Informe de Funcionalidad y Diseño

1. **Funcionalidad**

El proyecto está diseñado para interactuar con la API de Polygon.io, obtener datos financieros, guardarlos en una base de datos SQLite y visualizarlos en gráficos. A continuación, se describe la funcionalidad de cada clase:

- **Clase Api:** Se encarga de la comunicación con la API de Polygon.io. Su método *consultar_datos* permite obtener datos para un ticker y un rango de fechas en particular.
- **Clase BaseDeDatos:** Gestiona la interacción con la base de datos SQLite. Permite crear las tablas, insertar datos y consultar los mismos para su actualización/visualización.
- **Clase App:** Es la clase principal que coordina las acciones de las otras clases. Realiza la validación de fechas, la agrupación de fechas faltantes para la consulta a la API, la actualización de datos en la base de datos y la visualización de gráficos.
- **Clase Menu:** Se encarga de mostrar las opciones del menú al usuario, capturar sus elecciones y ejecutar las acciones correspondientes a cada opción.
- Clase **StreamlitApp**: Incluye las funcionalidades necesarias para crear una interfaz web interactiva utilizando Streamlit, permitiendo a los usuarios visualizar gráficos y realizar consultas a través de una aplicación web.

2. **Diseño**

El proyecto sigue un diseño modular, donde cada clase tiene una responsabilidad específica. Este enfoque facilita el mantenimiento y la escalabilidad del proyecto.

3. **Flujo de Trabajo**

1. El usuario inicia la aplicación ejecutando el script *tpfinal.py* o *streamlit_app.py* según la interfaz deseada.
2. Se instancia la clase *App*, que crea una instancia de *BaseDeDatos* y *Api*.
3. Se muestra el menú principal al usuario (en consola o en la interfaz web).
4. El usuario selecciona una opción.
5. La clase *App* ejecuta las acciones correspondientes a la opción seleccionada, interactuando con *BaseDeDatos* y *Api*.
6. Se muestra el resultado al usuario.
7. En caso de la aplicación de consola, se vuelve al menú principal hasta que el usuario seleccione la opción de salir.

4. **Consideraciones Adicionales**

- El proyecto utiliza la librería *requests* para interactuar con la API de Polygon.io.
- La librería *sqlite3* se utiliza para trabajar con la base de datos SQLite.
- La librería *matplotlib* y *pandas* se utilizan para la generación de gráficos.
- La librería *Streamlit* se utiliza para crear la interfaz web interactiva.
- El proyecto guarda las credenciales de la API en un archivo .env para facilitar la configuración y mantener la seguridad de la clave de acceso a la API.

5. **Conclusiones**

El proyecto presenta un diseño modular, funcionalidad completa y un flujo de trabajo intuitivo. El uso de clases contribuye a la claridad, el mantenimiento y la escalabilidad del proyecto. La integración de Streamlit proporciona una experiencia de usuario mejorada, permitiendo la visualización interactiva de la información. La documentación proporcionada en los comentarios y en este informe facilita la comprensión y el uso del proyecto.