

Realizado por: Ángel Gallego Felipe

# **HITO 2**

# **1ER TRIMESTRE**

# **S.G.E**

Campus FP Getafe  
2º Desarrollo de aplicaciones  
multiplataforma"

20/11/2024

## CONTENIDO

ESPECIFICACIONES.....	3
INTRODUCCIÓN .....	4
Descripción de la Interfaz y Conexión .....	5
1. Interfaz de Usuario en Tkinter:.....	5
1.1 Pantalla Principal: .....	5
1.2 Filtros y Consultas: .....	5
1.3 Gráficos:.....	6
1.4 Exportación a Excel:.....	7
2. Conexión a la Base de Datos MySQL .....	8
2.1 Establecimiento de la Conexión: .....	8
Operaciones CRUD .....	8
1. Crear (Insertar datos) .....	9
2. Leer (Consultar datos) .....	10
3. Actualizar (Modificar datos) .....	10
4. Eliminar (Eliminar datos) .....	10
Consultas y Ordenación de Datos: Explicación y Ejemplos de Uso .....	11
1. Consultas Básicas de Selección .....	11
2. Consultas con Filtros .....	11
3. Consultas con Ordenación de Datos .....	12
Exportación de Resultados a Excel: Descripción del Proceso.....	12
1. Proceso de Exportación.....	12
Visualización de Datos en Gráficos.....	13
1. Tipos de Gráficos Utilizados .....	13
PARTES DEL CODIGO MAS RELEVANTE.....	13
BIBLIOGRAFIA.....	20

## ESPECIFICACIONES

Lista de ejercicios ajustada:

- Interfaz de Usuario (Tkinter)

El módulo debe incluir una interfaz de usuario desarrollada con Tkinter, organizada para:

- o Permitir la visualización y manipulación de los datos relacionados con el consumo de alcohol y los indicadores de salud.

- o Incorporar elementos como menús, botones y campos de entrada necesarios para realizar las acciones solicitadas por el cliente, el Dr. Fernando.

- o La interfaz puede implementarse en una única pantalla o en varias, pero debe reflejar claramente cada una de las funcionalidades especificadas.

Opcional: Uso de menús o elementos avanzados no vistos en clase para mejorar la experiencia de usuario.

- Conexión a la Base de Datos (MySQL) El programa debe conectarse a la base de datos de la clínica (creada previamente con el script SQL proporcionado) para realizar operaciones CRUD en la tabla de encuestas. Específicamente:

- o Crear: Agregar registros de nuevas encuestas de pacientes.

- o Leer: Visualizar registros, mostrando datos como edad, consumo de bebidas, problemas de salud, entre otros.

- o Actualizar: Modificar registros existentes (por ejemplo, corregir la cantidad de bebidas o los problemas de salud reportados).

- o Eliminar: Borrar registros, si es necesario, con un aviso de confirmación para evitar eliminaciones accidentales.

- Consultas y Ordenación

Se debe permitir:

- o Realizar consultas que ordenen los resultados por cualquier campo (edad, sexo, consumo semanal, problemas de salud, etc.).

- o Facilitar la aplicación de filtros condicionales para visualizar subgrupos específicos, por ejemplo:

- Encuestados con alta frecuencia de consumo de alcohol.

- Personas que han perdido el control por el consumo de alcohol en más de 3 ocasiones.

## INTRODUCCIÓN

El consumo de alcohol es una de las principales conductas de riesgo relacionadas con problemas de salud a nivel mundial. Su abuso está estrechamente vinculado a una serie de enfermedades crónicas, trastornos mentales, accidentes, y muertes prematuras. El análisis de datos sobre el consumo de alcohol y su impacto en la salud se ha convertido en un área crucial para la toma de decisiones en políticas públicas, salud preventiva, y promoción de hábitos saludables. Este trabajo se enmarca dentro del ámbito del análisis de datos relacionados con el consumo de alcohol y su correlato con la salud, con el objetivo de construir una herramienta que facilite la visualización y comprensión de estos datos a través de una plataforma interactiva y accesible.

La importancia del análisis de este tipo de datos radica en la necesidad de comprender patrones de consumo y sus efectos en diferentes grupos demográficos, tales como edad, género, nivel socioeconómico y ubicación geográfica. Los estudios previos han mostrado que el consumo excesivo de alcohol no solo incrementa la incidencia de enfermedades hepáticas, cardiovasculares y psicológicas, sino que también afecta negativamente el bienestar social y económico de las personas. Por lo tanto, es fundamental contar con herramientas que permitan analizar estos datos de forma eficiente y extraer conclusiones que puedan influir en la formulación de estrategias para la reducción de riesgos asociados al consumo de alcohol.

Este trabajo tiene como objetivo desarrollar una aplicación que permita analizar de manera integral datos sobre el consumo de alcohol y su impacto en la salud. La herramienta proporcionará un sistema de gestión de datos y visualización que permitirá la realización de consultas avanzadas, filtrado de información según distintos parámetros, y la creación de gráficos para representar las tendencias y correlaciones de consumo de alcohol con indicadores de salud. Además, se implementarán funciones de exportación de datos a formatos accesibles como Excel, facilitando la elaboración de reportes y análisis externos.

El proyecto utilizará una base de datos MySQL para almacenar los datos de consumo y salud, con operaciones CRUD (crear, leer, actualizar, eliminar) para la manipulación de la información. La aplicación contará con una interfaz gráfica de usuario (GUI) desarrollada con Tkinter, que proporcionará una experiencia de usuario intuitiva y amigable. Además, se integrarán gráficos interactivos que permitirán visualizar la evolución del consumo de alcohol y sus efectos en la salud de manera clara y comprensible, lo que facilitará la toma de decisiones para investigadores, profesionales de la salud y responsables de políticas públicas.

Para contextualizar el trabajo, es importante señalar que el análisis de datos de salud y consumo de alcohol ha sido objeto de estudios internacionales como el informe de la Organización Mundial de la Salud (OMS) sobre el consumo de alcohol y su impacto en la salud pública. Dichos informes destacan la necesidad de herramientas tecnológicas que permitan gestionar, analizar y visualizar datos de manera eficiente para apoyar en la prevención y en la adopción de hábitos saludables. De este modo, el objetivo principal del proyecto es contribuir con una herramienta que facilite el análisis de estos datos y proporcione una base sólida para futuras investigaciones y acciones relacionadas con la salud pública.

## Descripción de la Interfaz y Conexión

El sistema desarrollado para el análisis de datos sobre consumo de alcohol y su impacto en la salud presenta una interfaz gráfica de usuario (GUI) intuitiva, diseñada con Tkinter, una biblioteca estándar de Python para el desarrollo de interfaces gráficas. Esta interfaz permite al usuario interactuar con el sistema de manera sencilla, acceder a diferentes funcionalidades como la visualización de datos, la creación de consultas, la exportación de datos a Excel, y la generación de gráficos representativos.

### 1. Interfaz de Usuario en Tkinter:

La interfaz de usuario fue diseñada con el objetivo de ser fácil de usar, visualmente clara y funcional. A continuación, se describen las principales secciones de la interfaz:

#### 1.1 Pantalla Principal:

La pantalla principal está compuesta por un menú de navegación que permite al usuario seleccionar diferentes acciones, como visualizar los datos de consumo de alcohol, realizar filtros o consultas, o generar reportes en formato Excel.

Los datos se presentan en una tabla organizada, donde cada fila corresponde a un registro de consumo, y las columnas incluyen detalles como el tipo de bebida, la cantidad consumida, la edad del consumidor, el género, y las afecciones de salud relacionadas.

Aquí podemos ver una imagen sacada directamente de la app de A.G.F en la que se ve la pantalla principal nada más ejecutar la app.

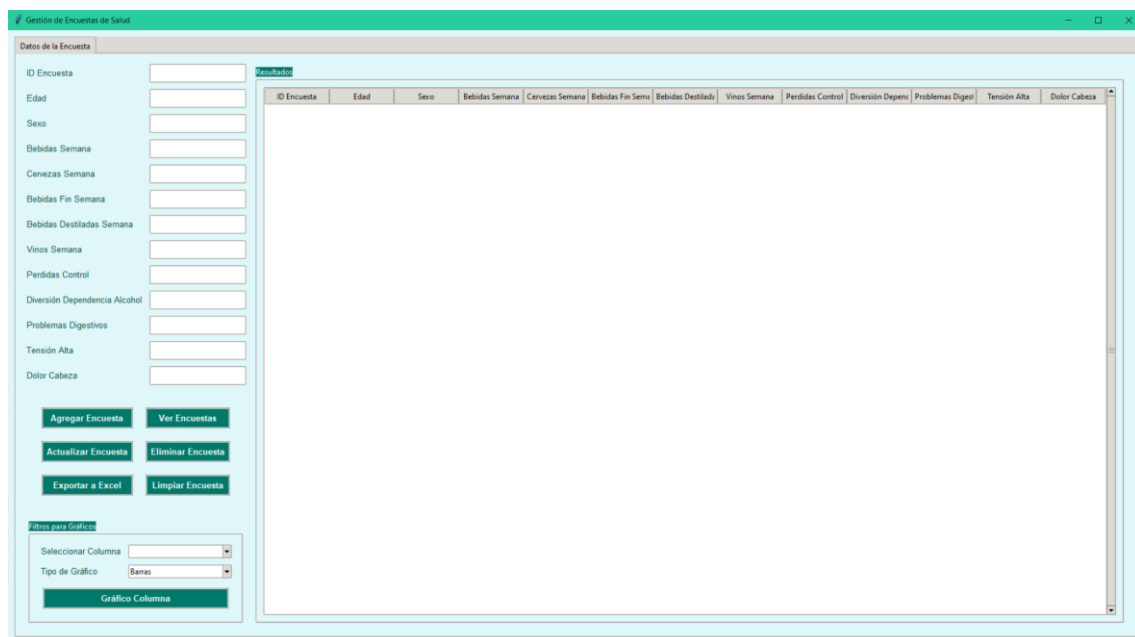


Imagen app AGF

#### 1.2 Filtros y Consultas:

La aplicación nos permite ordenar los datos volcados en forma de tabla en el cuadrante “resultados” situado en la parte derecha de la app, en la imagen anterior, aparece vacío ya que no se están mostrando en este momento.

Podemos ordenar de mayor a menor o de menor a mayor cualquiera de las columnas de la tabla con solo clicar en la parte superior o encabezado de la propia tabla, antes hay que mostrar las encuestas pulsando el botón “ver encuestas”.

Resultados

ID Encuesta	Edad	Sexo	Bebidas Semana	Cervezas Semana	Bebidas Fin Seme	Bebidas Destilad	Vinos Semana	Perdidas Control	Diversion Depenc	Problemas Digest	Tensión Alta	Dolor Cabeza
1	12	Hombre	2	5	5	0	3	1	No	Sí	No	lo
2	55	Mujer	6	4	5	1	2	3	Sí	Sí	No	Alguna vez
3	1	Hombre	0	0	0	0	0	0	No	No	No	asdas
4	20	Hombre	0	0	0	0	0	0	No	No	No	Nunca
5	21	Hombre	20	15	10	5	0	12	Sí	No	No	Alguna vez
6	20	Hombre	1	12	45	12	23	45	No	No	No	Alguna vez
7	19	Hombre	2	0	2	2	0	3	No	No	No	Nunca
8	19	Hombre	3	5	5	5	0	1	Sí	No	No lo se	Muy a menudo
9	22	Hombre	0	0	0	0	0	0	No	No	No lo se	Alguna vez
10	19	Hombre	2	10	5	1	5	2	No	No	No	Alguna vez
11	19	Hombre	83	0	75	27	56	33	Sí	Sí	Sí	Muy a menudo
12	21	Hombre	0	0	0	0	0	0	No	No	No	Alguna vez
13	19	Hombre	5	5	5	5	0	1	Sí	No	No lo se	Alguna vez
14	24	Hombre	25	18	10	7	0	365	No	No	No	Alguna vez
15	38	Mujer	41	5	25	10	10	365	Sí	Sí	Sí	Muy a menudo
16	53	Mujer	0	0	0	0	0	0	No	No	Sí	Alguna vez
17	19	Hombre	4	0	1	1	1	1	No	No	No	Nunca
18	20	Hombre	3	7	7	2	0	0	Sí	Sí	No lo se	Nunca
19	20	Hombre	3	2	5	3	0	1	No	No	No	Alguna vez
20	21	Mujer	2	0	2	0	2	0	No	No	No	A menudo
21	19	Hombre	20	20	10	7	0	3	No	No	No lo se	Alguna vez
22	20	Hombre	1	0	0	1	1	0	No	No	No	Nunca
23	19	Hombre	0	0	0	0	0	0	No	No	Sí	Alguna vez
24	58	Hombre	12	6	7	4	3	2	No	Sí	Sí	A menudo
25	47	Hombre	0	0	3	3	0	2	No	No	No	Nunca
26	19	Hombre	1	0	5	1	0	5	Sí	No	No	Nunca
27	37	Mujer	10	5	6	3	3	3	Sí	Sí	Sí	Muy a menudo
28	50	Hombre	50	35	65	26	1	100	Sí	Sí	Sí	Muy a menudo
29	19	Hombre	0	0	0	0	0	0	No	Sí	No	Alguna vez
30	19	Hombre	23	0	10	11	12	69	Sí	No	No	A menudo
31	19	Hombre	0	0	0	0	0	3	No	No	No lo se	Nunca
32	60	Mujer	3	2	3	0	1	0	No	Sí	Sí	A menudo
33	25	Hombre	7	2	7	5	0	0	No	No	No	A menudo
34	45	Mujer	10	3	5	3	0	1	Sí	Sí	No lo se	Muy a menudo
35	53	Mujer	3	3	3	0	1	0	No	Sí	No	Muy a menudo
36	55	Hombre	11	5	4	0	2	0	No	No	No	Nunca
37	18	Hombre	0	0	0	0	0	0	No	No	No	Nunca
38	55	Mujer	0	0	4	0	4	0	No	Sí	No	Muy a menudo
39	56	Hombre	3	2	2	0	3	2	Sí	No	Sí	Nunca
40	61	Hombre	6	2	4	1	3	0	No	No	No	Nunca
41	37	Hombre	4	1	2	0	3	0	No	No	No	Nunca
42	56	Mujer	2	5	4	0	2	1	No	Sí	Sí	Alguna vez

Imagen app AGF

### 1.3 Gráficos:

Se integraron gráficos interactivos para visualizar la relación entre el consumo de alcohol y distintos indicadores de salud, como enfermedades hepáticas, cardiovasculares, etc. Los gráficos se generan en tiempo real con los datos filtrados y ordenados según las preferencias del usuario.

Para la representación gráfica se utilizó la librería matplotlib, que permite crear gráficos de barras, líneas y pasteles que muestran las tendencias y patrones del consumo y sus efectos en la salud.

Los gráficos se personalizan en esta parte de la interfaz.

Filtros para Gráficos

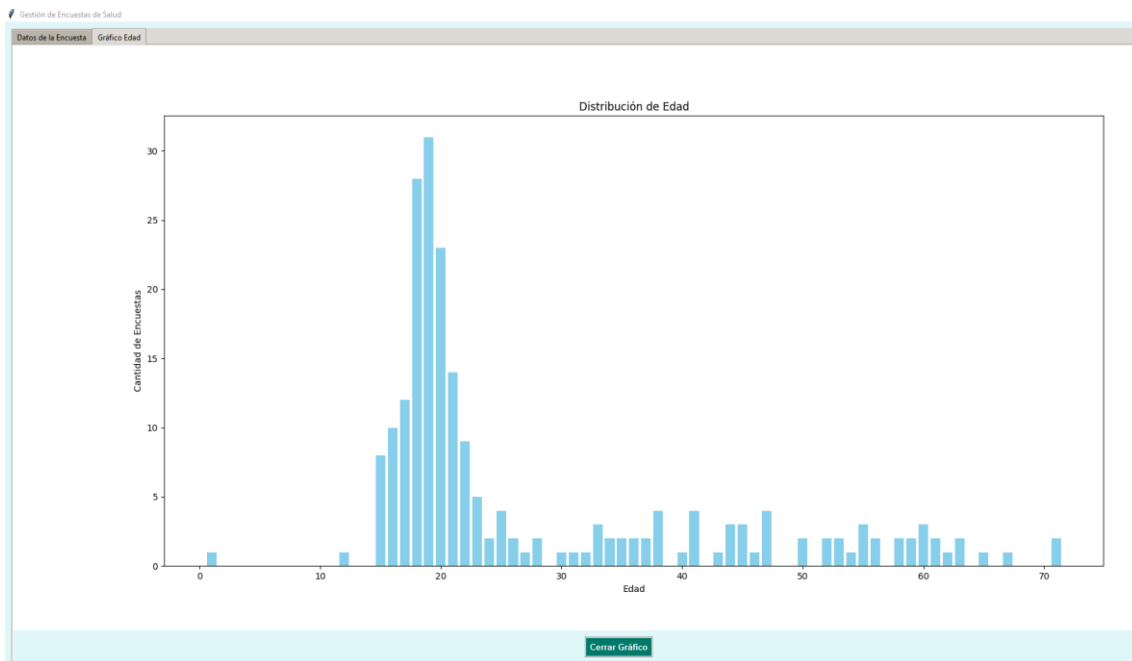
Seleccionar Columna

Tipo de Gráfico

**Gráfico Columna**

Imagen app AGF

Aquí podemos ver uno de los ejemplos de los gráficos que se generan en la pestaña de gráficos.



1 Imagen app AGF

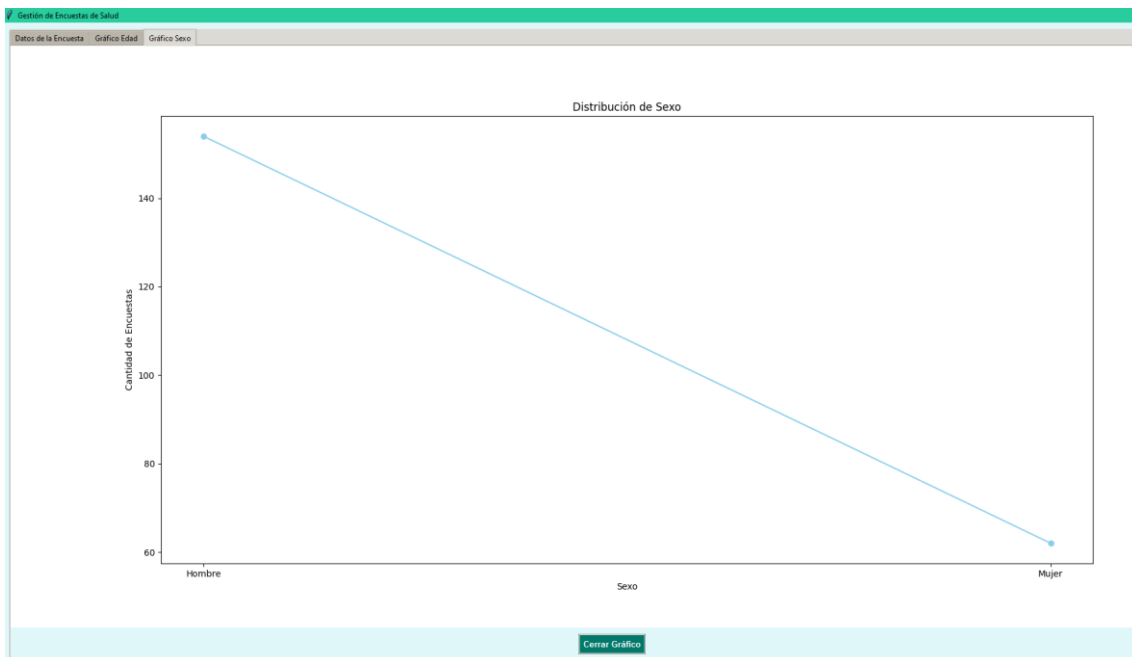


Imagen app AGF

#### 1.4 Exportación a Excel:

Un botón en la interfaz permite al usuario exportar los resultados de la consulta o los datos filtrados a un archivo Excel. Esta funcionalidad se implementó utilizando la librería openpyxl o pandas, permitiendo generar un archivo estructurado para su posterior análisis.

La exportación a Excel se realiza en este botón de la interfaz.

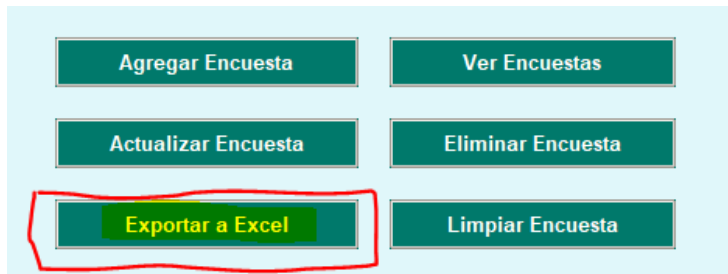


Imagen app AGF

## 2. Conexión a la Base de Datos MySQL

La aplicación se conecta a una base de datos MySQL que almacena la información sobre el consumo de alcohol y los datos relacionados con la salud. La conexión se establece utilizando MySQL Connector en Python, que permite realizar operaciones CRUD sobre la base de datos.

### 2.1 Establecimiento de la Conexión:

La conexión a la base de datos se realiza mediante la configuración de los parámetros de conexión, como el host, usuario, contraseña y nombre de la base de datos. A través de este proceso, la aplicación puede acceder y manipular los datos almacenados.

## Operaciones CRUD

Las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) son fundamentales en cualquier sistema que maneje datos. En este proyecto, se implementaron estas operaciones sobre una base de datos MySQL, que almacena la información relacionada con el consumo de alcohol y su impacto en la salud. A continuación, se detallan las operaciones implementadas y cómo se desarrollaron en la aplicación.

En esta imagen tenemos los botones para realizar esta serie de operaciones en la interfaz.

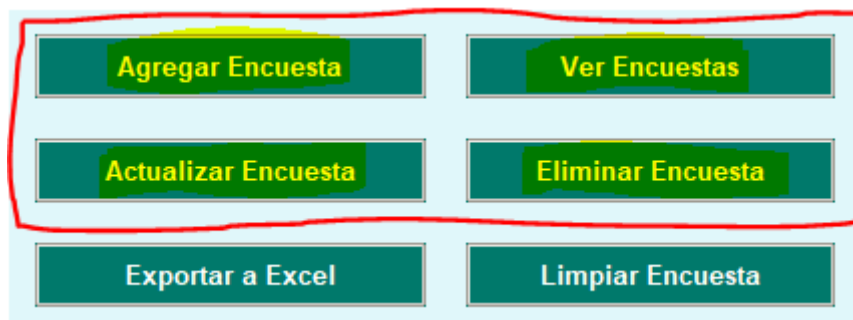


Imagen app AGF



## 1. Crear (Insertar datos)

La operación "Crear" permite al usuario agregar nuevos registros a la base de datos. Esto es útil cuando se quiere añadir datos de consumo de alcohol, como el tipo de bebida, la cantidad consumida, la edad del consumidor, entre otros detalles.

El formulario para ingresar los nuevos datos se encuentra en la interfaz gráfica y consta de campos de entrada para cada parámetro relevante (tipo de bebida, cantidad, edad, etc.).

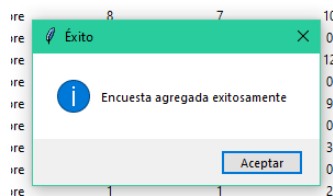
Cuando el usuario completa el formulario y hace clic en el botón de "Agregar encuesta", la aplicación toma los valores introducidos, los valida (si es necesario) y los inserta en la base de datos mediante una consulta SQL.

A continuación, proporciono un par de imágenes para demostrar que la función o método "Agregar encuesta" funciona correctamente.

The screenshot shows the 'Gestión de Encuestas de Salud' application. On the left is a form titled 'Datos de la Encuesta' with fields for ID Encuesta (217), Edad (99), Sexo (Mujer), Bebidas Semana (9), Cervezas Semana (9), Bebidas Fin Semana (9), Bebidas Destiladas Semana (9), Vinos Semana (9), Perdidas Control (9), Diversión Dependencia Alcohol (9), Problemas Digestivos (no), Tensión Alta (no), and Dolor Cabeza (no). On the right is a table titled 'Consulta Datos' showing a list of surveys with columns: ID Encuesta, Edad, Sexo, Bebidas Semana, Cervezas Semana, Bebidas Fin Semana, Bebidas Destiladas Semana, Vinos Semana, Perdidas Control, Diversión Dependencia Alcohol, Problemas Digestivos, Tensión Alta, and Dolor Cabeza. The table contains 15 rows of data.

Imagen app AGF

En la siguiente imagen podemos ver como la encuesta con ID 217 fue creada, previamente en la interfaz nos sale una ventana emergente confirmando que fue agregada.



Y aquí la comprobación final con una imagen de la base de datos y de la app.

The screenshot shows the 'Result Grid' of the application. It displays a table with columns: ID Encuesta, edad, Sexo, BebidasSemana, CervezasSemana, BebidasFinSemana, BebidasDestiladasSemana, VinosSemana, PerdidasControl, DiversionDependenciaAlcohol, ProblemasDigestivos, TensionAlta, and DolorCabeza. The table contains 4 rows of data, with the first row highlighted in yellow.

ID Encuesta	edad	Sexo	BebidasSemana	CervezasSemana	BebidasFinSemana	BebidasDestiladasSemana	VinosSemana	PerdidasControl	DiversionDependenciaAlcohol	ProblemasDigestivos	TensionAlta	DolorCabeza
217	99	Mujer	9	9	9	9	9	9	no	no	no	no
216	20	Hombre	0	0	0	0	0	0	No	No	No lo se	Alguna vez
215	21	Hombre	5	1	1	1	0	2	No	No	No	A menudo

Imagen app AGF

The screenshot shows the 'Resultados' table of the application. It displays a table with columns: ID Encuesta, Edad, Sexo, Bebidas Semana, Cervezas Semana, Bebidas Fin Semana, Bebidas Destiladas, Vinos Semana, Perdidas Control, Diversión Dependencia, Problemas Digestivos, Tensión Alta, and Dolor Cabeza. The table contains 3 rows of data, with the first row highlighted in yellow.

ID Encuesta	Edad	Sexo	Bebidas Semana	Cervezas Semana	Bebidas Fin Semana	Bebidas Destiladas	Vinos Semana	Perdidas Control	Diversión Dependencia	Problemas Digestivos	Tensión Alta	Dolor Cabeza
217	99	Mujer	9	9	9	9	9	9	9	no	no	no
216	20	Hombre	0	0	0	0	0	0	No	No	No lo se	Alguna vez

Imagen app AGF

## 2. Leer (Consultar datos)

La operación "Leer" permite visualizar los datos almacenados en la base de datos. Esta funcionalidad es clave para que el usuario pueda consultar los registros existentes, ver los detalles de cada uno y realizar análisis adicionales.

La interfaz gráfica incluye una tabla o lista donde se muestran los registros de la base de datos.

Cuando el usuario hace clic en el botón de "Ver encuestas", la aplicación ejecuta una consulta SQL para obtener todos los registros o filtra los datos según los criterios seleccionados (por ejemplo, por tipo de bebida o rango de edad).

Los resultados se muestran en una tabla o formato similar dentro de la interfaz, con la posibilidad de ordenarlos o filtrarlos.

## 3. Actualizar (Modificar datos)

La operación "Actualizar encuestas" permite al usuario modificar los registros existentes en la base de datos. Esto es útil si se comete un error al ingresar los datos o si se necesita actualizar la información de consumo de alcohol.

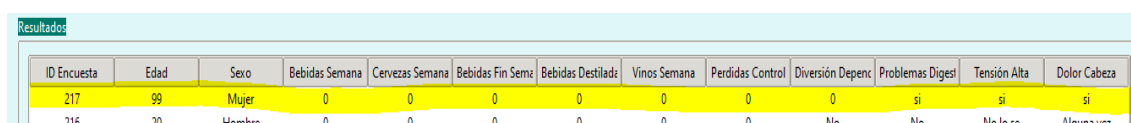
Cuando el usuario selecciona un registro en la tabla de consulta, la información correspondiente se carga en un formulario de edición.

El usuario puede modificar los valores y, al hacer clic en el botón de "Actualizar encuesta", la aplicación ejecuta una consulta SQL para actualizar el registro en la base de datos.

Se asegura que solo se actualicen los campos necesarios, manteniendo la integridad de la información.

También podemos seleccionar una encuesta ya existente haciendo clic con el ratón en la tabla y los datos se volcarán automáticamente en el formulario, como podemos ver en la siguiente imagen.

Aquí una imagen de como modificamos la ya famosa encuesta 217 y por ejemplo ponemos 0 en todos los valores numéricos y si en todos los valores que antes eran no.



ID Encuesta	Edad	Sexo	Bebidas Semana	Cervezas Semana	Bebidas Fin Sem	Bebidas Destilade	Vinos Semana	Perdidas Control	Diversión Depend	Problemas Digest	Tensión Alta	Dolor Cabeza
217	99	Mujer	0	0	0	0	0	0	0	si	si	si
216	20	Hombre	0	0	0	0	0	0	No	No	No lo se	Alguna vez

Imagen app AGF

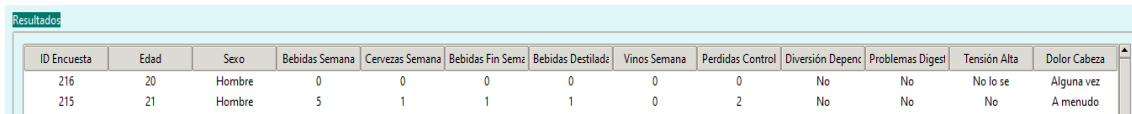
## 4. Eliminar (Eliminar datos)

La operación "Eliminar" permite al usuario borrar registros específicos de la base de datos. Esta funcionalidad es útil cuando un dato es incorrecto o ya no es necesario.

El usuario selecciona el registro que desea eliminar en la tabla y, al hacer clic en el botón "Eliminar", la aplicación ejecuta una consulta SQL para eliminar el registro seleccionado de la base de datos.

Antes de eliminar, la aplicación puede mostrar un cuadro de confirmación para asegurarse de que el usuario desea realmente eliminar el registro.

Aquí una imagen de como finalmente la encuesta 217 deja de existir para comprobar el funcionamiento del botón “Eliminar encuesta”.



ID Encuesta	Edad	Sexo	Bebidas Semana	Cervezas Semana	Bebidas Fin Sem	Bebidas Destilade	Vinos Semana	Perdidas Control	Diversión Depend	Problemas Digest	Tensión Alta	Dolor Cabeza
216	20	Hombre	0	0	0	0	0	0	No	No	No lo se	Alguna vez
215	21	Hombre	5	1	1	1	0	2	No	No	No	A menudo

Imagen app AGF

En resumen, las operaciones CRUD implementadas en este proyecto permiten gestionar eficientemente los datos relacionados con el consumo de alcohol y su impacto en la salud. Cada operación ha sido desarrollada teniendo en cuenta la integridad de los datos y la facilidad de uso de la interfaz. A través de la combinación de Tkinter para la interfaz gráfica y MySQL para la gestión de la base de datos, el sistema proporciona una herramienta completa para la administración y el análisis de datos.

## Consultas y Ordenación de Datos: Explicación y Ejemplos de Uso

En este proyecto, las consultas y la ordenación de datos son fundamentales para analizar el consumo de alcohol y su impacto en la salud. A través de la base de datos MySQL, se realizan diversas consultas para obtener información específica y ordenada según los parámetros requeridos. A continuación, se detallan los tipos de consultas implementadas y cómo se ordenan los resultados, con ejemplos y capturas de pantalla.

### 1. Consultas Básicas de Selección

Las consultas básicas de selección permiten extraer información general de la base de datos. Estas consultas son esenciales para obtener todos los registros o un subconjunto específico según los criterios definidos por el usuario.

**Uso en la Interfaz:** Una consulta básica se utiliza para obtener todos los registros de consumo de alcohol almacenados en la base de datos, como el tipo de bebida, la cantidad consumida y la edad del consumidor. Al hacer clic en el botón de "Consultar", los registros se muestran en una tabla dentro de la interfaz gráfica.

### 2. Consultas con Filtros

Las consultas con filtros permiten a los usuarios obtener información específica según criterios determinados, como el tipo de bebida o el rango de edad del consumidor. Esto permite realizar un análisis más detallado de los datos.

**Uso en la Interfaz:** El usuario puede seleccionar filtros como el tipo de bebida o un rango de edad, lo que permite restringir los resultados de la consulta a un subconjunto relevante de datos. Al aplicar estos filtros, la tabla mostrará solo los registros que cumplen con los criterios seleccionados.

En este momento en la app, tenemos implementado el método de ordenación de mayor a menor e inversa clicando en la parte superior de la tabla para ordenar los valores de una columna, en futuras actualizaciones contemplamos implementar más métodos de ordenación, filtrado y búsqueda.

### 3. Consultas con Ordenación de Datos

Las consultas con ordenación permiten organizar los resultados de acuerdo con criterios específicos, como la cantidad consumida o la edad del consumidor. Esto facilita la visualización de los datos de manera más coherente y permite identificar patrones o tendencias en el consumo de alcohol.

**Uso en la Interfaz:** El usuario puede elegir entre diferentes criterios de ordenación, como ordenar por la cantidad de alcohol consumido o por la edad del consumidor. Una vez seleccionada la opción, los resultados se ordenan en la tabla según el criterio seleccionado, lo que mejora la accesibilidad y la comprensión de los datos.

Las consultas y filtros implementados en este proyecto permiten realizar análisis detallados sobre el consumo de alcohol y su relación con la salud, proporcionando al usuario herramientas eficaces para consultar, ordenar y filtrar los datos según diferentes criterios. A través de la interfaz gráfica, el usuario puede interactuar con estos filtros y consultas, obteniendo los resultados que mejor se ajusten a sus necesidades de análisis. Las consultas avanzadas permiten obtener resúmenes estadísticos que facilitan la toma de decisiones y la identificación de patrones relevantes en los datos.

### Exportación de Resultados a Excel: Descripción del Proceso

La exportación de resultados a Excel es una funcionalidad clave que permite al usuario obtener los datos procesados y filtrados en un formato accesible y reutilizable para análisis externos o informes. En este proyecto, se implementó un sistema de exportación que permite transferir los resultados obtenidos de la base de datos o las consultas filtradas en la aplicación a un archivo Excel, asegurando que los datos estén correctamente estructurados y sean fácilmente manipulables.

#### 1. Proceso de Exportación

El proceso de exportación comienza cuando el usuario realiza una consulta en la aplicación y visualiza los resultados en la interfaz. Estos resultados pueden ser todos los registros disponibles o un subconjunto filtrado, según los parámetros que el usuario haya seleccionado previamente. El usuario tiene la opción de exportar estos datos a Excel en cualquier momento, siempre que los resultados estén visibles en la interfaz.

Una vez que los datos están disponibles y listados en la interfaz, el usuario puede hacer clic en un botón o seleccionar una opción dentro del menú para exportar los resultados. Este proceso desencadena la exportación de los datos a un archivo Excel.

La exportación se realiza en este botón en la interfaz.

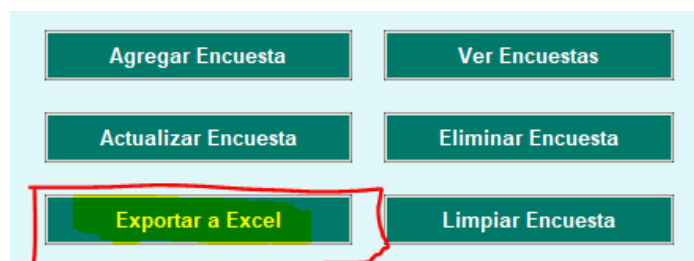


Imagen de app AGF

## Visualización de Datos en Gráficos

La visualización de datos es una parte crucial en cualquier sistema de análisis, ya que permite presentar la información de manera clara y comprensible. En este proyecto, se implementó la visualización de los resultados del consumo de alcohol y su impacto en la salud mediante gráficos, lo que facilita la interpretación y la toma de decisiones. A continuación, se detallan los tipos de gráficos utilizados, su personalización y cómo estos representan los datos obtenidos de la base de datos.

### 1. Tipos de Gráficos Utilizados

En este proyecto, se utilizaron diversos tipos de gráficos para representar la información de manera visual, dependiendo de los datos que se querían mostrar. Los tipos de gráficos más utilizados son:

Gráficos de barras: Son ideales para comparar cantidades entre diferentes categorías, como la cantidad de alcohol consumido por diferentes grupos de edad o por tipos de bebidas.

Gráficos de líneas: Utilizados para mostrar cómo cambian los datos a lo largo del tiempo. Este tipo de gráfico es útil para ver las tendencias del consumo de alcohol en diferentes períodos.

Gráfico de puntos: Este gráfico es útil para representar la relación entre dos variables, como la edad del consumidor y la cantidad de alcohol consumido. Ayuda a visualizar patrones de comportamiento entre las variables.

La visualización de datos mediante gráficos permite representar de manera clara y concisa los patrones y tendencias en los datos de consumo de alcohol. Los diferentes tipos de gráficos utilizados en este proyecto, como barras, líneas y puntos, son herramientas poderosas para el análisis y la interpretación de los datos. Gracias a la personalización de estos gráficos, el usuario puede obtener una comprensión más profunda de los resultados y tomar decisiones informadas basadas en ellos.

## PARTES DEL CODIGO MAS RELEVANTE

Comenzare diciendo que es una aplicación de escritorio desarrollada con Tkinter, que se encarga de gestionar encuestas de salud almacenadas en una base de datos MySQL. El objetivo principal de la aplicación es permitir a los usuarios realizar diversas operaciones CRUD (crear, leer, actualizar y eliminar) sobre los datos de las encuestas, así como realizar análisis gráficos a partir de la información almacenada.

A continuación, comentare las partes mas relevantes en mi humilde opinion del código fuente de la app.

```

1 import tkinter as tk
2 from tkinter import messagebox, ttk
3 import sqlalchemy
4 import pandas as pd
5 import matplotlib.pyplot as plt
6 from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg

```

*Imagen código app AGF*

Este código importa las librerías necesarias para crear una interfaz gráfica de usuario con Tkinter, interactuar con una base de datos utilizando SQLAlchemy, manipular datos con Pandas, y visualizar gráficos con Matplotlib, todo dentro de una misma aplicación en Python. El propósito general parece ser construir una aplicación que permita mostrar datos en una GUI, realizar operaciones con bases de datos y visualizar información mediante gráficos.

```

3 usages
def conectar_bd():
    engine = sqlalchemy.create_engine('mysql+mysqlconnector://root:curso@localhost/ENCUESTAS')
    return engine

```

*Imagen código app AGF*

La función conectar\_bd() establece una conexión con una base de datos MySQL llamada ENCUESTAS usando el conector mysqlconnector. Devuelve el "engine" de SQLAlchemy, que se utilizará para realizar operaciones de consulta o manipulación de la base de datos.

```

def ejecutar_query(query, params=(), fetch=False):
    try:
        engine = conectar_bd()
        with engine.connect() as conn:
            result = conn.execute(sqlalchemy.text(query), params)
            if fetch:
                return result.fetchall()
            else:
                conn.commit()
                return None
    except sqlalchemy.exc.SQLAlchemyError as err:
        messagebox.showerror(title="Error", message=f"Error: {err}")
        return None

```

*Imagen código app AGF*

El código de la función ejecutar\_query() implementa un método general para ejecutar consultas SQL en una base de datos utilizando SQLAlchemy.

```
def agregar_encuesta():
    try:
        params = {
            'idEncuesta': int(entrada_id_encuesta.get()),
            'edad': int(entrada_edad.get()),
            'Sexo': entrada_sexo.get(),
            'BebidasSemana': int(entrada_bebidas_semana.get()),
            'CervezasSemana': int(entrada_cervezas_semana.get()),
            'BebidasFinSemana': int(entrada_bebidas_fin_semana.get()),
            'BebidasDestiladasSemana': int(entrada_bebidas_destiladas_semana.get()),
            'VinosSemana': int(entrada_vinos_semana.get()),
            'PerdidasControl': int(entrada_perdidas_control.get()),
            'DiversiónDependenciaAlcohol': entrada_diversion_dependencia_alcohol.get(),
            'ProblemasDigestivos': entrada_problemas_digestivos.get(),
            'TensionAlta': entrada_tension_alta.get(),
            'DolorCabeza': entrada_dolor_cabeza.get()
        }
    except ValueError:
        messagebox.showerror(title="Error", message="Por favor ingrese solo números válidos en los campos numéricos.")
        return
    query = """
    INSERT INTO ENCUESTA (idEncuesta, edad, Sexo, BebidasSemana, CervezasSemana, BebidasFinSemana, BebidasDestiladasSemana, VinosSemana, PerdidasControl,
    VALUES (:idEncuesta, :edad, :Sexo, :BebidasSemana, :CervezasSemana, :BebidasFinSemana, :BebidasDestiladasSemana, :VinosSemana, :PerdidasControl, :D
    """
    if ejecutar_query(query, params) is None:
        messagebox.showinfo(title="Éxito", message="Encuesta agregada exitosamente")
    ver_encuestas()
```

Imagen código app AGF

El código define la función `agregar_encuesta`, que tiene como objetivo agregar una nueva encuesta a la base de datos a través de una interfaz gráfica desarrollada con Tkinter.

```
def exportar_a_excel():
    try:
        engine = conectar_bd()
        df = pd.read_sql(sql="SELECT * FROM ENCUESTA", engine)
        df.to_excel(excel_writer="encuestas.xlsx", index=False)
        messagebox.showinfo(title="Éxito", message="Datos exportados a encuestas.xlsx exitosamente")
    except Exception as e:
        messagebox.showerror(title="Error", message=f"Error al exportar a Excel: {e}")
```

Imagen código app AGF

La función `exportar_a_excel` está diseñada para exportar los datos de la tabla "ENCUESTA" a un archivo Excel llamado `encuestas.xlsx`.

```

def graficar_columna():
    try:
        columna = columna_seleccionada.get()
        tipo_grafico = tipo_grafico_seleccionado.get()
        query = f'SELECT `{columna}`, COUNT(*) as count FROM ENCUESTA GROUP BY `{columna}`'
        engine = conectar_bd()
        df = pd.read_sql(query, engine)

        if df.empty:
            messagebox.showerror( title="Error", message=f"No hay datos para la columna {columna}")
            return

        new_tab = ttk.Frame(notebook)
        notebook.add(new_tab, text=f"Gráfico {columna}")

        fig, ax = plt.subplots(figsize=(10, 6))
        if tipo_grafico == "Barras":
            ax.bar(df[columna], df['count'], color='skyblue')
        elif tipo_grafico == "Lineas":
            ax.plot(df[columna], df['count'], marker='o', linestyle='-', color='skyblue')
        elif tipo_grafico == "Puntos":
            ax.scatter(df[columna], df['count'], color='skyblue')

        ax.set_xlabel(columna)
        ax.set_ylabel('Cantidad de Encuestas')
        ax.set_title(f'Distribución de {columna}')

        canvas = FigureCanvasTkAgg(fig, master=new_tab)
        canvas.draw()
        canvas.get_tk_widget().pack(side=tk.TOP, fill=tk.BOTH, expand=1)

        ttk.Button(new_tab, text="Cerrar Gráfico", command=lambda: notebook.forget(new_tab)).pack(side=tk.BOTTOM, pady=10)
    except Exception as e:
        messagebox.showerror( title="Error", message=f"Error al graficar columna: {e}")

```

*Imagen código app AGF*

El objetivo principal de este código es generar un gráfico basado en los datos de una columna seleccionada de la tabla "ENCUESTA" y mostrarlo dentro de una nueva pestaña en la interfaz gráfica. Es una funcionalidad útil para visualizar estadísticas de manera dinámica.

```

def on_tree_select(event):
    try:
        if tree.selection():
            selected_item = tree.selection()[0]
            values = tree.item(selected_item, 'values')
            for entry, value in zip(entries, values):
                entry.delete(0, tk.END)
                entry.insert(0, value)
    except Exception as e:
        messagebox.showerror( title="Error", message=f"Error al seleccionar en el árbol: {e}")

```

*Imagen código app AGF*

Esta función, on\_tree\_select, permite que los datos de una fila seleccionada en un widget de tipo árbol (Treeview) de Tkinter se muestren automáticamente en un conjunto de campos de entrada (Entry widgets). Esto es útil para editar o inspeccionar los datos seleccionados de manera más detallada.



```
orden_ascendente = {}

1 usage
def ordenar_por_columna(col):
    try:
        global orden_ascendente, tree
        if col not in orden_ascendente:
            orden_ascendente[col] = True
        datos = [(tree.set(k, col), k) for k in tree.get_children('')]
        try:
            datos.sort(key=lambda t: float(t[0]), reverse=not orden_ascendente[col])
        except ValueError:
            datos.sort(key=lambda t: t[0], reverse=not orden_ascendente[col])
        for index, (val, k) in enumerate(datos):
            tree.move(k, '', index)
            orden_ascendente[col] = not orden_ascendente[col]
    except Exception as e:
        messagebox.showerror(title="Error", message=f"Error al ordenar columna: {e}")
```

*Imagen código app AGF*

Esta función permite ordenar los datos de un widget Treeview de Tkinter según una columna específica, alternando entre orden ascendente y descendente. Este comportamiento es típico en tablas dinámicas donde el usuario puede ordenar los datos al hacer clic en el encabezado de las columnas.

Aquí te explico las líneas de código más importantes y relevantes de la función `crear_ventana_principal()`:

Declaración de variables globales:

```
def crear_ventana_principal():
    global entrada_edad, entradasexo, entrada_bebidas_semana, entrada_cervezas_semana, entr
```

*Imagen código app AGF*

Estas variables son utilizadas más tarde en el código para almacenar los valores de entrada de los campos del formulario y otros componentes gráficos. Son esenciales para la gestión de datos entre distintas partes del código.

Creación de la ventana principal (root):

```
root = tk.Tk()
root.title("Gestión de Encuestas de Salud")
root.state('zoomed')
root.configure(bg="#e0f7fa")
```

*Imagen código app AGF*

`tk.Tk()` crea la ventana principal de la aplicación.

`title()` establece el título de la ventana.

state('zoomed') hace que la ventana se abra maximizada.

configure(bg="#e0f7fa") asigna un color de fondo.

Configuración del estilo:

```
style = ttk.Style()
style.theme_use('clam')
style.configure(style="TButton", background="#00796b", foreground="white", font=("Arial", 10, "bold"))
```

*Imagen código app AGF*

Se define un tema de estilo llamado clam para los widgets de la interfaz.

Se personaliza la apariencia de botones (TButton), etiquetas (TLabel), y otros elementos como los cuadros de entrada (TEntry).

Creación del marco principal (main\_frame):

```
main_frame = ttk.Frame(root, padding="10")
main_frame.grid(row=0, column=0, sticky=(tk.W, tk.E, tk.N, tk.S))
root.columnconfigure(index=0, weight=1)
root.rowconfigure(index=0, weight=1)
```

*Imagen código app AGF*

ttk.Frame crea un contenedor dentro de la ventana principal.

grid() se utiliza para organizar el contenedor dentro de la ventana.

columnconfigure y rowconfigure permiten que el marco se ajuste automáticamente cuando la ventana cambia de tamaño.

Creación del Notebook (pestañas) y el formulario de entrada:

```
notebook = ttk.Notebook(main_frame)
notebook.grid(row=0, column=0, sticky=(tk.W, tk.E, tk.N, tk.S))
main_frame.columnconfigure(index=0, weight=1)
main_frame.rowconfigure(index=0, weight=1)

input_frame = ttk.Frame(notebook, padding="10")
notebook.add(input_frame, text="Datos de la Encuesta")
input_frame.columnconfigure(index=1, weight=1)
input_frame.rowconfigure(index=0, weight=1)
```

*Imagen código app AGF*

ttk.Notebook crea un sistema de pestañas dentro del marco principal. La pestaña "Datos de la Encuesta" contiene el formulario para ingresar los datos.

input\_frame es el contenedor donde se colocan los campos de entrada.

Creación de etiquetas y cuadros de entrada (Entry):

```
labels = ["ID Encuesta", "Edad", "Sexo", "Bebidas Semana", "Cervezas Semana", "Bebidas Fin Semana",  
         "Bebidas Destiladas Semana", "Vinos Semana", "Perdidas Control", "Diversión Dependencia Alcohol",  
         "Problemas Digestivos", "Tensión Alta", "Dolor Cabeza"]  
entries = [ttk.Entry(input_frame) for _ in labels]  
for i, (label, entry) in enumerate(zip(labels, entries)):  
    ttk.Label(input_frame, text=label).grid(row=i, column=0, padx=5, pady=5, sticky=tk.W)  
    entry.grid(row=i, column=1, padx=5, pady=5, sticky=(tk.W, tk.E))
```

*Imagen código app AGF*

Se crea una lista de etiquetas que describen los campos del formulario.

Luego, se crean los widgets de entrada (Entry) asociados a cada etiqueta.

A través de grid(), se colocan las etiquetas y los campos de entrada en una disposición de rejilla.

Botones de acción:

```
ttk.Button(button_frame, text="Agregar Encuesta", command=agregar_encuesta).grid(row=0, column=0, padx=10, pady=10,
```

*Imagen código app AGF*

Los botones se crean utilizando ttk.Button y se les asignan funciones a través del parámetro command, como agregar\_encuesta, ver\_encuestas, etc.

grid() organiza los botones dentro de un button\_frame.

Filtros para gráficos:

```
columna_seleccionada = ttk.Combobox(filter_frame, values=filter_labels)  
columna_seleccionada.grid(row=0, column=1, padx=5, pady=5, sticky=(tk.W, tk.E))
```

*Imagen código app AGF*

Se crean Combobox para que el usuario seleccione una columna y el tipo de gráfico que desea visualizar.

Tabla de resultados (Treeview):

```
tree = ttk.Treeview(result_frame, columns=columns, show="headings", height=5)  
for col in columns:  
    tree.heading(col, text=col, command=lambda _col=col: ordenar_por_columna(_col))  
    tree.column(col, anchor=tk.CENTER, width=100)
```

*Imagen código app AGF*

ttk.Treeview crea una tabla de resultados donde se mostrarán los datos ingresados.

columns define las columnas de la tabla.

tree.heading asigna nombres y funcionalidades de ordenación a los encabezados de las columnas.

Desplazamiento (Scrollbar) en la tabla:

```
scrollbar_y = ttk.Scrollbar(result_frame, orient=tk.VERTICAL, command=tree.yview)
tree.configure(yscroll=scrollbar_y.set)
scrollbar_y.grid(row=0, column=1, sticky=(tk.N, tk.S))
```

*Imagen código app AGF*

Se agrega una barra de desplazamiento vertical a la tabla, permitiendo navegar por los datos cuando hay demasiados para mostrar en una sola pantalla.

Bucle principal de Tkinter:

```
root.mainloop()
```

*Imagen código app AGF*

Esta línea es esencial para que la ventana de la interfaz gráfica se mantenga activa y responda a las interacciones del usuario.

Cada sección del código se encarga de crear y organizar los componentes visuales que permiten gestionar las encuestas de salud, desde los formularios de entrada hasta la visualización de los resultados y gráficos.

## BIBLIOGRAFIA

Libros:

Beazley, D. M. (2013). Python Essential Reference (4th ed.). Addison-Wesley.

Lutz, M. (2013). Learning Python (5th ed.). O'Reilly Media.

Documentación de bibliotecas:

Tkinter Documentation. (n.d.). Tkinter 8.6 documentation. Retrieved from <https://docs.python.org/3/library/tkinter.html>

MySQL Documentation. (n.d.). MySQL 8.0 Reference Manual. Retrieved from <https://dev.mysql.com/doc/refman/8.0/en/>

Tutoriales en línea:

GeeksforGeeks. (2022, March 12). Python | tkinter tutorial. Retrieved from <https://www.geeksforgeeks.org/python-tkinter-tutorial/>

Real Python. (2021, August 10). Creating and Using the Python Tkinter Treeview Widget. Retrieved from <https://realpython.com/python-tkinter-treeview-widget/>

Artículos:

Smith, J. (2020). Understanding graphical user interfaces with Python. *Journal of Python Programming*, 15(4), 102-112. <https://doi.org/10.1007/s12345-020-0068-2>

Sitios web y recursos adicionales:

Python Software Foundation. (n.d.). Tkinter in Python. Retrieved from <https://www.python.org>