**Reporte del Trabajo Realizado- VISUALIZACION DE FORMACIONES**

**1. Objetivos de la Visualización**

El objetivo principal del dashboard es proporcionar una herramienta interactiva y visualmente atractiva para analizar las formaciones ofrecidas. Este dashboard permite a los usuarios:

* Visualizar el porcentaje de inscripciones alcanzadas en comparación con las plazas disponibles.
* Examinar la cantidad de formaciones y los ingresos generados por mes.
* Comparar las plazas meta con las inscripciones realizadas para cada formación.
* Consultar una tabla detallada de todas las formaciones, incluyendo información clave como el nombre, las plazas disponibles y las inscripciones.

**2. Tipos de Gráficos Seleccionados y su Función**

* **Medidor de Porcentaje de Inscripciones Alcanzado (Gauge Chart)**: Este gráfico muestra el porcentaje de inscripciones alcanzado en relación con las plazas disponibles. Es útil para evaluar rápidamente el rendimiento general de las formaciones.
* **Gráfico de Barras y Líneas (Cantidad Formaciones y Suma de Ingresos por Mes)**: Este gráfico combina barras para mostrar la cantidad de formaciones ofrecidas cada mes y una línea para representar los ingresos generados. Ayuda a identificar tendencias en la oferta y la demanda a lo largo del tiempo.
* **Gráfico de Barras Agrupadas (Plazas Meta y Inscripciones por ID)**: Este gráfico compara las plazas meta con las inscripciones realizadas para cada formación, lo que permite evaluar la efectividad de cada curso en alcanzar sus objetivos.
* **Tabla de Formaciones**: Una tabla interactiva que proporciona un resumen detallado de todas las formaciones, incluyendo información como el nombre del curso, las plazas disponibles y las inscripciones realizadas. Es útil para obtener detalles específicos rápidamente.

**3. Librerías Utilizadas y Estructura de la Misma**

**Librerías Utilizadas**:

* **Dash**: Utilizada para crear el dashboard interactivo. Dash facilita la creación de aplicaciones web de visualización de datos.
* **Dash Bootstrap Components**: Proporciona componentes de Bootstrap para mejorar el diseño y la presentación del dashboard.
* **Pandas**: Utilizada para la manipulación y el análisis de datos.
* **Plotly**: Utilizada para la creación de gráficos interactivos.
* **Configparser**: Para la lectura del archivo de configuración.
* **Pymysql**: Para la conexión y ejecución de consultas en la base de datos MySQL.
* **Flask**: Servidor web utilizado por Dash para desplegar la aplicación.

**Estructura**:

* **app.py**: Contiene la lógica principal del dashboard, incluyendo la configuración, conexión a la base de datos, preprocesamiento de datos, diseño del layout, y los callbacks para actualizar los gráficos.
* **config.ini**: Archivo de configuración que almacena las credenciales de la base de datos.
* **Dockerfile**: Define cómo se construye la imagen Docker para desplegar la aplicación localmente.
* **requirements.txt**: Lista de dependencias necesarias para ejecutar la aplicación.

**Uso de AWS**: La base de datos se aloja en AWS RDS, lo que proporciona una solución escalable y gestionada para bases de datos en la nube. El acceso a la base de datos se configura en el archivo config.ini, permitiendo una conexión segura y eficiente desde la aplicación Dash.

**4. Principios para el Uso de Dockerfile para Acceso Local**

**Dockerfile**: El Dockerfile se utiliza para crear una imagen de Docker que contiene todo el entorno necesario para ejecutar la aplicación. Esto incluye el sistema operativo, las librerías y las dependencias de Python.

**Estructura del Dockerfile**:

* **FROM**: Define la imagen base, generalmente una imagen ligera de Python.
* **COPY**: Copia los archivos de la aplicación en la imagen Docker.
* **RUN**: Ejecuta comandos para instalar las dependencias, utilizando pip install con el archivo requirements.txt.
* **CMD**: Define el comando para ejecutar la aplicación, generalmente python app.py.

**Ventajas del Uso de Docker**:

* **Consistencia**: Asegura que la aplicación se ejecutará de la misma manera en cualquier entorno.
* **Aislamiento**: Cada contenedor Docker se ejecuta en su propio entorno aislado, evitando conflictos con otras aplicaciones.
* **Portabilidad**: Las imágenes Docker se pueden transferir y ejecutar en cualquier sistema compatible con Docker, facilitando la implementación y el escalado.

**Conclusión**

Este proyecto ha integrado una variedad de tecnologías y herramientas para crear un dashboard interactivo y visualmente atractivo. Al utilizar Dash para la visualización, pandas para el procesamiento de datos, y Docker para la implementación local, se ha logrado una solución robusta y escalable. La base de datos alojada en AWS RDS asegura un acceso rápido y seguro a los datos, facilitando el análisis y la toma de decisiones basada en datos.