

Generador de datos

Generador de Datos

**NOMBRE Y APELLIDOS: Jorge Wang Wang**

**CICLO: DAM**

**TUTOR/A: Francisco Aliseda**

**CURSO: 23/24**

# Contenido

La aplicación es un generador de datos para base de datos, la cual el usuario al acceder a ella vera una serie de opciones en forma de botones como por ejemplo nombre, DNI, imágenes, etc. Y cada uno de ellos tendrá su propio modulo o una caja con distintas propiedades, que permiten personalizar los datos de cada opción especifica, como, por ejemplo, si en nombre quiere que muestre los apellidos, y si estos apellidos desean que estén juntos en nombre, o estén separados en otra columna, o que al generar un país te permita seleccionar de que continente quieres que sean.

Una vez seleccionado los datos a generar, habrá que escribir la cantidad de datos que desea generar, con un máximo de 1000 para no saturar la aplicación, además también se deberá elegir el documento o formato de salida con la que saldrán los datos, como si es en XML, JSON, CSV, etc. Otra función que tiene es que se puede cambiar el idioma al inglés, no obstante, no solo te cambia el idioma de los módulos y los botones, sino que también cambia el sistema de generación de datos, es decir, generara información del reino unido, como sus DNI, seguridad Social, etc.

# Justificación del proyecto

Cuando Empecé el curso, siempre tenia que programar con base de datos, ya sea en programación, en lenguaje de marcas, o en la propia asignatura de base de datos, en todos ellos siempre me surgía el mismo problema, que datos probar, tengo que insertar como 6 datos, pero no se me ocurre, luego tardo lo suyo y al final decido poner el mismo nombre y mismos valores, o cambiándolos un poco, la cual luego me podía equivocar, o confundir por ser muy similares. Por ello pensé, ojala existiese alguna aplicación que te permite crear datos o rellenar la base de datos, la cual si existían pagina webs porque los descubrí hace poco, pero claro te generaban datos que no estaban ni verificados, es decir si creabas 10 DNI, y lo pasabas en un comprobador de DNI, con suerte 1 de 10 era valido y el resto no. Y además el único que parecía decente, estaba en ingles y generaba los datos en ese idioma. Otro momento, aunque este ya es un poco mas posterior luego de decidir el proyecto, en las practicas mientras hacíamos una macro, había que probar que si la tabla fuese mas larga la macros siguiese funcionando, y allí nos pasamos un buen rato creando personas falsas y añadiéndoles datos falsos, y justo pensé que con mi aplicación a lo mejor habría sido más rápido. Por ello me decidí a crear una aplicación que sirva para validar datos o generar validos de otras paginas o base de datos para poder llenar de información

# Contenido

## 3.1 Herramientas y Lenguajes Utilizados

Para crear esta aplicación decide usar Visual Studio que es un IDE la cual estoy acostumbrado usar y tienes varias extensiones que me permiten acelerar procesos o mejorar visualmente la forma de ver los datos, como SQLite viewer que permite ver en una tabla la base de datos de SQLite o rainbow csv que te Colorea cada columna de un csv y así puedes relacionarlo mejor, Además de tener varios atajo y funcionalidades que me permite acelerar algunos procesos al programar.

Para el lenguaje utilizado decidí usar Python, debido es uno de los mejores en el manejo de datos por su sintaxis sencilla. Al requerir escribir menos líneas para hacer lo mismo que otros lenguajes de programación, y también sobre todo por su amplias ecosistema de librerías, la cual muchas permiten el manejo de datos, las librerías que use son:

* *BeautifulSoup y Requests*: dos librerías que permiten realizar scrapping, la cual requests te permite realizar solicitudes http a páginas, y combinado con BeautifulSoup, esta te permite sacar los documentos HTML o XML y manejarlos de forma sencilla. Estas dos bibliotecas fueron una de las mayores razones de seleccionar Python como lenguaje para programar.
* *PyQt6*: A la hora de decidir que librería usar para crear la interfaz, decidí usar PyQt6 en vez de Pyside o TKinter debido a que era más completa a comparación de los otros dos, puede ser más complicado usarlo debido a que tiene una estructura distinta a otras librerías, pero eso permitió realizar más funciones a diferencia de las otras dos librerías, además ya tiene configurado automáticamente el reajuste si la pantalla se cambia de tamaño.
* *Faker*: Es una librería que genera datos falsos, como mi aplicación, decidí no abusar de ella porque sino mi TFG solo seria crear una interfaz y luego abusar de la librería. Pero para no desperdiciarla lo decidí usar en un solo modulo, en el de direcciones debido a que no encontraba datos y no encontraba una API interesante con la cual poder usar y sacar información de ella, además de que casi todas tenían limite o no te contestaban para pasarte una clave.
* *Random*: una librería que tiene varias funciones la cual te selecciona de forma pseudoaleatoria los datos que le pases a sus métodos, como si es un array, o dos valores como un 0 y un 9 y te selecciona un valor del medio o incluso decimales entre ellos.
* *Json*: es una biblioteca que permite el manejo de los documentos con extensión de json, la cual permite convertir archivos en objetos Python, la cual use para abrir y usar los json con los idiomas de cada módulo, y de la aplicación principal, y también permite convertir al revés, es decir, de un objeto de Python a json como use para imprimir uno de los formatos de salida.
* *Sqlite3 y pymongo*: dos librerías que sirven para el manejo de base de datos, un para base de datos relacionales como SQLite, y otros para no relacionales como MongoDB, te permiten insertar, borrar, actualizar y buscar datos en base de datos.
* *Flask*: Es un microFramework que permite la creación de paginas web pequeñas y rápidas, en su caso lo use para crear un REST API. La genialidad de esta librería es que fácil crear ya que es muy simple y flexible, y para crear rutas solo se necesita de una línea y un método.
* *String*: es una popular librería que contiene una gran cantidad de constantes para trabajar con cadenas de texto, para esta aplicación lo use para crear array mas rápido en vez de escribir el abecedario o los números posible enteros de uno en uno.

Para la base de datos decidí usar base de datos no relacionales, y base de datos relacionales, como escribí antes en la librería de sqlite3 y pymongo. Pero MongoDB es el mas utilizado debido a que la mayoría de los datos no necesita relación o una segunda tabla ya que son datos simples y sin mucha complejidad, como uno que solo guarda los nombres, u otro que guarda solo apellidos. Eso es debido a que, como guarda sus datos en json, y por ello es más dinámico y flexible a posibles cambios o expansiones, además de que su lectura y escritura es mucho mas rápida, debido a no necesitar de conexiones complejos como las bases de datos SQL. Pero como dije antes también use SQLite debido a que hay algunos datos más complejos que necesitarían mas de una tabla, la cual es el de coches que explicare mas adelante

## 3.2 Diseño de la interfaz

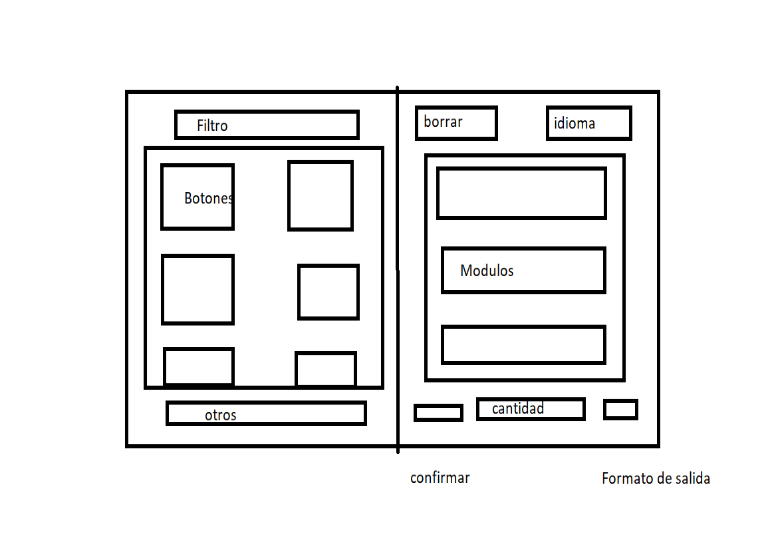
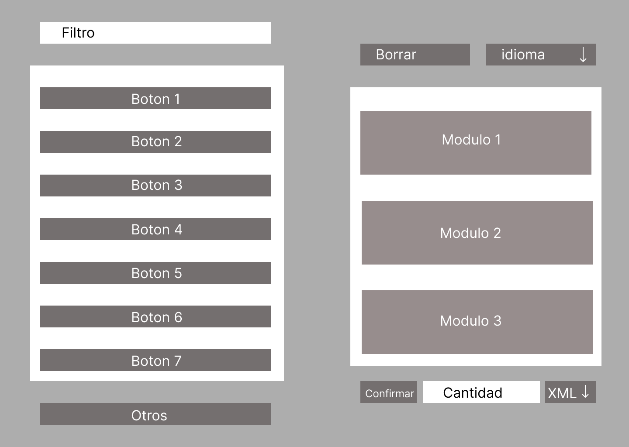
Para la creación de la aplicación antes de programar la interfaz, empecé diseñando la aplicación en Paint, la cual seria el primer boceto para la aplicación como se puede ver en el boceto consistía en dos zonas, en donde la izquierda estarán los botones colocados en dos columnas, arriba un cuadro de texto que al escribir filtraría los botones y que comenzase por la letra o palabra escrita. Abajo estaría el botón especial de otros, en donde te permitirá subir un archivo csv y con ello crear un nuevo modulo con los datos de ese csv y genere aleatoriamente los textos que contengan. En la parte de la derecha, arriba habría un botón que es el de borrar la cual limpiaría los módulos que haya en él. Al lado del botón hay un desplegable en donde se encuentra los idiomas, por ahora solo hay dos opciones, el inglés y el español, en el futuro se podría incorporar el chino, alemán, francés, etc. Debajo de los botones se encuentra los módulos que aparecen tras pulsar un botón de la parte de la izquierda. Debajo de todo hay 1 botón que sirve para confirmar y empezar a imprimir los datos, en el centro un cuadro de texto para indicar la cantidad, y a la derecha de toda una lista despegable que te permite seleccionar el formato de salida, es decir, JSON, XML, CSV, etc. El problema surgía que cuando empecé creando a programar la interfaz, hice una prueba para el filtro y surgió que si escondía los botones que no comenzase por la letra o palabra del cuadro del texto, pero el GridLayout() dejaba unos huecos muy grandes y los botones no subían. Por ello decidí cambiar el diseño de la aplicación y además dejar un dibujo mucho más fácil de leer y entender. Por ello use una herramienta nueva, en vez de Paint use Figma, que es una aplicación web o de escritorio que te permite diseñar páginas web y aplicaciones. Tras ello hice un diseño mucho mas elaborado en donde los colores representaban los widgets de la aplicación, si es color blanco son los cuadros de texto, si es gris oscuro son botones, pero si tienen una flecha son despegables. Y lo mas claros son los módulos. La diferencia del boceta a la hecha con Figma es que pasamos de dos columnas de botones a solo 1, debido a que QVBoxlayout cuando se esconde un botón, sube aquel que estaba abajo, y si lo haces volver aparecer, se sitúa en la misma posición que estaba anteriormente, es decir que se encarga automáticamente de gestionar los botones filtrados. Un detalle que se me olvido es que volví cambiar el diseño, pero al ser muy pequeño decidí no crear otro diseño, ya que solo añadí un nuevo cuadro de texto arriba entre el botón de borrar y el desplegado de idiomas para que pueda escribir el nombre de su base de datos o su tabla de SQL.

Ilustración 1 Boceto de la aplicación

Ilustración 2 Diseño con Figma de la Aplicación

## 3.3 Organización de carpetas

## 3.4 búsqueda de información

## 3.5 Programa Principal

## 3.6 Módulos

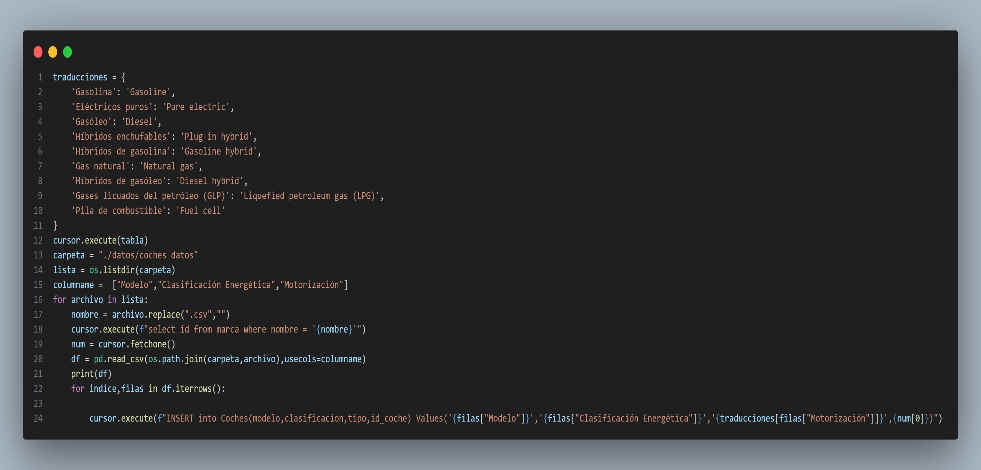
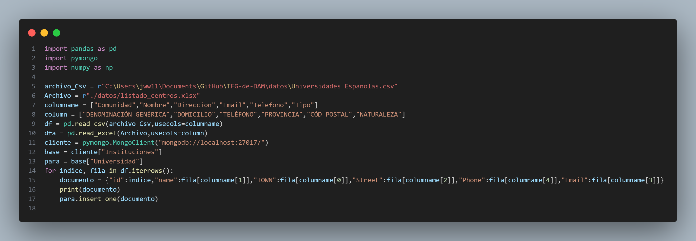
## 3.7 Servidor REST API

## 3.8 Formatos de Salida

# Conclusiones

# Bibliografia

****

****

****

Texto

Descripción generada automáticamente

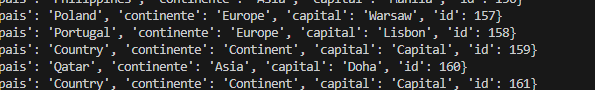
Texto

Descripción generada automáticamente









Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente