UNIVERSIDAD DE LA SALLE

Nombre de la materia

Aplicaciones 1

Nombre del programa

Nombres de los estudiantes

Juan Sebastián Vargas Pachón

Oscar Raúl Mahecha Sánchez

Nombre de la actividad

Actividad 4: Aplicación web interactiva para el análisis de mortalidad en Colombia

Unidad # 2

Aplicación web con dashborad interactivo en Python

Nombre del profesor

Cristian Duney Bermúdez Quintero

Fecha

4 de nov. de 2025

Tabla de contenido

Introducción

En el desarrollo de la actividad 4, se ha construido una aplicación web, la cual, tiene como propósito permitir el análisis de la mortalidad en Colombia durante el año 2019, a partir de los datos abiertos publicados por el Departamento Administrativo Nacional de Estadística (DANE).

Mediante el uso de herramientas de análisis y visualización interactiva, se busca facilitar la exploración de las causas de muerte, la distribución geográfica de los casos y las tendencias por sexo y edad, contribuyendo a la comprensión de los principales factores asociados a la mortalidad en el país.

El proyecto fue implementado en Python utilizando el framework Dash, que permite construir aplicaciones web analíticas integrando procesamiento de datos, componentes gráficos y lógica interactiva en una sola interfaz. Los datos fueron transformados, normalizados y combinados para generar una base consolidada que alimenta los indicadores y visualizaciones presentadas.

La aplicación se desplegó como servicio en la nube mediante Azure App Service, asegurando su accesibilidad pública y el cumplimiento de los principios de escalabilidad, persistencia y disponibilidad propios de una arquitectura PaaS.

Se configuró un entorno virtual con dependencias específicas definidas en el archivo requirements.txt, y se gestionó la ejecución mediante el servidor Gunicorn, optimizado para entornos Linux con balance de carga y rendimiento ajustado a las necesidades de la aplicación.

1. Objetivo

Desarrollar, implementar y desplegar una aplicación web interactiva que permita analizar los datos de mortalidad en Colombia durante el año 2019, utilizando herramientas de programación en Python, técnicas de análisis estadístico y visualización dinámica y un servicio de PaaS.

2. Estructura del proyecto

El proyecto se encuentra organizado en una estructura modular que facilita su comprensión, ejecución y despliegue. Cada carpeta y archivo cumple un propósito específico en la construcción de la aplicación, desde el procesamiento de los datos hasta su publicación en la nube.

- app.py: archivo principal que ejecuta la aplicación web. Contiene la lógica de carga, limpieza y análisis de los datos, así como la generación de gráficos interactivos mediante los frameworks Dash y Plotly.
- requirements.txt: lista de dependencias necesarias para ejecutar la aplicación como pandas, plotly, openpyxl, gunicorn y dash, asegurando compatibilidad y portabilidad del entorno.
- runtime.txt: define la versión de Python 3.11 utilizada para el despliegue en Azure App Service, garantizando la estabilidad del entorno PaaS.
- build.sh: script de inicialización que automatiza la construcción y preparación del entorno antes del despliegue continuo.
- .github/workflows/appservice-deploy.yml: flujo de integración y despliegue continuo configurado para publicar automáticamente la aplicación desde GitHub hacia Azure App Service.
- data/: carpeta que contiene los archivos de entrada utilizados para el análisis de mortalidad. Incluye:
- Fallecimientos.xlsx: base principal con los registros de mortalidad de Colombia (2019).
- Descripcion_cod_fall.xlsx: diccionario con la clasificación CIE-10 de las causas de muerte.
- Dep_mun.xlsx y Ubi_Dep_mun.xlsx: tablas de referencia para la codificación y localización de los departamentos y municipios según el DANE.

- datos.xlsx: archivo consolidado que integra los conjuntos anteriores para optimizar la lectura y procesamiento en la aplicación.
- README.txt: archivo informativo del conjunto de datos administrado mediante Git LFS, para garantizar la correcta carga de archivos pesados.
- README.md: documento explicativo que describe el propósito del proyecto, las instrucciones de instalación, los requerimientos técnicos y el enlace al despliegue en Azure.

3. Requisitos

El entorno de desarrollo y ejecución del proyecto está basado en Python 3.11, versión especificada en el archivo runtime.txt, y gestionado en la nube mediante Azure App Service bajo el modelo Platform as a Service (PaaS). Para garantizar la correcta instalación y portabilidad del entorno, todas las dependencias se encuentran documentadas en el archivo requirements.txt.

Las principales librerías utilizadas son:

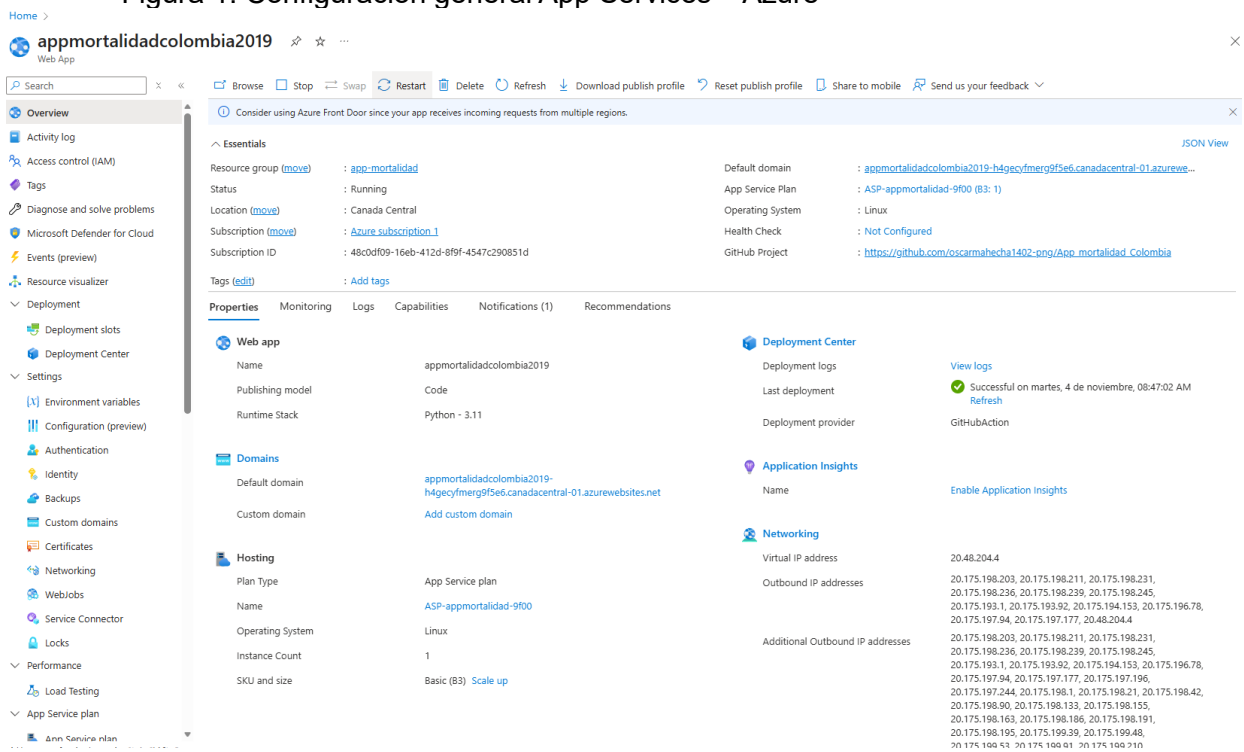
- Dash: Framework principal para el desarrollo de la aplicación web, encargado de la estructura del dashboard, la gestión de callbacks y la visualización interactiva.
- Plotly: Biblioteca de visualización que permite generar gráficos dinámicos e interactivos integrados en la interfaz del dashboard.
- Pandas: Librería para la manipulación y análisis de los datos de mortalidad, permitiendo limpiar, combinar y transformar los archivos Excel en estructuras tabulares eficientes.
- OpenPyXL: Módulo necesario para la lectura y escritura de archivos en formato Excel, utilizado en la etapa de carga y procesamiento inicial.
- Gunicorn: Servidor HTTP de alto rendimiento empleado para el despliegue de la aplicación en producción, compatible con entornos Linux en Azure.

Además, el entorno utiliza GitHub Actions como servicio de integración continua para automatizar el proceso de despliegue desde el repositorio hacia Azure, y Git LFS para gestionar los archivos de datos de gran tamaño contenidos en la carpeta data/.

4. Despliegue

El despliegue de la aplicación se realizó en Azure App Service, bajo el modelo Platform as a Service (PaaS), lo que permite ejecutar aplicaciones web sin necesidad de administrar la infraestructura subyacente. Esta elección se fundamentó en la integración nativa con GitHub Actions, la escalabilidad automática del servicio y la compatibilidad total con entornos basados en Python 3.11. En la figura 1 se aprecia la configuración general en Azure.

Figura 1. Configuración general App Services – Azure



Fuente: Elaboración propia

El flujo de integración y despliegue continuo fue configurado mediante el archivo .github/workflows/appservice-deploy.yml, el cual define las acciones necesarias para compilar, empaquetar y publicar automáticamente el código desde el repositorio de GitHub hacia la instancia activa en Azure.

La ejecución de la aplicación se realiza mediante el servidor Gunicorn, configurado en el comando de inicio (appCommandLine) como:

gunicorn app:server --workers 2 --threads 2 --timeout 600

Esta configuración optimiza el rendimiento del dashboard, permitiendo múltiples peticiones simultáneas y un tiempo de espera extendido para operaciones con volúmenes de datos considerables.

Durante las pruebas de desempeño, la aplicación fue inicialmente desplegada en el plan gratuito F1, y posteriormente escalada verticalmente al plan B3 Basic)} para aumentar los recursos de CPU y memoria, mejorando notablemente la velocidad de carga de los filtros y gráficos. El monitoreo en tiempo real se realizó a través de Log Stream y Application Insights, herramientas nativas de Azure que permiten registrar y analizar el comportamiento del servidor y la respuesta de la aplicación.

5. Software

El desarrollo de la aplicación se realizó utilizando un conjunto de herramientas integradas que garantizan la trazabilidad, reproducibilidad y automatización del proceso completo, desde la codificación hasta el despliegue en producción.

Lenguaje de programación:

Se empleó Python 3.11, elegido por su estabilidad, amplia comunidad de soporte y compatibilidad con librerías de análisis de datos y visualización.

Entorno de desarrollo:

El proyecto fue desarrollado en Visual Studio Code, configurado con extensiones para la gestión de entornos virtuales, control de versiones con Git y soporte para despliegues hacia Azure.

Frameworks y librerías:

Se utilizaron Dash y Plotly para la construcción de la interfaz web interactiva y los gráficos dinámicos. Pandas y OpenPyXL fueron empleadas para el procesamiento y manejo de datos tabulares provenientes de archivos Excel.

Control de versiones y automatización:

El repositorio del proyecto se gestionó a través de GitHub, permitiendo el control de cambios, la colaboración y la ejecución de flujos automatizados mediante GitHub Actions, definidos en el archivo .yml del repositorio.

Despliegue en la nube:

La aplicación fue desplegada en Azure App Service, aprovechando su capacidad de ejecutar aplicaciones bajo un modelo PaaS (Platform as a Service) con soporte para Python y GitHub Actions. Se utilizó el servidor Gunicorn para la ejecución en producción, optimizando la distribución de cargas concurrentes y la gestión eficiente de recursos.

6. Instalación

Para ejecutar la aplicación localmente, se debe clonar el repositorio desde GitHub y configurar un entorno virtual de Python que garantice la instalación controlada de las dependencias. El procedimiento recomendado es el siguiente:

1. Clonar el repositorio

Desde la terminal o consola del sistema, ejecutar el siguiente comando:

- git clone https://github.com/oscarmahecha1402/App_mortalidad_Colombia.git
- cd App_mortalidad_Colombia

2. Crear el entorno virtual

Dentro del directorio del proyecto, crear y activar un entorno virtual de Python:

- python -m venv antenv
- # En Windows:
- antenv\Scripts\activate
- # En Linux o macOS:
- source antenv/bin/activate

3. Instalar las dependencias

Una vez activado el entorno, instalar los paquetes requeridos que se encuentran listados en el archivo requirements.txt:

- pip install -r requirements.txt

4. Ejecución de la aplicación

Ejecutar el archivo principal app.py para iniciar el servidor local y acceder al dashboard interactivo:

- python app.py

La aplicación quedará disponible en el navegador en la dirección: http://127.0.0.1:8050

7. Visualización de resultados

La aplicación permite explorar de forma interactiva los principales indicadores de mortalidad en Colombia durante el año 2019, a partir de los microdatos del DANE. La figura 2 corresponde al mapa geográfico de fallecidos por departamento, que refleja una concentración significativa en los departamentos con mayor densidad poblacional, destacándose Antioquia, Valle del Cauca, Cundinamarca y Bogotá D.C. Esta distribución evidencia la correlación entre la magnitud de población y la frecuencia de defunciones.

Figura2. Mapa interactivo

Fuente: Elaboración propia

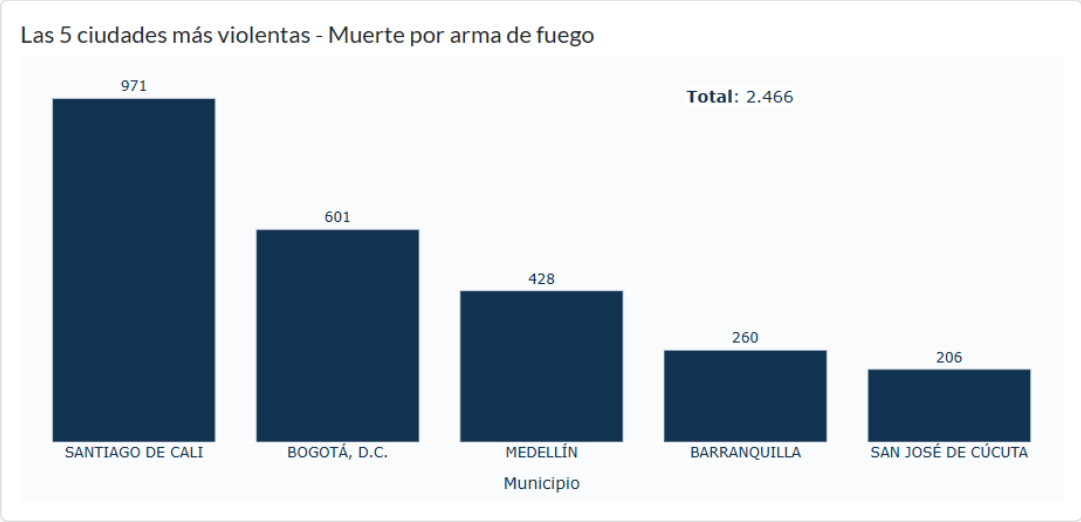
La figura 3, es el gráfico de variación mensual de fallecimientos muestra una oscilación a lo largo del año, con picos en los meses de enero, julio y diciembre, y una reducción marcada en febrero.

Figura 3. Gráfico de líneas variación mensual

Fuente: Elaboración propia

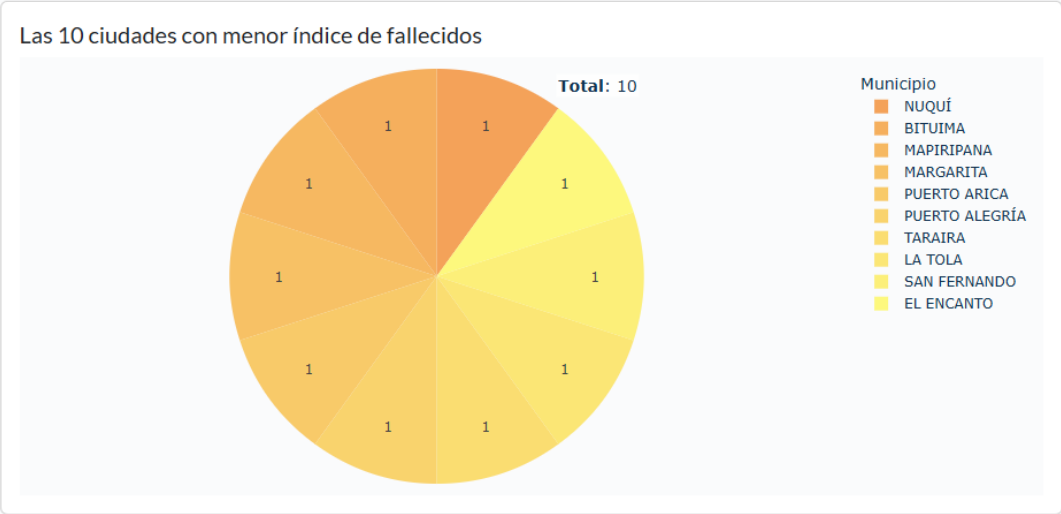
En cuanto a la distribución de causas violentas, las cinco ciudades con mayor número de muertes por arma de fuego, como se aprecia en la figura 4, fueron Santiago de Cali, Bogotá D.C., Medellín, Barranquilla y Cúcuta, lo que confirma su relación con los contextos urbanos de alta densidad y presencia de dinámicas delictivas complejas. Por contraste, en la figura 5, se muestran municipios como Nuquí, Mapiripana y La Tola registraron un solo fallecimiento, lo que representa territorios con baja incidencia o con limitaciones en el reporte estadístico.

Figura 4. Gráfico de barras, las 5 ciudades más violentas



Fuente: Elaboración propia

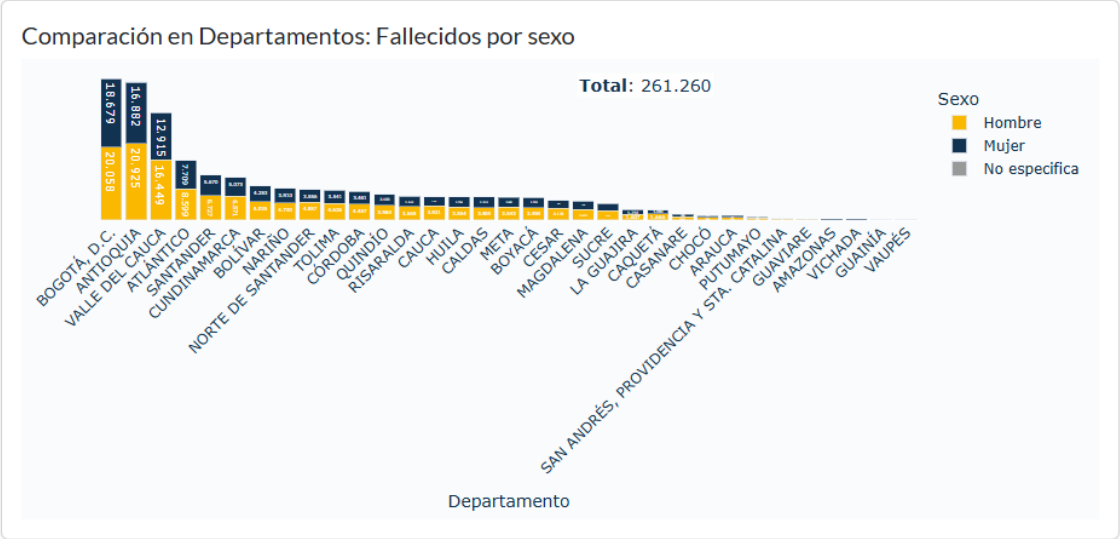
Figura 5. Gráfico de torta – 10 ciudades con menor índice de fallecidos



Fuente: Elaboración propia

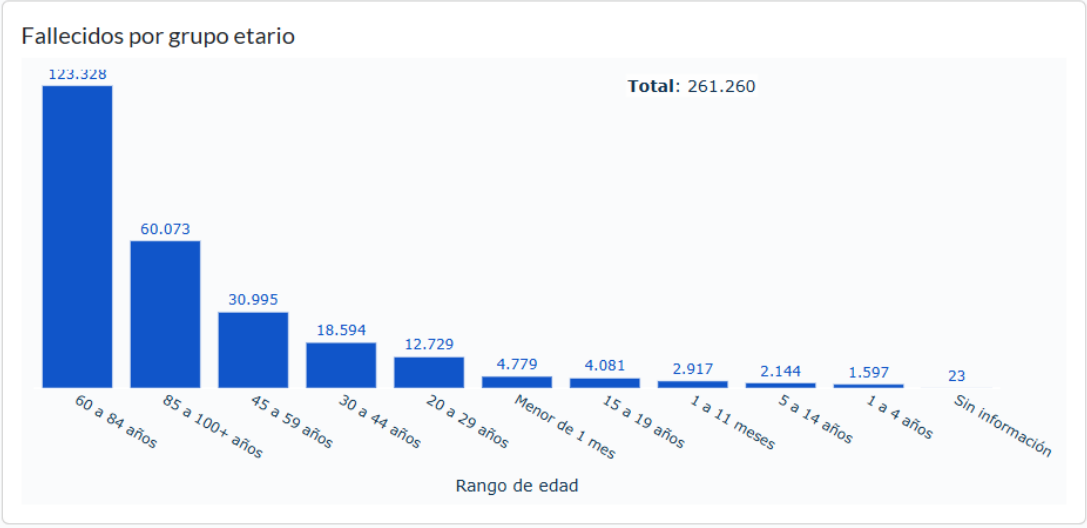
Las figuras 6 y 7, permiten el análisis comparativo de la mortalidad por sexo y grupo etario evidencia que, en la mayoría de los departamentos, particularmente en Bogotá D.C. y Antioquia. De igual forma, la mayor proporción de fallecimientos se concentra en los grupos de 60 a 84 años y 85 años o más, reflejando el impacto del envejecimiento poblacional y la prevalencia de enfermedades crónicas.

Figura 6. Gráfico de barras, comparativo por Departamentos



Fuente: Elaboración propia

Gráfico 7. Gráfico de barras - Fallecidos por grupo etario



Fuente: Elaboración propia

La figura 8. Presenta la tabla de las diez principales causas de muerte, la cual, muestra que las patologías cardiovasculares y respiratorias ocupan los primeros lugares, encabezadas por el infarto agudo de miocardio y la enfermedad pulmonar obstructiva crónica (EPOC).

Figura 8. Top 10, causas de fallecimientos

Causas: Top 10 causas (código, nombre, total)

Código	Nombre	Total de casos
I219	Infarto agudo del miocardio, sin otra especificacion	38257
J449	Enfermedad pulmonar obstructiva cronica, no especificada	7802
J440	Enfermedad pulmonar obstructiva cronica con infeccion aguda de las vias respiratorias inferiores	6867
J189	Neumonia, no especificada	6083
C169	Tumor maligno del estomago, parte no especificada	5468
X954	Agresion con disparo de otras armas de fuego, y las no especificadas, calles y carreteras	4810
C349	Tumor maligno de los bronquios o del pulmon, parte no especificada	4775
C509	Tumor maligno de la mama, parte no especificada	3798
I110	Enfermedad cardiaca hipertensiva con insuficiencia cardiaca (congestiva)	3373
X958	Agresion con disparo de otras armas de fuego, y las no especificadas, otro lugar especificado	3028
—	**Total**	**84.261**

Fuente: Elaboración propia

8. Dirección Web de la aplicación y el repositorio.

En los siguientes enlaces, se puede acceder a la aplicación y al repositorio de Github:

Actividad 4. Aplicación web interactiva para el análisis de mortalidad en Colombia

https://github.com/oscarmahecha1402-png/App_mortalidad_Colombia

Conclusiones

La construcción de la aplicación permitió integrar distintas etapas del análisis de datos, desde la limpieza y transformación hasta la visualización interactiva en un entorno local para verificar su funcionamiento inicial y luego, completamente automatizado y accesible desde la nube. Este proceso evidenció la capacidad del lenguaje Python, junto con Dash y Plotly, para desarrollar soluciones analíticas eficientes, escalables y alineadas con los principios de interoperabilidad y transparencia de los datos públicos.

El despliegue en Azure App Service demostró la importancia de los modelos PaaS en la implementación de proyectos con requerimientos de disponibilidad, escalabilidad y control de versiones. La incorporación de GitHub Actions como herramienta de integración y despliegue continuo garantizó la actualización automática del entorno y la trazabilidad completa del proceso de desarrollo, optimizando el ciclo de vida de la aplicación.

El análisis de mortalidad en Colombia durante 2019 permitió identificar patrones relevantes en la distribución geográfica, etaria y por causas de muerte. Los hallazgos reflejan la necesidad de fortalecer las políticas de salud pública orientadas a la prevención de enfermedades crónicas y la reducción de muertes violentas, especialmente en los principales centros urbanos del país, consolidando la utilidad del análisis de datos como instrumento para la toma de decisiones basada en evidencia.