Para este taller documente sus resultados y conclusión en un **documento de entrega** (archivo de texto).

1. Instalando Dash

1. Dash es un framework basado en Python, Flask, Plotly.js para desarrollar aplicaciones web orientadas a datos. En este taller desplegaremos nuestras primeras aplicaciones basadas en Dash usando ejemplos que se pueden encontrar en el tutorial de Dash `https://dash.plotly.com/`.

2. Lance una instancia ubuntu en EC2 similar a las usadas en talleres anteriores.

3. Actualice tu base de datos de repositorios de paquetes e installe pip3 para python3.
   ```
   sudo apt update
   sudo apt install python3-pip
   pip3 --version
   ```

4. Instale paquetes de python: plotly y pandas `pip3 install plotly`
   `pip3 install pandas`

5. Instale dash para python (verifique que se instalen dash_core_components y dash_html_components)
   `pip3 install dash`

6. Instale el paquete gunicorn para python
   `pip3 install gunicorn`

7. E instale la aplicación gunicorn para linux
   `sudo apt-get install gunicorn`

8. Copie ahora los archivos appY.py que encontrará en e-aulas, con Y=0,0b,1,2,3, en un folder en su instancia.

9. En el folder donde copió los archivos ejecute
   `cp app0.py app.py`
   para crear el archivo app.py como copia de app0.py

10. Abra el archivo app.py creado y examine sus contenidos. En su **documento de entrega** explique los elementos app, server, df, fig, app.layout, así como el método main.

11. En el folder donde se encuentra el archivo app.py ejecute
    `gunicorn --workers 1 --log-level=debug --timeout 60 --bind :8050 app:server`

Explique qué hace este comando y las opciones usadas. Para esto debe buscar la documentación gunicorn. Describa qué hace el paquete gunicorn.

12. Abra el puerto 8050 de la instancia y visite el sitio IP:puerto. Compare la página que observa con el código. Revise su respuesta anterior sobre los elementos del programa. Por **slack** envíe la IP:puerto.

13. **Extra**: note que si cierra la conexión a la terminal su programa se termina. Para evitar esto puede usar `screen`. Detenga la ejecución del servidor y modifique su comando de ejecución así
    `screen gunicorn --workers 1 --log-level=debug --timeout 60 --bind :8050 app:server`
    El `screen` crea una pantalla de terminal en la que se ejecuta. Puede ahora abandonar esta terminal con CTRL+A y CTRL+D (D de detach). Para volver a la terminal desacoplada ejecute
    `screen -r`
    (r de reattach). Así, puede desacoplar la terminal de ejecución del servidor y desconectarse luego de la terminal principal. Cuando vuelva a ingresar a la terminal reacople la terminal del servidor si lo desea.

2. Más aplicaciones en Dash

1. Cancele la ejecución del servidor en curso. Siguiendo un procedimiento al anterior, cree un archivo app.py usando app1.py.

2. Lance nuevamente el servidor usando gunicorn pero con el nuevo app.py. Verifique qué muestra la aplicación en el navegador.

3. Examine el archivo app.py y explique en su **documento de entrega** los elementos app.layout, @app.callback, la función update_output_div, y la relación entre estos 3 elementos.

4. Cancele la ejecución del servidor en curso. Siguiendo un procedimiento al anterior, cree un archivo app.py usando app2.py.

5. Lance nuevamente el servidor usando gunicorn pero con el nuevo app.py. Verifique qué muestra la aplicación en el navegador.

6. Examine el archivo app.py y explique en su **documento de entrega** los elementos app.layout (Graph, Slider), @app.callback (Input, Output), la función update_figure, y la relación entre estos 3 elementos.

7. Cancele la ejecución del servidor en curso. Siguiendo un procedimiento al anterior, cree un archivo app.py usando app3.py.

8. Lance nuevamente el servidor usando gunicorn pero con el nuevo app.py. Verifique qué muestra la aplicación en el navegador.

9. Por **slack** envíe la IP:puerto con esta última aplicación en ejecución.