Tecnológico de Monterrey | Educación Continua

Aplicación Web de Ciencia de datos

Programas LIVE

SS1 | Aprender

Tecnológico de Monterrey | Educación Continua

Bienvenida y presentación

Educación Continua

Bienvenida

“En algún lugar, algo increíble está esperando ser conocido”

-Carl Sagan

Educación Continua

Objetivo principal del módulo

- Explotarás las **ventajas** que tienen **Python** y la plataforma **Streamlit**, para generar Dashboards que analizen de forma interactiva los **datos** de las organizaciones para una correcta toma de decisiones, cumpliendo con los requerimientos de las **interfaces web**

Educación Continua

Objetivos particulares de la sesión

- Identificarás los **elementos fundamentales** para el diseño de dashboards
- Visualizarás los **datos** provenientes de fuentes de **datos externas** en una **plataforma web**
- Analizarás la información usando **representaciones gráficas** de forma interactiva con controles de usuario

Educación Continua

Agenda

1. **Sesión Síncrona 1**
   Aprender
2. **Trabajo asíncrono 1**
   Profundizar | Ruta de Aprendizaje
3. **Sesión Síncrona 2**
   Preparar para Aplicar
4. **Trabajo asíncrono 2**
   Aplicar en el trabajo | Reto

Educación Continua

Dinámica de participación y convivencia

Sesión grupal:

- Las dudas durante la sesión se manejarán a través del chat. Éstas serán respondidas en los momentos definidos para ello
- Participar en las dinámicas propuestas durante la sesión de acuerdo con las instrucciones dadas para cada una

Rooms:

- Los equipos se harán de manera aleatoria y serán sólo para las actividades de práctica
- Estar atentos a la notificación de zoom para unirse al *room*
- Seguir al pie de la letra las instrucciones para la dinámica en *rooms*
- Informar al moderador si te encuentras sólo en el *room* para reasignarte
- Te pedimos no salir del *room* que tienes asignado. Cualquier reasignación será realizada por parte del tutor o staff

Educación Continua

Acerca del facilitador

- Doctorado en **Ingeniería de software**
- Cómputo en la nube con orquestación de contenedores
- Desarrollo dirigido por pruebas con enfoque en Devops
- Frontend y Backend

Introducción

- Las **aplicaciones web** son una de las formas más convenientes de mostrar el trabajo que se realiza en la **ciencia de datos**
- La **creación de aplicaciones web** puede resultar abrumadora para muchos científicos de datos si no tienen experiencia en desarrollo web
- En **Python**, se pueden crear aplicaciones web interactivas con **Streamlit**; este marco increíble hace el trabajo difícil para hacer y diseñar elementos web y dejar que el científico de datos simplemente se centre en la parte de los datos y en el análisis

Educación Continua

Panorámica de la sesión

1

Sesión Síncrona 1
Aprender

- Tema 1: Plataforma Streamlit
- Tema 2: Manipulación de datos
- Tema 3: Interacción con componentes básicos
- Cierre de sesión

Educación Continua

Panorámica de la sesión

1

Sesión Síncrona 1
Aprender

Tema 1: Plataforma Streamlit

Tema 2: Manipulación de datos

Tema 3: Interacción con componentes básicos

Cierre de sesión

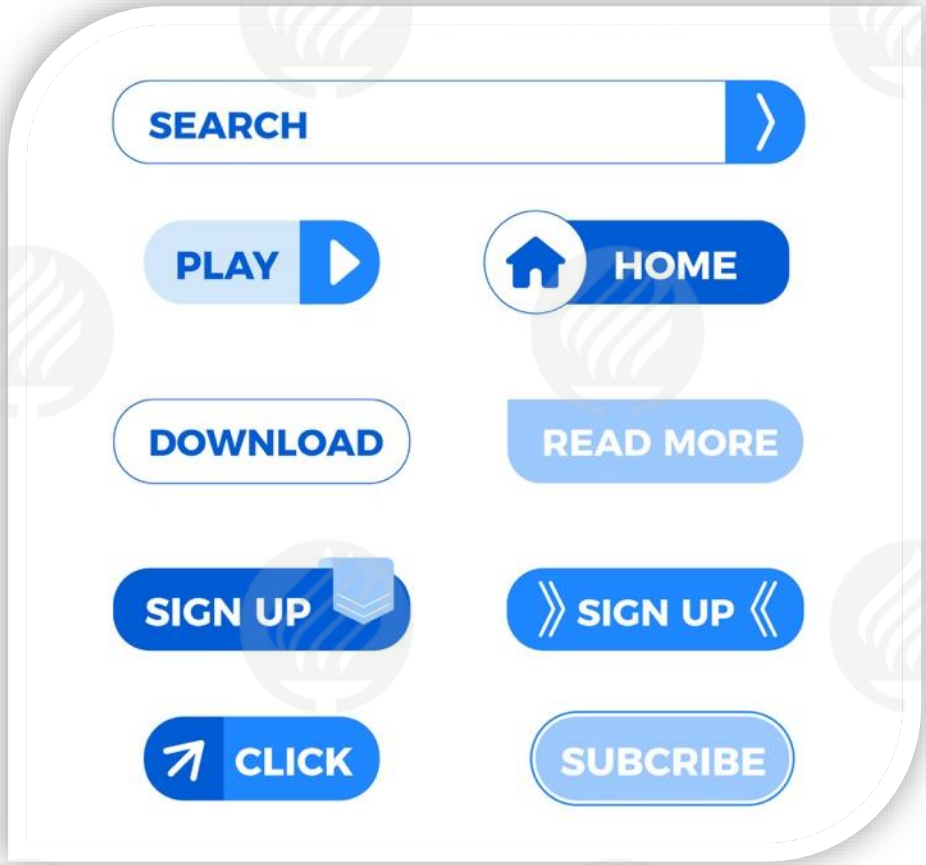
Propósito | Tema 1 Plataforma Streamlit

- **Streamlit** es un **paquete abierto de Python** que le ayuda a **crear aplicaciones web interactivas desplegables** sin ningún conocimiento de HTML, Javascript o CSS

- Es posible con pocas líneas de código crear un tablero interactivo simple, pero funcional, con sólo algunas librerías fundamentales como **Pandas y Numpy**

- Streamlit puede **actualizar automáticamente sus aplicaciones web** cada vez que se modifican las entradas de los datos externos

Propósito | Tema 1 Plataforma Streamlit



Streamlit es un paquete que posee una variedad de **controles interactivos y características para creardashboards** en minutos.

Estos son algunos ejemplos:

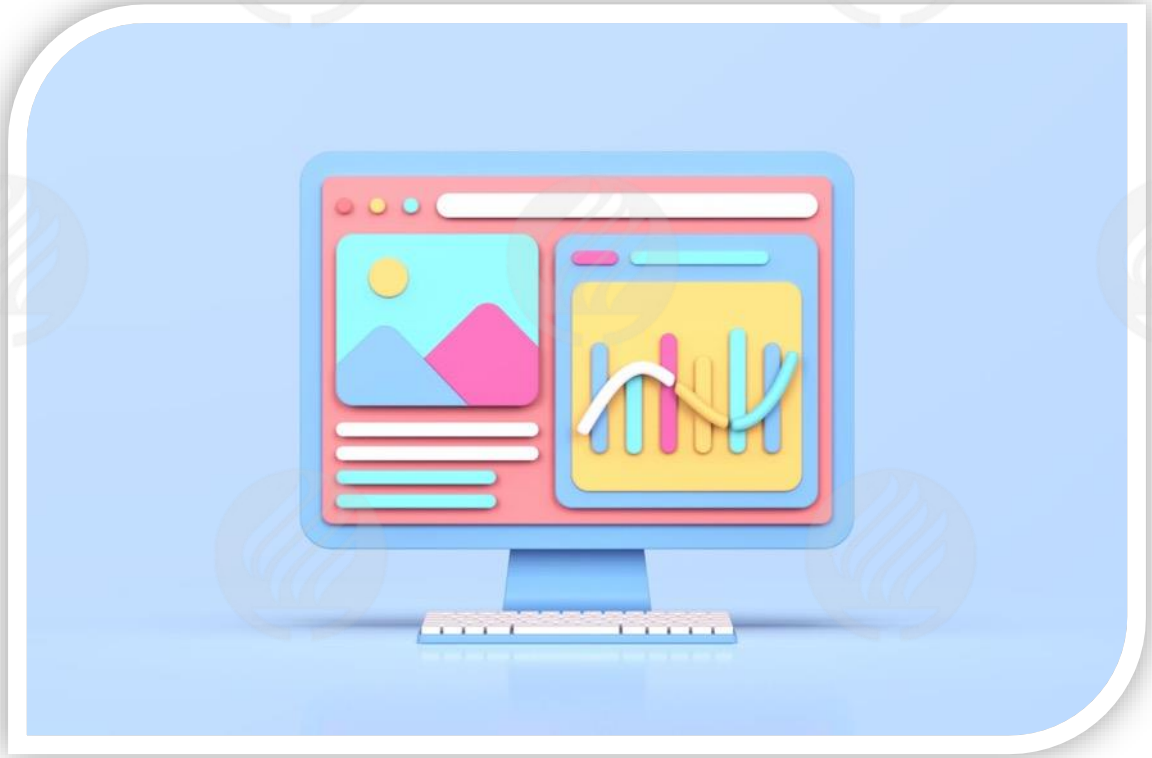
- Acceso a dataframes de Pandas
- Manejo de caché
- Control button
- Control selectbox
- Control text_input
- Control sidebar
- Control slider
- Control radio
- Control checkbox
- Control date_input
- Gráficas
- Visualización de mapas

Propósito | Tema 1 Plataforma Streamlit

Crear una **aplicación simple** en **Streamlit**, que deberá contener al menos las siguientes secciones:

- ✓ Titulo
- ✓ Encabezados
- ✓ Descripción del proyecto
- ✓ Controles interactivos, gráficos, datos (se analizarán en la siguiente sección)

Plataforma Streamlit | Tema 1



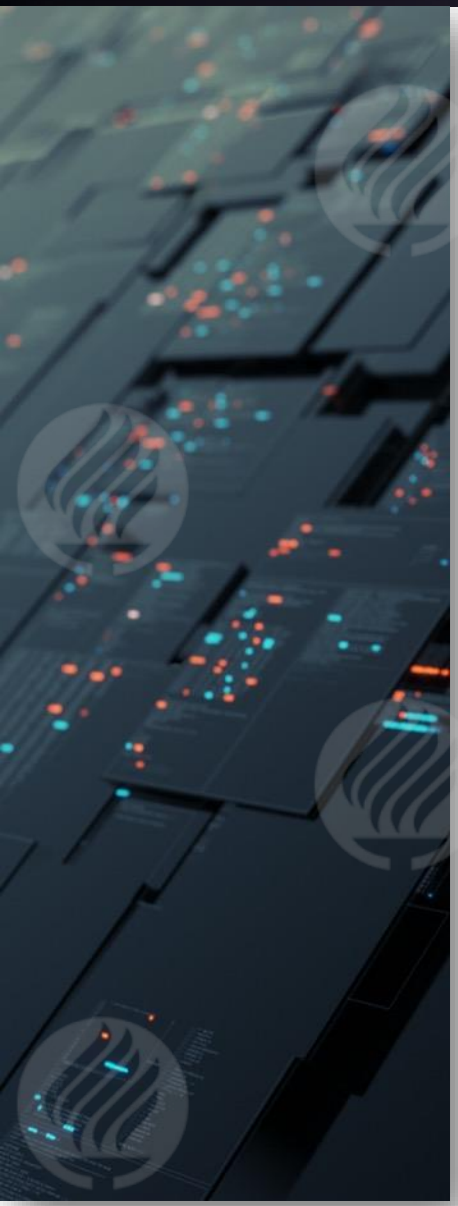
Cuando se inicia un proyecto en **Streamlit** deberá seguirse una serie de pasos recomendados:

- ✓ importar el paquete de **streamlit**
- ✓ importar otros paquetes básicos de python como pandas y/o numpy
- ✓ usando la función **st.title** colocar el título del proyecto
- ✓ **st.header** para poner un subtítulo
- ✓ **st.write** para desplegar información

Plataforma Streamlit | Tema 1

Una aplicación simple en **Streamlit** luciría así :

```
import streamlit as st

st.title('My first app')
st.header('My first app')

st.write("""
# Ejemplo Simple de una App de
Predicción

Esta app predice mis datos!

""")
```



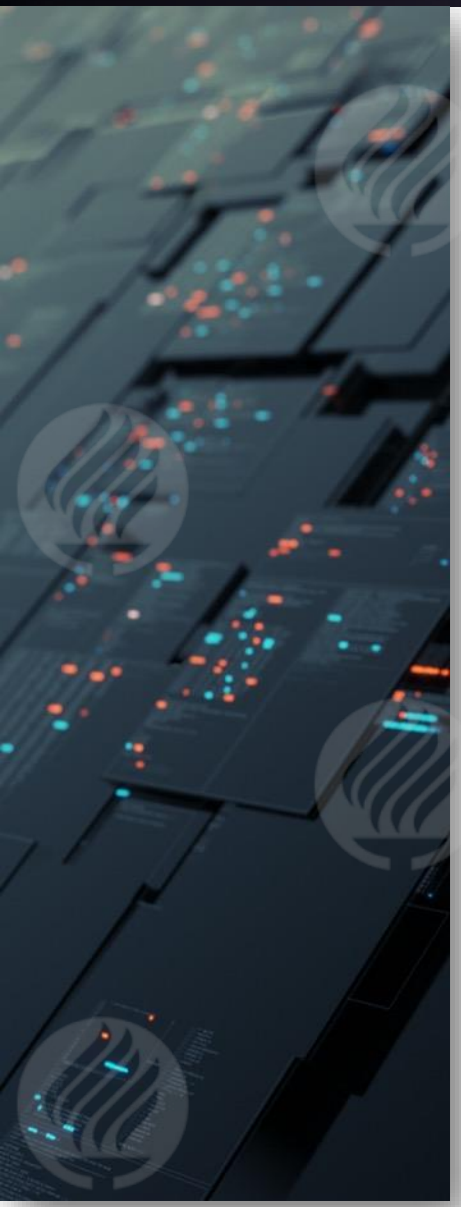
¿Cómo crear una aplicación Streamlit?

- El primero paso que debemos hacer es abrir un archivo en **Google Colab**
- Instalar el paquete **Streamlit: !pip install streamlit**

  Aparecerá un mensaje de alerta que indica que es necesario realizar una reinicialización. Es necesario dar clic en el botón que dice **"Restart Runtime"** y aparecerá un mensaje que pregunta si se desea reiniciar el sistema
- Posteriormente, vamos a instalar **ngrok**, el cual proporciona una interfaz de usuario web en tiempo real donde se puede reproducir una aplicación web simulando un servidor web en producción:

```
!wget https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-linux-amd64.zip
```

Tema 1: Plataforma Streamlit



¿Cómo crear una aplicación Streamlit?

- Ahora se descomprime el archivo en el entorno de trabajo de **Google Colab** con la siguiente instrucción:

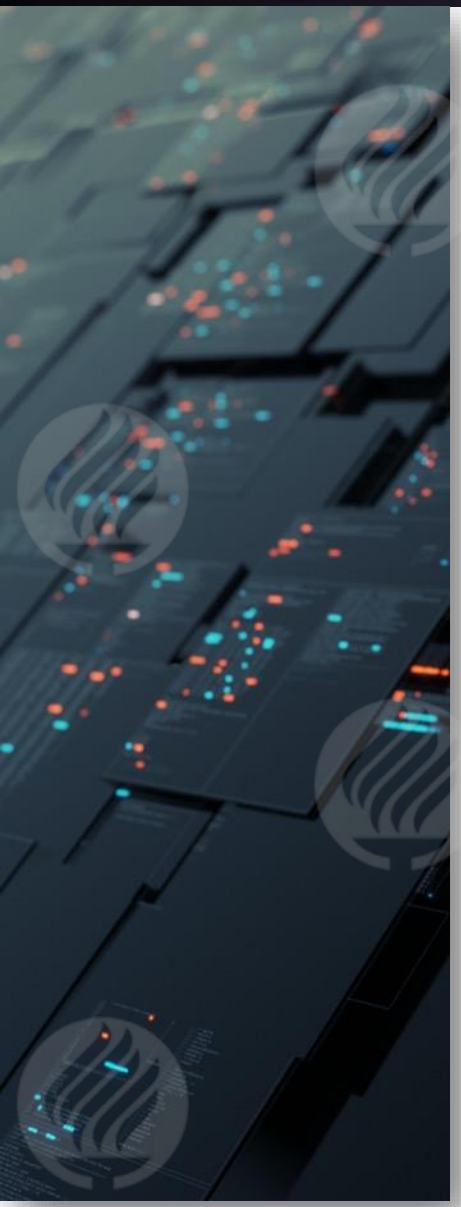
```
!unzip ngrok-stable-linux-amd64.zip
```

- Una vez que se ha descomprimido el archivo, es necesario **generar el servidor temporal** en donde se ejecutará nuestra aplicación web y en donde se le indica a Ngrok que vamos a utilizar el puerto 8501:

```
get_ipython().system_raw('./ngrok http 8501 &')
```

- Adicionalmente, es necesario incluir una instrucción que le indique a ngrok que deberá generar un túnel para que el servidor que estamos creando se pueda ejecutar en el entorno de **Google Colab**, la instrucción sería la siguiente:

```
!curl -s http://localhost:4040/api/tunnels | python3 -c \
 'import sys, json; print("Execute the next cell and the go to the
following URL: " +json.load(sys.stdin)["tunnels"][0]["public_url"])'
```



¿Cómo crear una aplicación Streamlit?

- Posteriormente, se deberá escribir el código de la aplicación en una celda de código

```
%%writefile primero.py
import streamlit as st
st.title("Mi Primera App con Streamlit")
st.header("Información sobre el Conjunto de Datos")
st.write("""
Este es un simple ejemplo de una app
""")
```

- Finalmente, se ejecuta la aplicación con la siguiente instrucción

```
!streamlit run /content/primero.py
```

- Para ejecutar el servidor temporal en donde se podrá visualizar la aplicación web, deberá dar clic en el url que aparece en la instrucción anterior.
  Execute the next cell and the go to the following URL:
  http://a9fe7569e5e1.ngrok.io
  Se cargará una pestaña en su navegador predeterminado y podrá visualizar la aplicación web que hemos creado

Educación Continua

Actividad | Aprendizaje activo

Abre el colab con la plantilla que te proporcionará el instructor para:

- Instalar **streamlit**
- Configurar y ejecutar **ngrok**
- Crear una aplicacion en **Streamlit**
- Poner el título "**Mi primer dashboard**"
- Poner header **"Streamlit App"**
- Poner el contenido **"Creado por: tunombre"**
- Visualizar la aplicación en tu navegador

Integración | Tema 1.1 Plataforma Streamlit

- El conocimiento de la **insfraestructura, recursos de hardware y software requeridos para construir y ejecutar una aplicacion en Streamlit son de vital importancia para desarrollar dashboards más sofisticados**

- Ahora que ya sabes cómo contruir y ejecutar una aplicación simple de Streamlit, **¿harás una ejercicio mental, analizando qué aplicaciones web te gustaría desarrollar en tu organización?**

Cierre: Concepto clave | Tema 1 Plataforma Streamlit

Streamlit permite crear dashboards de manera muy práctica y sencilla sin la necesidad de ser experto en lenguajes de programación propios de desarrollo web

Educación Continua

Panorámica de la sesión

1 **Sesión Síncrona 1**
Aprender

Tema 1: Plataforma Streamlit

Tema 2: Manipulación de datos

Tema 3: Interacción con componentes básicos

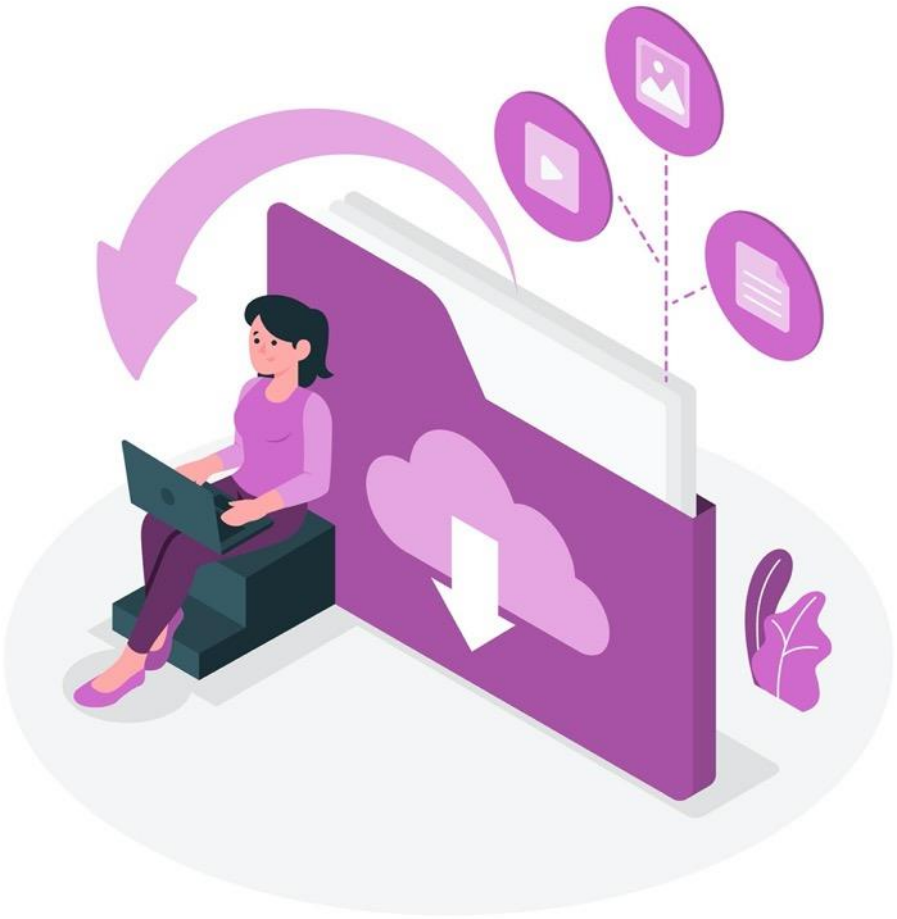
Cierre de sesión

Propósito | Tema 2 Manipulación de datos

- Las **plataformas web** regularmente consumen **grandes cantidades de información** de manera rápida e interactiva
- Con **Streamlit** podemos consultar fuentes de información, usando otros **paquetes como pandas y numpy**

Educación Continua

Streamlit con dataframes de Pandas | Tema 2.1



- En esta sección crearemos una aplicación en **Streamlit** que permita leer una **base de datos en formato .csv** y la despliegue como un dataframe

Educación Continua

Streamlit con dataframes de Pandas | Tema 2.1

Para realizar esta aplicación realizaremos los siguientes pasos

- importar la bibilioteca **streamlit**
- importar la biblioteca de **pandas**
- leer el archivo usando **read_csv()** de pandas
- desplegar usando la función **st.dataframe()** de streamlit

Educación Continua

Streamlit con dataframes de Pandas | Tema 2.1

Una aplicación básica de **Streamlit** con acceso a datos luciría así:

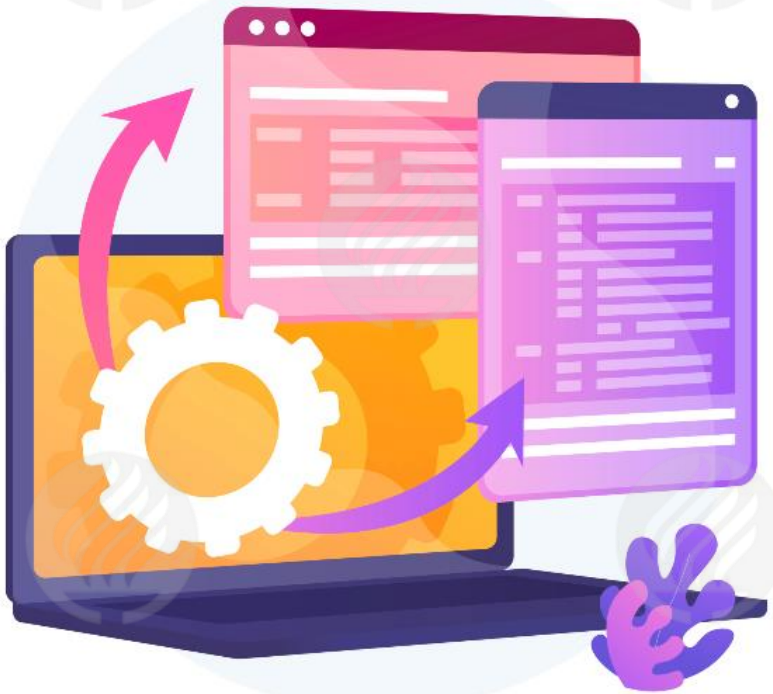
```
import pandas as pd
import streamlit as st

names_link = '/content/dataset.csv'

names_data = pd.read_csv(names_link)

# Create the title for the web app
st.title("Streamlit and pandas")

st.dataframe(names_data)
```

Actividad | Aprendizaje activo

Utiliza la misma libreta de **Colab** para:

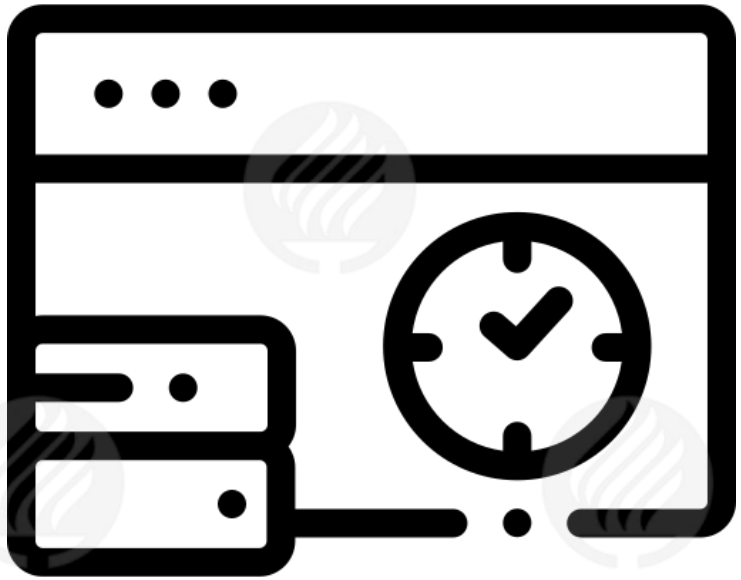
- Cargar en tu ambiente de Colab la base de datos en formato csv que te proporcionará el instructor
- Crear una aplicación en **Streamlit** y pandas para visualizar las columnas del dataset
- Ejecutar la aplicación y visualizarla en tu navegador, usando el mismo procedimiento analizado en el tema anterior

Estatuto caching | Tema 2.2



- Cuando leemos datos desde una **aplicación web**, este proceso suele **ser lento y consumir muchos recursos de ancho de banda**, dependiendo de la cantidad de datos y la velocidad de internet

Estatuto caching | Tema 2.2



- **Streamlit** tiene una propiedad que permite guardar en una memoria cache los datos leídos de una fuente de datos

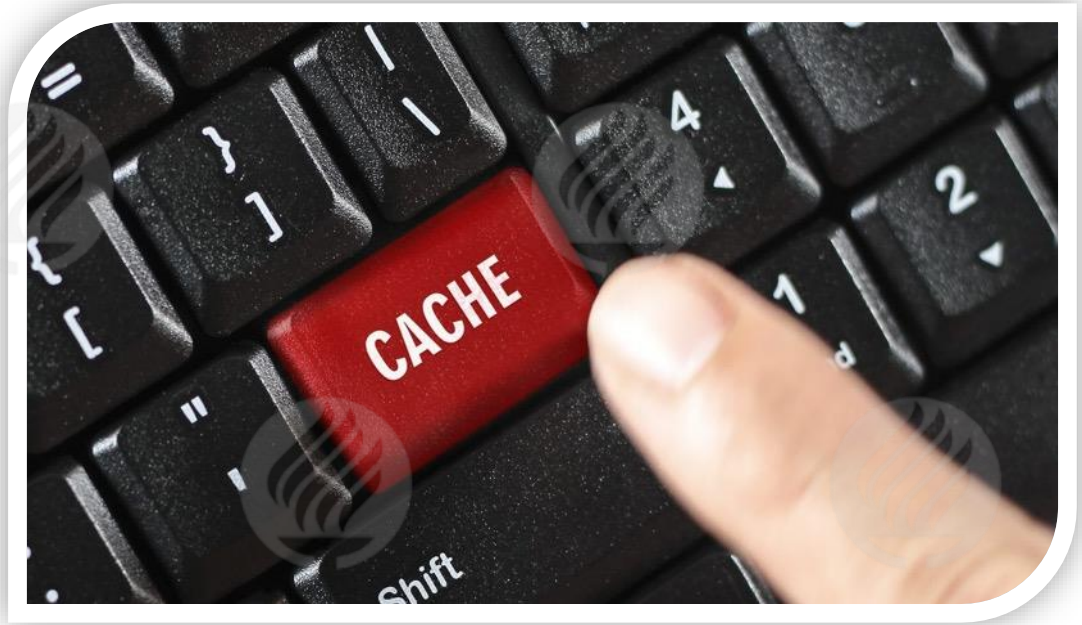
- Por tanto, si se realizada una llamada idéntica, los datos son tomados del **caché**, así se evita el tráfico y/o consumo de datos nuevamente

Estatuto caching | Tema 2.2

Streamlit solo realizará llamadas nuevas a la fuente de datos si se cumplen algunas de las siguientes condiciones:

- El **código fuente** de la función que realiza la llamada ha cambiado
- Ha cambiado la **fuente de datos**
- Se están realizando llamadas con **parámetros** diferentes

Estatuto caching | Tema 2.2



Para definir una función de Python como "**cacheable**", se utiliza el atributo **@st.cache**, como se muestra a continuación:

```
@st.cache
def load_data(nrows):
 data = pd.read_csv(DATA_URL,nrows=nrows)
 return data
```

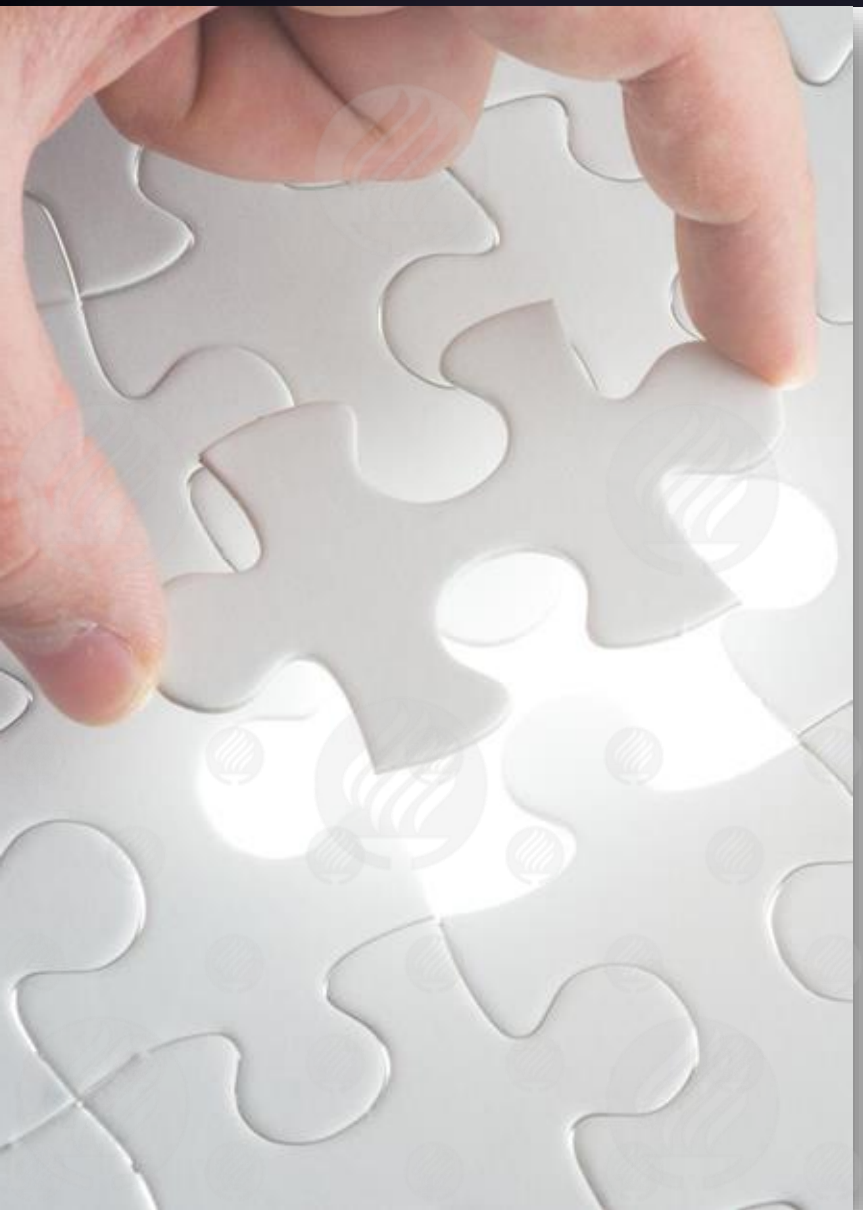
Educación Continua

Actividad | Aprendizaje activo

- Crear una dashboard en **Streamlit** que lea la fuente de datos del ejemplo anterior.
- Crear una función **load_data()** que permita recibir como parámetro el número de registros a obtener
- Agregar el atributo **@st.cache** a la función load_data()
- Poner **título y mensajes** que indiquen cuando se haya iniciado y finalizado la carga de datos

Educación Continua

Integración | Tema 2 : Manipulacion de datos



- **Streamlit** es una potente herramienta para crear aplicaciones web con acceso a datos
- **Pandas** y **Streamlit** trabajan de forma transparente para accesar a los datos y realizan el proceso de caché de forma automática con **st.cache()**
- ¿A qué aplicaciones le darías acceso de datos con caché en tu organización para minimizar tiempos de carga y reducir el tráfico de datos?

Cierre: Concepto clave | Tema 2 Manipulación de datos

Streamlit usa Pandas para la visualización de datos.

Streamlit permite el manejo de memoria caché entre llamadas a la fuente de datos para evitar tráfico innecesario

Educación Continua

Panorámica de la sesión

1 **Sesión Síncrona 1**
Aprender

Tema 1: Plataforma Streamlit

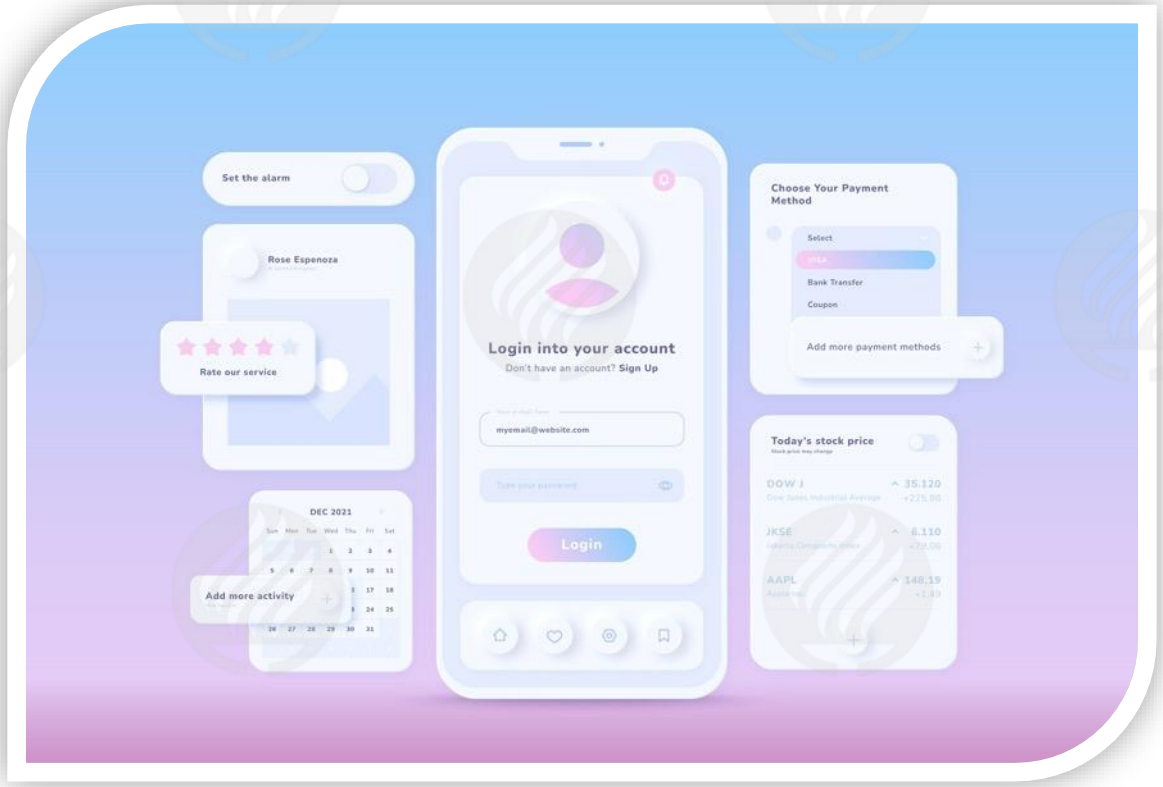
Tema 2: Manipulación de datos

Tema 3: Interacción con componentes básicos

Cierre de sesión

Educación Continua

Propósito | Tema 3 Interacción con componentes basicos



La **creación de dashboards interactivos** con la plataforma **Streamlit** y Python, usando componentes básicos como:

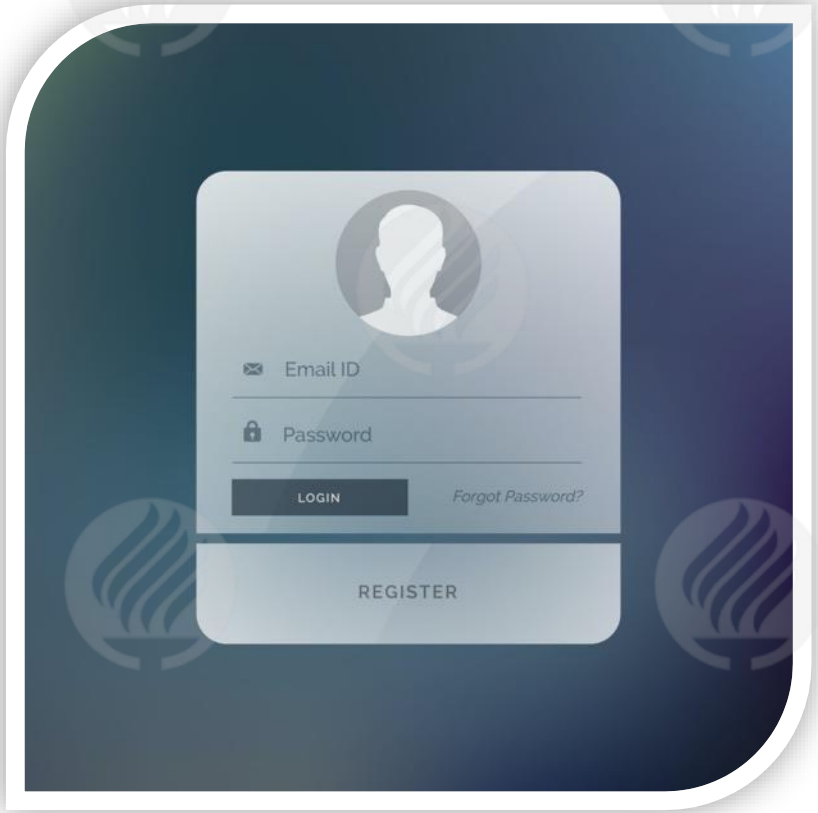
- ✓ Textbox
- ✓ Command button
- ✓ Selected box

Control text_input() | Tema 3.1

- Con **Streamlit** podemos incluir controles de texto para permitir al usuario capturar datos de entrada
- Con **st.text_input()** podemos definir un objeto para la entrada de datos, ejemplo:

```
mynombre = st.text_input(' teclea tu nombre ')
```

Control text_input() | Tema 3.1



- Con **st.text_input()** podemos validar y responder a la entrada de datos con una sentencia condicional **if**

```
myname = st.text_input('nombre :')
if (myname):
 st.write("tu nombre es : {myname}")
```

Control text_input() | Tema 3.1

- También podemos crear una función y llamarla pasando como **parámetro** el dato capturado

```
def bienvenida(nombre):
 mymensaje = 'bienvenido/a :' + nombre
 return mymensaje


myname = st.text_input('nombre :')
if (myname):
 mensaje = bienvenida(myname)
 st.write(" : {mensaje}")
```

Actividad | Aprendizaje activo

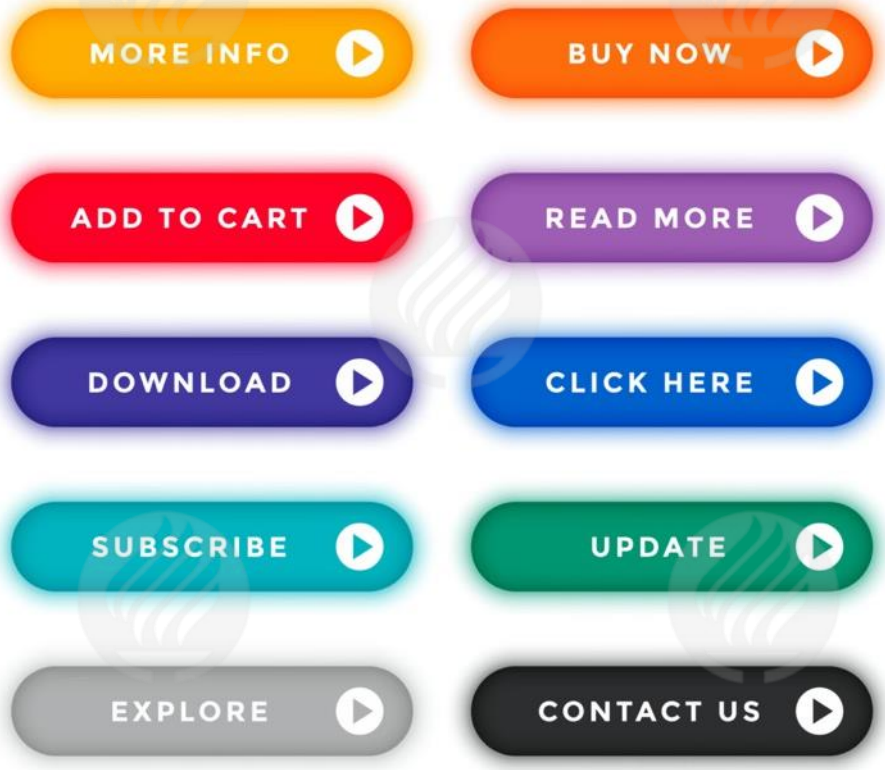
Utiliza la misma libreta de **Colab** para:

- Generar un nueva aplicación con **Streamlit** que se conecte a la base de datos csv
- Crear un control **text_input()** para capturar un nombre o las primeras letras de un nombre, ejemplo: Mary, Mar, Jimmy, Jim
- Visualizar en un **dataframe** las coincidencias encontradas del patrón de búsqueda
- Imprimir el número de coincidencias

Integración | Tema 3.1 : Control text_input()

- El **uso de widgets** como el control input_text permite crear aplicaciones web interactivas
- Ahora que ya sabes cómo hacerlo, piensa en los formularios de podrías crear en tus actividades diarias y analiza: **¿Qué aplicaciones le darías al control input_text, cuántos son necesarios, requieres controles adicionales para tener más funcionalidad?**

Control button() | Tema 3.2



- Con **Streamlit** podemos usar los controles tipo **Command Button** para permitir al usuario disparar una acción al dar click sobre un botón

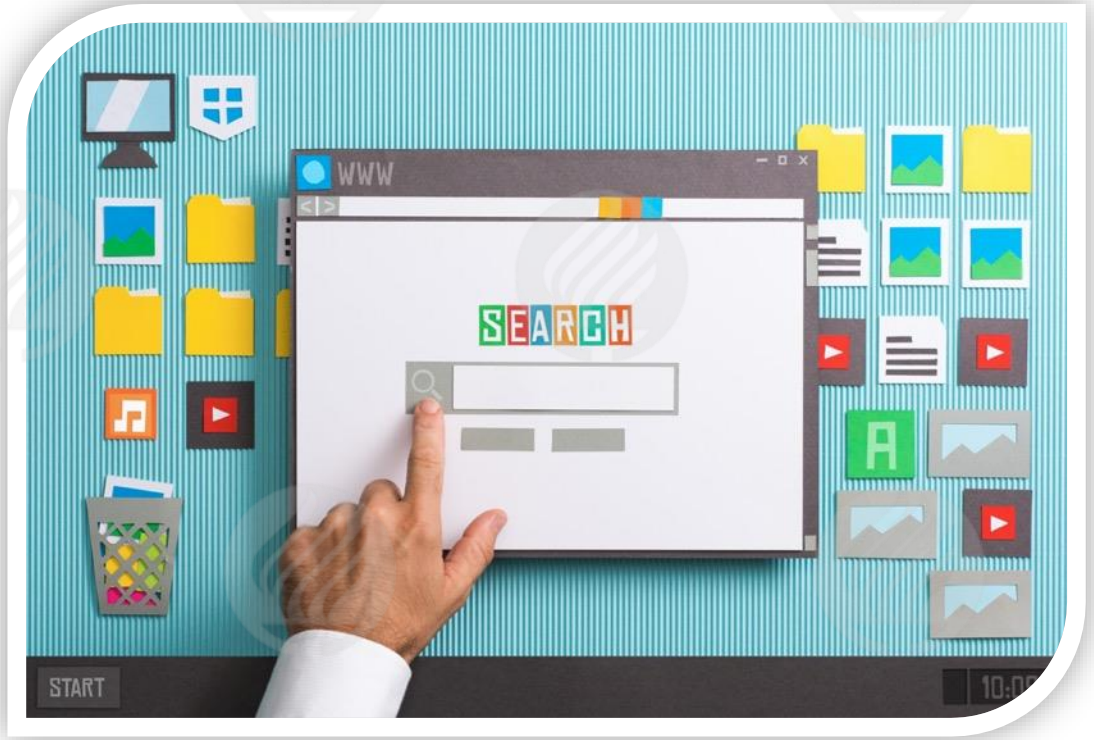
- Con **st.button()** creamos un botón de comando que responderá al click de usuario

Control button() | Tema 3.2

Regularmente **st.button()** se asocia a una sentencia condicional **if** para detectar si el usuario ha presionado click como se muestra a continuación

```
if st.button('Search'):
 st.write("Has presionado el boton Search")
```

Control button() | Tema 3.2



Es posible combinar **button** con otros controles como **text_input** para otorgar más funcionalidad, ejemplo:

```
myname = st.text_input('name :')
if st.button('Search'):
 st.write(f"search name : {myname}")
```

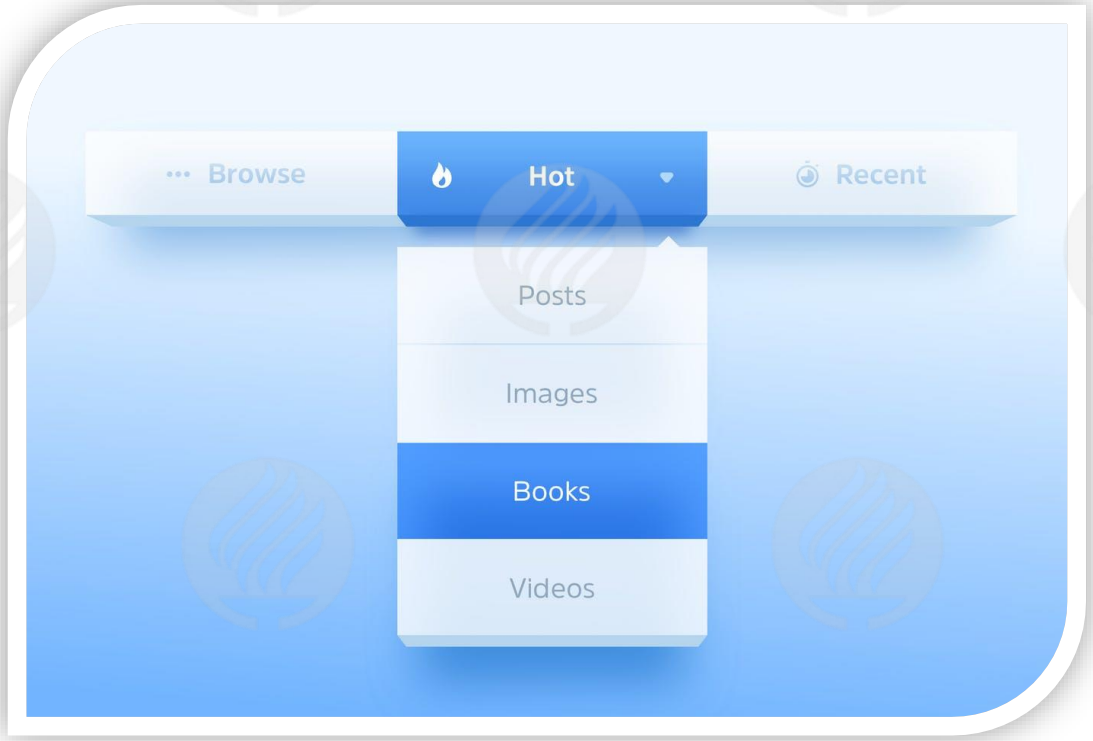
- Permitirá capturar un nombre en **myname**
- Crear un **button**
- Responder al evento click e imprimir el texto capturado

Actividad | Aprendizaje activo

Utiliza la misma libreta de **Colab** para:

- Generar un nueva aplicación **Streamlit**, se conecte a la base de datos csv
- Permitir buscar nombres de la base de datos por el campo **index**
- Definir 2 **text_input** y un **button** y buscar un conjunto de nombres por el rango capturado

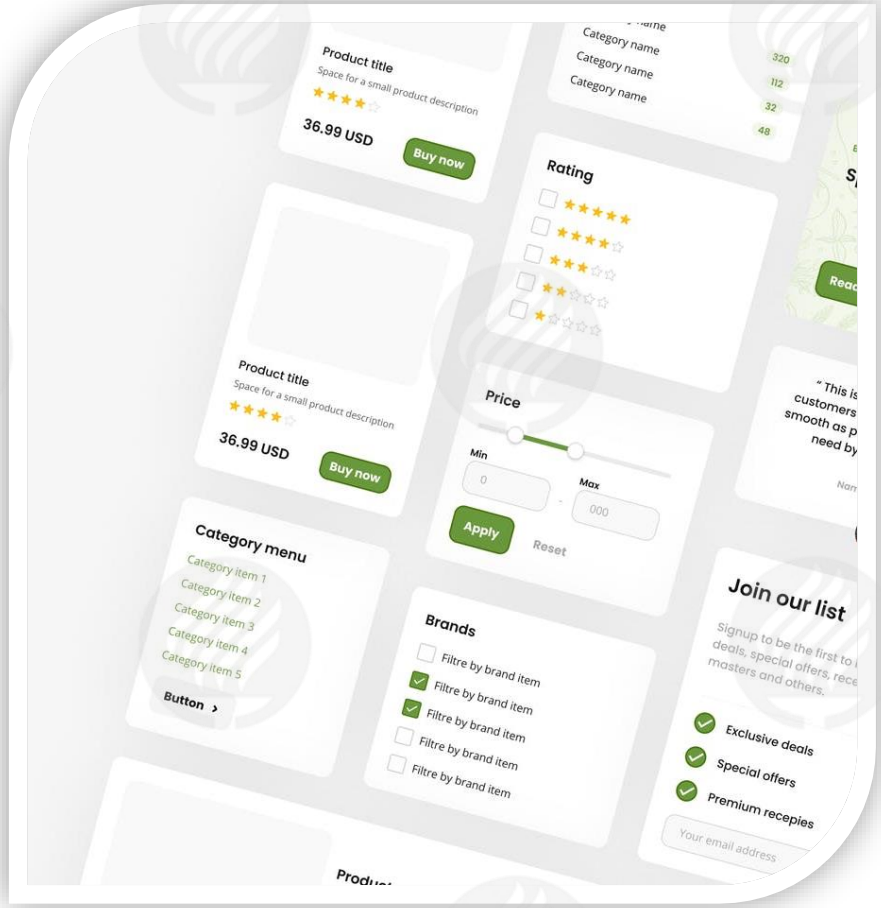
Control Selectedbox | Tema 3.3



- El control **selectbox** nos permite crear listas desplegables a partir de un conjunto de datos
- La fuente de datos puede ser una **lista**, una columna de **dataframe**, entre otros

Educación Continua

Control Selectedbox | Tema 3.3



Con **st.selectbox()** creamos una lista desplegable, indicando el mensaje a mostrar, y la fuente de datos, como se muestra a continuación.

```
selected_sex = st.selectbox("Select Sex",
data['sex'].unique())
```

Para este ejemplo data es un **dataframe** con una columna llamada sex

Control Selectedbox | Tema 3.3

Podemos usar el dato seleccionado para filtrar datos de un dataframe:

```
selected_sex = st.selectbox("Select Sex",
data['sex'].unique())
st.write(f"Selected Option: {selected_sex!r}")
filtered_data_sex = data[data['sex'] ==
selected_sex
```

- Creará una lista desplegable con la columna del **dataframe**
- Guardará en **selected_sex** el valor seleccionado
- Se crea un nuevo **dataframe** con los registros que sean iguales al valor seleccionado

Actividad | Aprendizaje activo

Utiliza la misma libreta de **Colab** para:

- Generar un nueva aplicación **Streamlit**, se conecte a la base de datos csv
- Definir un **selectedbox** y un **button**
- Asociar el control **selectedbox** al campo **sex** del dataframe
- Al dar click en el boton, deberá filtrar por el valor seleccionado indicado la cantidad de hombres o mujeres de la base de datos

Integración | Tema 3

- Los **controles de texto**, **botones de comando** y **selected box** son sumamente útiles en todos los formularios de captura de datos para realizar la acción interactiva con los usuarios de los dashboard

Cierre: Concepto clave | Tema 3

Streamlit ofrece muchos controles de usuario interactivos útiles para crear dashboards con acceso a datos de una manera muy amigable, pero a la vez potente

Educación Continua

Panorámica de la sesión

1 **Sesión Síncrona 1**
Aprender

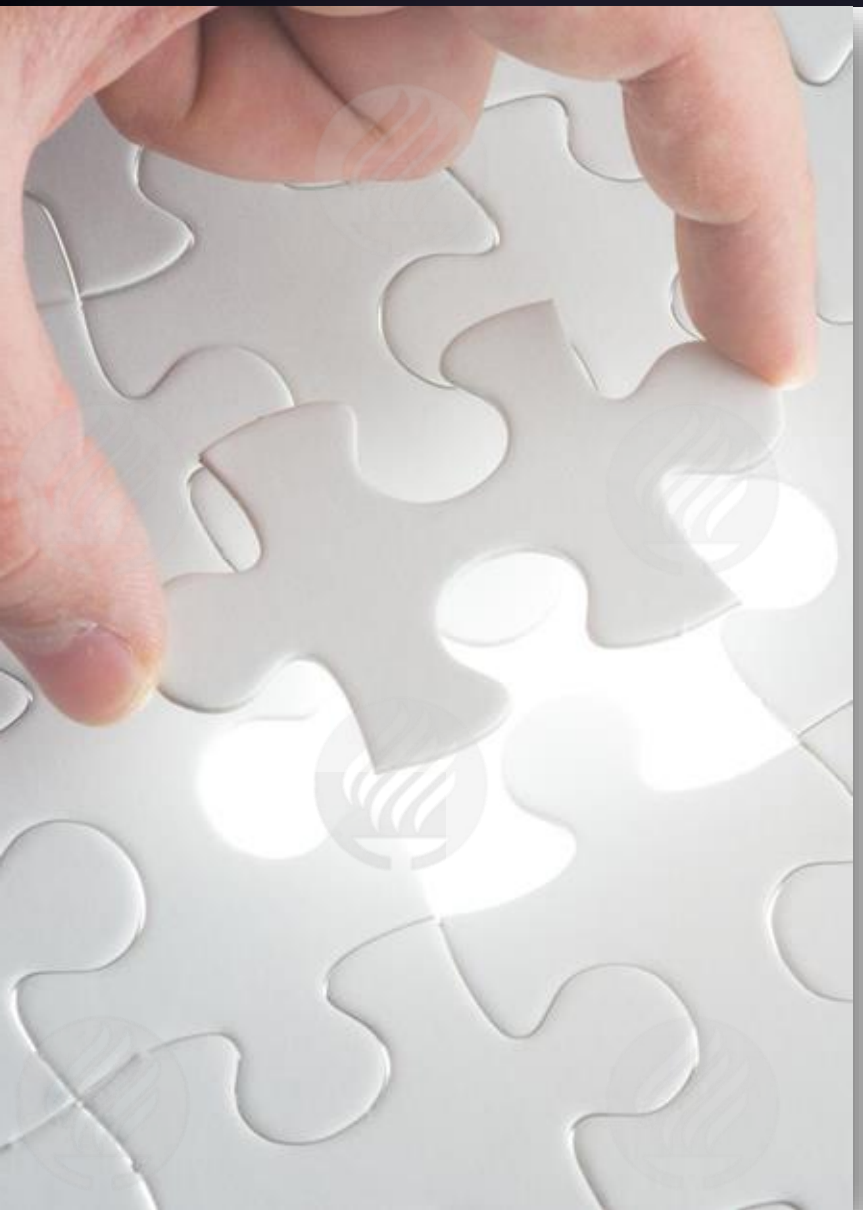
- Tema 1: Plataforma Streamlit
- Tema 2: Manipulación de datos
- Tema 3: Interacción con componentes básicos
- **Cierre de sesión**

Tecnológico de Monterrey | Educación Continua

Cierre de la sesión

Educación Continua

Cierre | Sesión Sincrónica 1 [Aprender]



- ¿Eres capaz de crear una aplicación **Streamlit** con acceso a datos y con controles de usuario simples?
- ¿Puedes filtrar y sumarizar los datos?
- ¿En qué situaciones usarías esta tecnología para resolver **situaciones reales** en tu organización?

Cierre: Concepto clave

La construcción y puesta en marcha de aplicaciones web con acceso a datos con la plataforma Streamlit es un proceso sencillo y eficaz para un científico de datos

Educación Continua

Siguientes pasos | Semana de trabajo asíncrono [Profundizar]

1. **Sesión Síncrona 1**
   Aprender
2. **Trabajo asíncrono 1**
   Profundizar | Ruta de Aprendizaje
3. **Sesión Síncrona 2**
   Preparar para Aplicar
4. **Trabajo asíncrono 2**
   Aplicar en el trabajo | Reto

- La **compresión** de la estructura básica de una aplicación interactiva con **StreamLit** con Python y los conceptos elementales para la conectividad con una fuente de datos te permitirán asimilar los temas de la sesión Profundizar

- Ya estás familiarizado con las funciones básicas de Streamlit: **estructura básica**, **manejo de datos con caché** y **controles interactivos basicos**, ahora continuarás profundizando en ellas para crear **Gráficas**, hacer menús tipo **SideBar,** uso de componentes más sofisticados como **slider**, **checkbox** y otros empleados en la programación web

Tecnológico de Monterrey | Educación Continua

Gracias | Programas LIVE