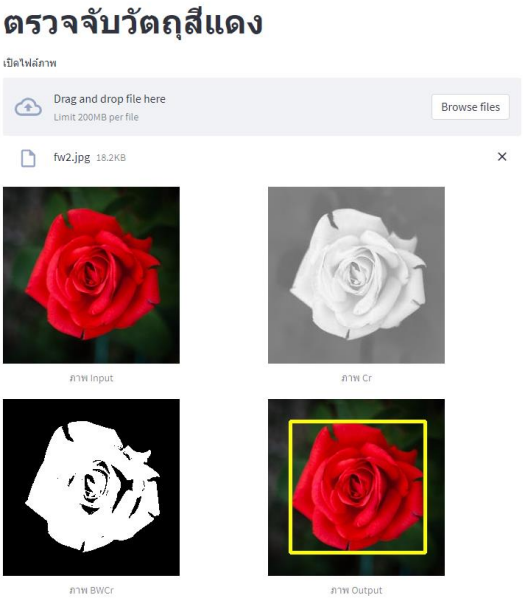
ตัวอย่างที่12 การตรวจจับวัตถุสีแดงในภาพนิ่งโดยใช้ Cr

PYSL057.py - D:\VDO WepApp\code_pycvwebapp2\PYSL057.py (3.7.9)

File Edit Format Run Options Window Help

```
import streamlit as st
import cv2
import numpy as np

st.title("ตรวจจับวัตถุสีแดง")
img_file = st.file_uploader("เปิดไฟล์ภาพ")

if img_file is not None:
    file_bytes = np.asarray(bytearray(img_file.read()), dtype=np.uint8)
    img = cv2.imdecode(file_bytes, cv2.IMREAD_COLOR)
    #------------------------------------------------
    imgYCrCb  = cv2.cvtColor(img,cv2.COLOR_BGR2YCR_CB)

    channels = cv2.split(imgYCrCb)# Y 0   Cr 1   Cb 2
    Cr = channels[1]

    ret,BW = cv2.threshold(Cr,190,255,cv2.THRESH_BINARY)

    contours, hierarchy = cv2.findContours(BW,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIM

    if len(contours) > 0: #เจอ
        areas = [cv2.contourArea(c) for c in contours]
        max_index = np.argmax(areas) #[ 20000 350000 456 ] 0 1 2
        cnt = contours[max_index]

        x,y,w,h = cv2.boundingRect(cnt)

        img_out = img.copy()
        cv2.rectangle(img_out,(x,y),(x+w,y+h),(0,255,255),4) #BGR
        #------------------------------------------------
        col1, col2 = st.columns(2)
        col1.image(img, caption='ภาพ Input',channels="BGR")
        col2.image(Cr, caption='ภาพ Cr')

        col1, col2 = st.columns(2)
        col1.image(BW, caption='ภาพ BWCr')
        col2.image(img_out, caption='ภาพ Output',channels="BGR")
```

Ln: 1 Col: 0

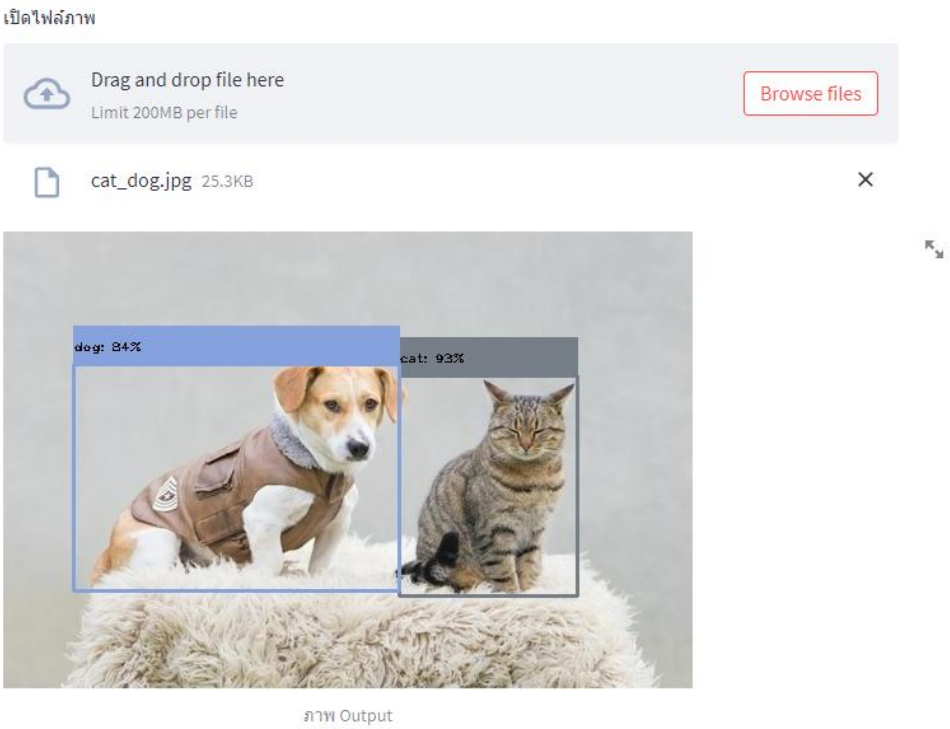


ตัวอย่างที่13 การตรวจจับวัตถุโดยใช้ YOLO Model ex20obj.onnx กับภาพนิ่ง

PYSL059.py - D:\VDO WepApp\code_pycvwebapp2\PYSL059.py (3.7.9)

File Edit Format Run Options Window Help

```python
import streamlit as st
import cv2
import numpy as np
from yolo_predictions import YOLO_Pred

yolo = YOLO_Pred('ex20obj.onnx','ex20obj.yaml')

img_file = st.file_uploader("เปิดไฟล์ภาพ")

if img_file is not None:
    file_bytes = np.asarray(bytearray(img_file.read()), dtype=np.uint8)
    img = cv2.imdecode(file_bytes, cv2.IMREAD_COLOR)
    #----------------------------------------------
    pred_image, obj_box = yolo.predictions(img)
    #----------------------------------------------
    st.image(pred_image, caption='ภาพ Output',channels="BGR")


```



ตัวอย่างที่14 การตรวจจับวัตถุโดยใช้ YOLO Model ex20obj.onnx กับภาพนิ่ง พร้อมแสดงชื่อวัตถุ

PYSL060.py - D:\VDO WepApp\code_pycvwebapp2\PYSL060.py (3.7.9)

File Edit Format Run Options Window Help

```python
import streamlit as st
import cv2
import numpy as np
from yolo_predictions import YOLO_Pred

yolo = YOLO_Pred('ex20obj.onnx','ex20obj.yaml')

img_file = st.file_uploader("เปิดไฟล์ภาพ")

if img_file is not None:
    file_bytes = np.asarray(bytearray(img_file.read()), dtype=np.uint8)
    img = cv2.imdecode(file_bytes, cv2.IMREAD_COLOR)
    #----------------------------------------------
    count = 0  #***
    pred_image, obj_box = yolo.predictions(img)
    print(obj_box)
    if len(obj_box) > 0: #เจอวัตถุใน 20 อย่างนัน
        obj_names = ''
        for obj in obj_box:
            obj_names = obj_names + obj[4] + ' '
            if obj[4] == "person":  #***
                count = count + 1   #***
        text_obj = 'ตรวจพบ ' + obj_names
    else:
        text_obj = 'ไม่พบวัตถุ'
    #----------------------------------------------
    st.image(pred_image, caption='ภาพ Output',channels="BGR")
    st.text(text_obj)
    st.text("พบ person " + str(count) + " คน") #***
```

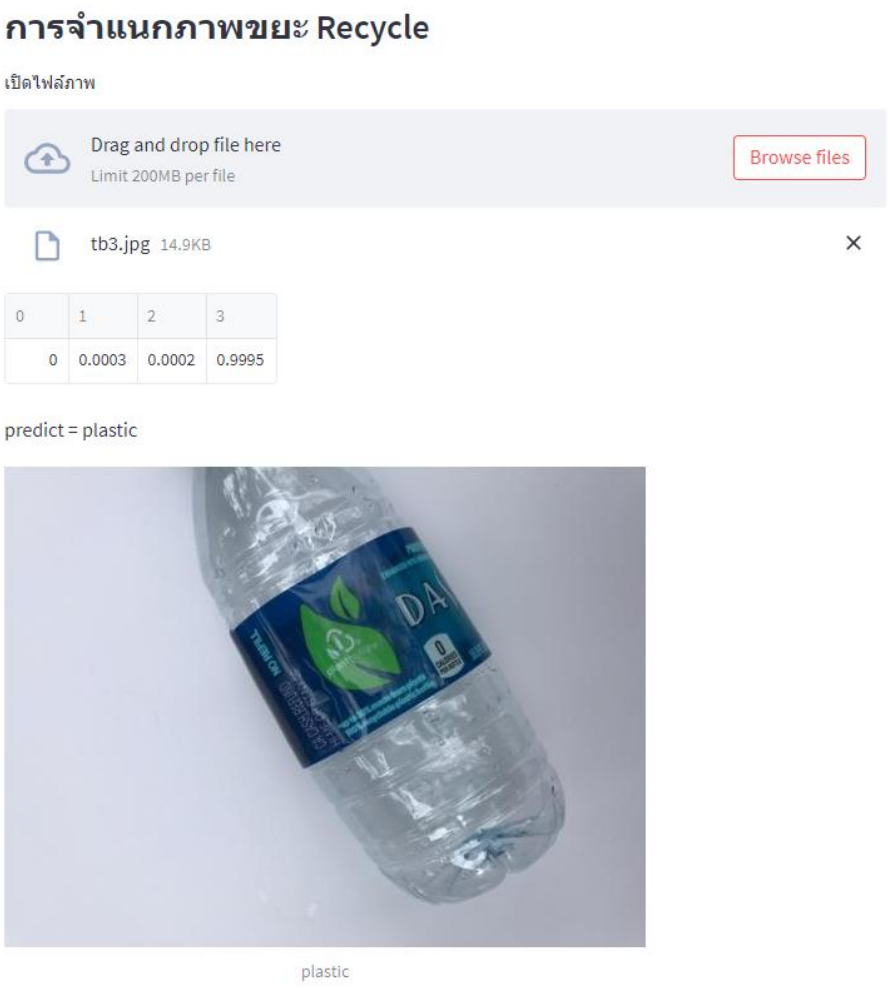
ตัวอย่างที่15 การจำแนกภาพขยะ Recycle ได้แก่ cardboard, glass, metal, plastic โดยใช้ CNN Model recycle.h5 กับภาพนิ่ง

PYSL062.py - D:\VDO WepApp\code_pycvwebapp2\PYSL062.py (3.7.9)

File Edit Format Run Options Window Help

```
import streamlit as st
import cv2
import numpy as np

from tensorflow.keras.preprocessing.image import load_img, img_to_array
from tensorflow.keras.models import load_model

from tensorflow.keras.applications.resnet50 import preprocess_input

model = load_model('recycle.h5',compile=False)

target_img_shape=(128,128)

st.subheader("การจำแนกภาพขยะ Recycle")
img_file = st.file_uploader("เปิดไฟล์ภาพ")

if img_file is not None:
    file_bytes = np.asarray(bytearray(img_file.read()), dtype=np.uint8)
    img = cv2.imdecode(file_bytes, cv2.IMREAD_COLOR)
    #-----------------------------------------------
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB )
    test_image = cv2.resize(img,target_img_shape)

    test_image = img_to_array(test_image)
    test_image = preprocess_input(test_image)

    test_image = np.expand_dims(test_image,axis=0) # (1, 128, 128, 3)

    result = model.predict(test_image)
    st.write(result)

    class_answer = np.argmax(result,axis=1)
    if class_answer == 0:
        predict = 'cardboard'
    elif class_answer == 1:
        predict = 'glass'
    elif class_answer == 2:
        predict = 'metal'
    elif class_answer == 3:
        predict = 'plastic'
    st.write("predict = "+predict)
    #-----------------------------------------------
    st.image(img ,caption=predict,channels="RGB")
```

Ln: 1 Col: 0



การจัดการกล้อง Webcam

การเชื่อมต่อกล้อง Webcam

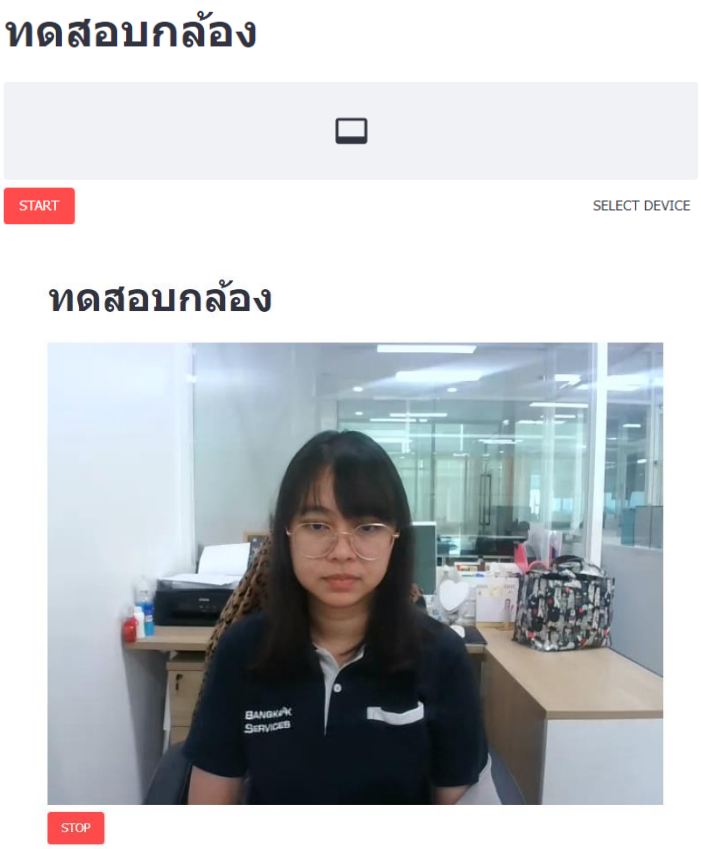
```python
import streamlit as st
from streamlit_webrtc import webrtc_streamer

st.title("ทดสอบกล้อง")

webrtc_streamer(key="test",
                media_stream_constraints={"video": True,"audio": False})
```

นำเข้าไลบรารี streamlit เพื่อใช้ในการสร้างแอปพลิเคชันเว็บ และ streamlit_webrtc เพื่อใช้ในการสร้างสตรีมเว็บแคม

st.title() ใช้สร้างหัวข้อในแอปพลิเคชันเว็บ โดย "ทดสอบกล้อง" เป็นข้อความที่จะแสดงเป็นหัวข้อ

webrtc_streamer () เป็นฟังก์ชันที่ใช้ในการสตรีมวิดีโอและเสียงผ่านเว็บ โดยรับพารามิเตอร์หลักเป็น key ซึ่งเป็นรหัสที่ใช้เพื่อแยกและระบุวิดีโอและพารามิเตอร์ media_stream_constraints ซึ่งกำหนดเงื่อนไขสำหรับสตรีมวิดีโอและเสียง ในที่นี้กำหนดให้สตรีมวิดีโอเท่านั้น (ไม่มีเสียง) ด้วยการให้ค่า {"video": True, "audio": False}

ผลการรันโปรแกรม



การ Flip ภาพจากกล้อง

```python
import streamlit as st
from streamlit_webrtc import webrtc_streamer
import av
import cv2

st.title("การกลับภาพ")

class VideoProcessor:  #สร้าง class สำหรับประมวลผลภาพ
    def recv(self, frame):
        img = frame.to_ndarray(format="bgr24") #24bit 8B 8G 8R
        #------------------------------
        img = cv2.flip(img,1) #แกน x 0 บน-ล่าง  แกน y 1 ซ้าย-ขวา
        #------------------------------
        return av.VideoFrame.from_ndarray(img, format="bgr24")

webrtc_streamer(key="test",
                video_processor_factory=VideoProcessor, #คลาส
                media_stream_constraints={"video": True,"audio": False})

```

นำเข้าไลบรารี streamlit เพื่อใช้ในการสร้างแอปพลิเคชันเว็บ และ streamlit_webrtc เพื่อใช้ในการสร้างสตรีมเว็บแคม นอกจากนี้ยังนำเข้าไลบรารี av และ cv2 เพื่อใช้ในการจัดการวิดีโอและการประมวลผลภาพ

กำหนดหัวข้อของแอปพลิเคชันเว็บเป็น "การ Flip ภาพ"

สร้างคลาส VideoProcessor สำหรับการประมวลผลวิดีโอ ในคลาสนี้ เรากำหนดเมธอด recv() เพื่อรับเฟรมวิดีโอที่สตรีมจากกล้องเว็บ ซึ่งจะนำเข้าเป็นรูปแบบ bgr24 (Blue-Green-Red 24-bit) และทำการ flip ภาพ ใช้ฟังก์ชัน cv2.flip() ของ OpenCV เพื่อทำการกลับด้านภาพ โดยในที่นี้กำหนดให้ flip แกน y (ซ้าย-ขวา)

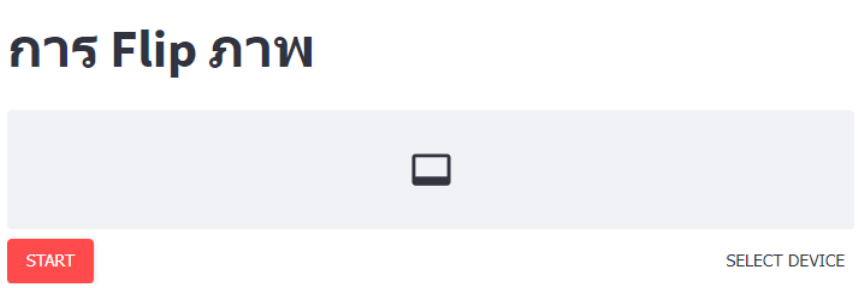
* flip แกน x (บน-ล่าง) img = cv2.flip(img,0)

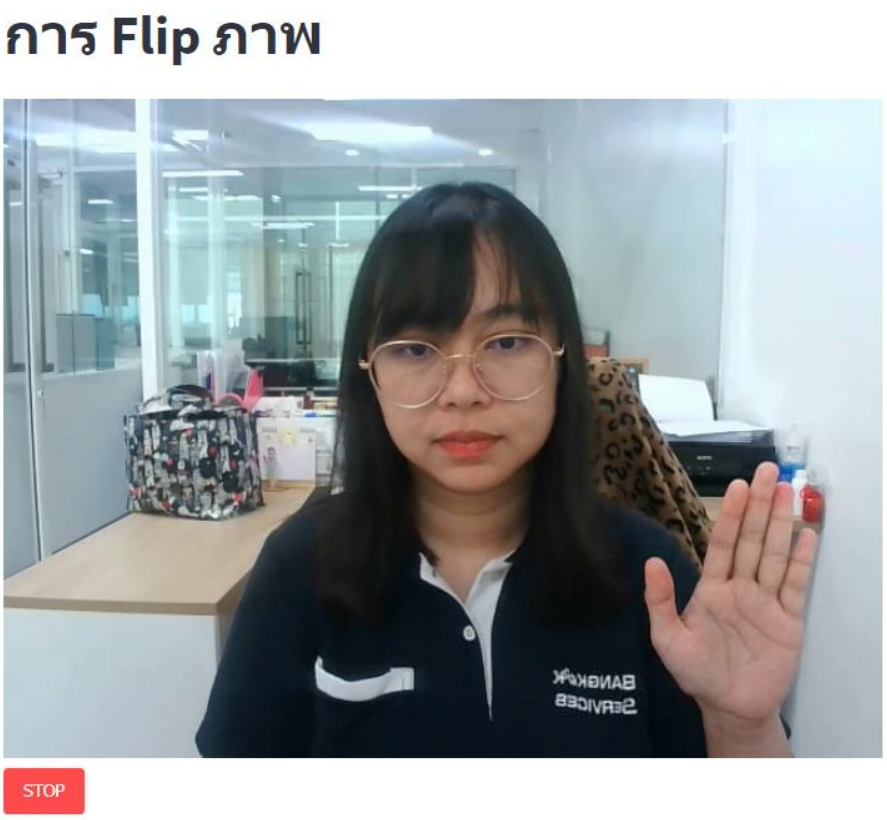
* flip แกน y (ซ้าย-ขวา) img = cv2.flip(img,1)

ใช้ webrtc_streamer() เพื่อเริ่มสตรีมวิดีโอผ่านเว็บ ระบุ key="test" เพื่อระบุวิดีโอเร็กคอร์ด และ video_processor_factory=VideoProcessor เพื่อระบุคลาส VideoProcessor ที่ใช้ในการประมวลผลวิดีโอ นอกจากนี้ยังระบุ media_stream_constraints={"video": True, "audio": False} เพื่อเฉพาะการสตรีมวิดีโอเท่านั้น

ดังนั้นโค้ดนี้จะสร้างเว็บแอปพลิเคชันที่มีหัวข้อ "การ Flip ภาพ" และจะแสดงวิดีโอจากกล้องเว็บ โดยภาพจะถูกกลับด้านในแกน y (ซ้าย-ขวา) โดยใช้ OpenCV ก่อนที่จะแสดงผลออกทางเว็บไซต์

ผลการรันโปรแกรม





การตรวจจับใบหน้าด้วยวิธี Haar

PYSL075.py > ...

```python
import streamlit as st
from streamlit_webrtc import webrtc_streamer
import av
import cv2
import numpy as np

face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

st.title("ตรวจจับใบหน้า")

class VideoProcessor:
    def recv(self, frame):
        img = frame.to_ndarray(format="bgr24")
        #------------------------------------------
        img = cv2.flip(img,1)
        img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        faces = face_cascade.detectMultiScale(img_gray, 1.3, 5)
        for (x,y,w,h) in faces:
            cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,0),5)
        #------------------------------------------
        return av.VideoFrame.from_ndarray(img,format="bgr24")

webrtc_streamer(key="test",
                video_processor_factory=VideoProcessor,
                media_stream_constraints={"video": True,"audio": False})

```

นำเข้าไลบรารี streamlit เพื่อใช้ในการสร้างแอปพลิเคชันเว็บ และ streamlit_webrtc เพื่อใช้ในการสร้างสตรีมเว็บแคม นอกจากนี้ยังนำเข้าไลบรารี av และ cv2 เพื่อใช้ในการจัดการวิดีโอและการประมวลผลภาพ

การตรวจจับใบหน้าด้วย Haar Cascade Classifier ซึ่งเป็นวิธีการที่พอเห็นได้ง่ายและเป็นที่นิยมในการตรวจจับใบหน้าในภาพ เริ่มต้นจากการกำหนด Haar Cascade Classifier ด้วยการโหลดไฟล์ XML haarcascade_frontalface_default.xml ซึ่งเป็นโมเดลสำหรับตรวจจับใบหน้าที่ถูกฝึกสอนมาอย่างดี

กำหนดหัวข้อของแอปพลิเคชันเว็บเป็น "การ Flip ภาพ"

สร้างคลาส VideoProcessor ในเมธอด recv() เราทำการแปลงภาพเป็นภาพขาว-ดำ (grayscale) และใช้ detectMultiScale() เพื่อตรวจจับใบหน้าในภาพ หากตรวจพบใบหน้า เราจะวาดสี่เหลี่ยมรอบใบหน้าโดยใช้ cv2.rectangle() ที่มีสีเขียว และความหนาของเส้น 5 เพื่อทำให้เด่นขึ้น

ใช้ webrtc_streamer() เหมือนเดิมเพื่อเริ่มสตรีมวิดีโอ กำหนด key="test" เพื่อระบุวิดีโอเร็กคอร์ด และระบุ video_processor_factory=VideoProcessor เพื่อใช้คลาส VideoProcessor ที่เราสร้างไว้ในการประมวลผลวิดีโอ

ดังนั้น แอปพลิเคชันเว็บนี้จะแสดงวิดีโอจากกล้องเว็บ และตรวจจับใบหน้าในภาพ และวาดสี่เหลี่ยมรอบใบหน้าในเฟรมวิดีโอที่ตรวจพบใบหน้า

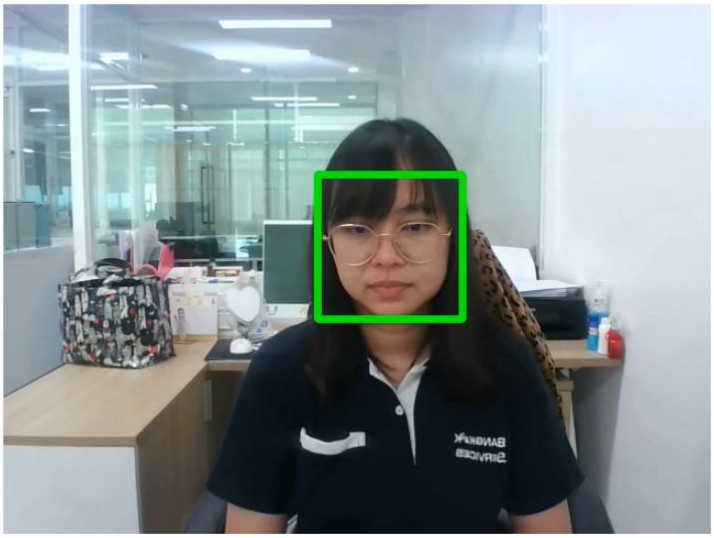
ผลการรันโปรแกรม

ตรวจจับใบหน้า

START

SELECT DEVICE

ตรวจจับใบหน้า



STOP