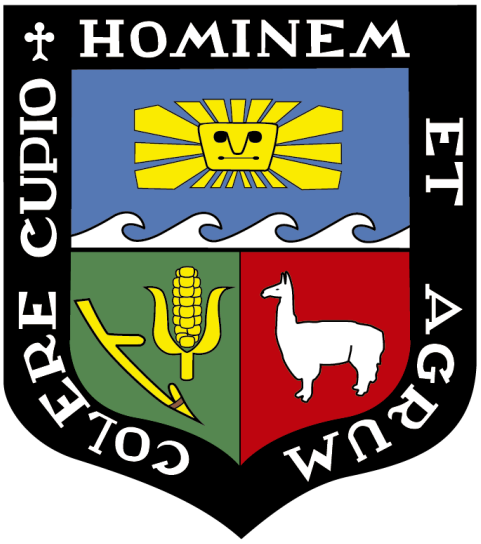
UNIVERSIDAD NACIONAL AGRARIA LA MOLINA



FACULTAD DE ECONOMÍA Y PLANIFICACIÓN

DPTO. DE ESTADÍSTICA E INFORMÁTICA

CURSO: Lenguaje de Programación II

Docente: Ana Vargas - UNALM, Perú

Nombre del proyecto: GEOVIDA

Repositorio del proyecto: https://github.com/maricielo-hc/TRABAJO_FINAL

	Apellidos y Nombres	Usuario Github	Código
1	Huamán Cóndor, Maricielo	maricielo-hc	20231494
2	Alfaro Flores, Dhalia Domini	Dominl	20231479
3	Flores Curi, Kemely Yamilé	kem19	20231491

2025

INDICE

I. Introducción

En la actualidad, el acceso a información confiable y actualizada es fundamental para la concientización y la toma de decisiones en temas medioambientales. Este proyecto tiene como objetivo recolectar, procesar y presentar datos relevantes sobre especies animales en peligro de extinción y los eventos naturales que contribuyen a esta problemática, mediante la automatización de procesos con tecnologías web y herramientas de código abierto.

Para lograrlo, se implementó una solución integral que incluye técnicas de web scraping, procesamiento de datos estructurados, automatización con GitHub Actions, y visualización interactiva utilizando HTML, CSS y JavaScript. Los datos provienen de diversas fuentes públicas, como Wikipedia, sitios de monitoreo ambiental y portales especializados. El resultado final es un sitio web estático que presenta de forma visual y accesible la información extraída, incluyendo gráficos interactivos, mapas temáticos y artículos relacionados. Este enfoque no solo permite explorar los datos de manera intuitiva, sino que también garantiza su actualización periódica sin intervención manual, gracias al uso de flujos de trabajo automatizados.

No se trata de un resultado cualquiera; de este amplio proceso nace GeoVida, una plataforma interactiva que permite a los usuarios explorar datos sobre animales amenazados, consultar artículos informativos y visualizar mapas interactivos sobre eventos que afectan el hábitat de estas especies. Este proyecto demuestra cómo la integración de tecnologías de desarrollo web, automatización y ciencia de datos puede contribuir a la creación de herramientas informáticas útiles, sostenibles y accesibles desde cualquier parte del mundo, a través de GitHub Pages.

II. Metodología

A. Proceso de extracción de datos

Para obtener la información deseada desde diversas páginas web, se utilizó web scraping, una técnica que permite extraer datos estructurados desde sitios públicos de Internet. En este proyecto se aplicó scraping para recolectar información desde varias plataformas, como Wikipedia, entre otras fuentes.

Cada script de extracción realiza, de forma general, los siguientes pasos:

- Envía una solicitud HTTP a la página objetivo.
- Utiliza la librería BeautifulSoup para parsear y analizar el contenido HTML.
- Extrae los datos relevantes mediante selectores específicos (clases, etiquetas, identificadores).
- Guarda los datos estructurados en archivos con formato .csv, .json o .js, según corresponda.

Ejemplo del archivo animales_lista.py:

```
import urllib.request
from bs4 import BeautifulSoup
import json
import re

# Leer y decodificar el HTML
url = 'https://www.peruecologico.com.pe/esp_extincion.htm'
response = urllib.request.urlopen(url)
html = response.read().decode('latin1')  # Importante para
acentos

# Parsear con BeautifulSoup
sopa = BeautifulSoup(html, 'html.parser')

# Buscar todos los <td> con width="380"
tds = sopa.find_all('td', width="380")

# Extraer y limpiar nombres científicos
nombres = []
for td in tds:
    texto = td.get_text(strip=True)

# Limpiar: eliminar saltos de línea, tabulaciones, y espacios
múltiples
    texto_limpio = re.sub(r'\s+', ' ', texto).strip()

    if texto_limpio and not
```

```
texto_limpio.lower().startswith("nombre"):
        nombres.append({"nombre_cientifico": texto_limpio})
```

Esta extracción fue diseñada para ser reproducible y mantener la integridad de los datos en cada ejecución, permitiendo una actualización constante y automática a través de GitHub Actions.

B. Herramientas y tecnologías

A lo largo del desarrollo del proyecto se implementaron diversas tecnologías que permitieron la recolección, automatización, procesamiento y visualización de datos. A continuación, se detallan las herramientas empleadas y su función dentro del flujo de trabajo:

- Python 3.x
  Fue el lenguaje principal utilizado para el desarrollo de los scripts de scraping y procesamiento de datos. Su versatilidad permitió integrar diversas librerías para manejar HTML, estructuras de datos y automatización.

- Librerías:

  - **`requests:`** Se utilizó para enviar solicitudes HTTP a las páginas web objetivo desde las cuales se extrajo la información.
  - **`BeautifulSoup (bs4):`** Permitió analizar el contenido HTML de las páginas web y extraer datos específicos mediante etiquetas o atributos.
  - **`pandas:`** Se empleó para limpiar, estructurar y guardar los datos en formatos como .csv y .json.

- Git y GitHub
  Se utilizó Git como sistema de control de versiones para gestionar el código fuente, y GitHub como plataforma de hospedaje del repositorio. Esto permitió trabajar de forma organizada y mantener un historial completo del desarrollo del proyecto.

- GitHub Actions
  Se implementaron dos flujos de trabajo automatizados, estos workflows ejecutan los scripts de scraping de forma periódica sin intervención manual. Las acciones definidas:

- Instalan automáticamente las dependencias del proyecto (requirements.txt).
- Ejecutan los scripts desde la carpeta scripts/.
- Actualizan los archivos de datos en la carpeta data/.

- HTML, CSS y JavaScript
  Estas tecnologías se usaron para construir las páginas web del proyecto. Los datos extraídos se integran en estas páginas mediante scripts de JavaScript que permiten mostrar información de forma dinámica e interactiva.

- Visualización de Datos:
  Se utilizaron librerías como Plotly, Leaflet y componentes HTML interactivos para representar los datos en gráficos y mapas. Los resultados se almacenan en archivos .html ubicados en la carpeta **`graficos/`**.

Esta combinación de tecnologías permitió desarrollar un sistema completo que automatiza la extracción de datos, los procesa, los presenta en un sitio web interactivo y mantiene todo actualizado a través de acciones programadas.

C. Estructura del repositorio

```
TRABAJO_FINAL/
│
├── .github/workflows/
│   └── acceder_articulos.yml
│   └── actualizar_evento.yml
├── scripts
│   └── acceder_articulos.py
│   └── animales_lista.py
│   └── …
├── data
│   └── especies_animales.json
│   └── eventos_naturales.json
│   └── …
├── js
│   └── animales_extintos.js
│   └── articulos_animales.js
│   └── …
├── graficos
│   └── grafico_top_especies.html
│   └── mapa_calor_sudamerica.html
│   └── …
├── imagenes
```

```
│    └── amenazadas_total.jpg
│    └── ajolote.gif
│    └── img_sismo.png
│    └── …
├── requirements.txt
├── README.md
├── style.css
├── estadísticas.html
├── articulos.html
├── mapa.html
├── sobre_nosotros.html
└── index.html
```

El repositorio está organizado en carpetas y archivos que separan las diferentes funcionalidades del proyecto: automatización, extracción de datos, visualización y presentación web. A continuación se detalla cada componente:

`.github/workflows/:` Contiene los archivos YAML que configuran las acciones automatizadas con GitHub Actions.

`scripts/:` Incluye todos los scripts en Python responsables del scraping y procesamiento de datos.

`data/:` Almacena los archivos .json que contienen los datos extraídos y estructurados por los scripts.

`js/:` Contiene archivos JavaScript utilizados en la visualización de datos dentro de las páginas web.

`graficos/:` Incluye archivos HTML generados con gráficos interactivos, como visualizaciones de especies y mapas.

`imagenes/:` Carpeta que almacena los recursos visuales utilizados en el sitio, como imágenes, íconos y gifs.

Archivos del sitio web:

- **`index.html:`** Página principal.
- **`articulos.html, estadísticas.html, mapa.html, sobre_nosotros.html:`** Subpáginas con contenido temático.
- **`style.css:`** Hoja de estilos para el diseño visual del sitio.

Archivos auxiliares:

- `requirements.txt:` Lista de dependencias del proyecto en Python.
- `README.md:` Documentación general del proyecto.

Esta estructura modular permite mantener un flujo de trabajo claro, separar la lógica de backend (extracción y automatización) del frontend (visualización web), y facilita el mantenimiento del proyecto.

III. Desarrollo

A. Fuente de datos

- **Sitios web** (scraping):
  - Mongabay - Acceso a noticias sobre animales en peligro de extinción.
  - Perú_ecológico - Obtenemos las especies en peligro.
  - WIKIPEDIA: ESPECIES EXTINTOS - Obtención de los animales extintos.
  - Wikipedia - Obtener el título del artículo de un animal.

- **APIs públicas**:
  - GBIF API – Avistamientos y metadatos de especies.
  - Obtención del nombre común de un animal.
  - iNaturalist API – Traducción de nombres científicos a nombres comunes.
  - Wikipedia - Obtención de artículos por nombre científico.
  - - Obtención del nombre común de un animal.
  - Nominatim API: Obtener las regiones según coordenadas (latitudes y longitudes)

B. Diseño del scraping y estructuración de los datos

Flujo general del proceso:

1. Web Scraping y APIs → especies_animales.json
2. Web Scraping → articulos_Animales.js
3. Normalización y limpieza → especies_nom_coord.json
4. Enriquecimiento de datos → especies_peligro.json
5. Estadísticas y conteo por país → estadisticas_sudamerica.js
6. Visualización interactiva → mapas y gráficos HTML

Datos extraídos e integrados

Los primeros 4 datos están organizados en la carpeta data/ y las últimas 2 en js/. Archivos clave:

Archivo CSV / JSON/Js	Contenido
especies_animales.json	Lista original extraída del sitio web.
especies_nom_coord.json	Especies con coordenadas y nombres comunes.
especies_peligro.json	Especies clasificadas según nivel de amenaza.
eventos_naturales.json	Datos de eventos (incendios, deforestación, etc.).
estadisticas_sudamerica.js	Datos finales con especies, país y número de avistamientos.
articulos_Animales.js	Datos de artículos de animales en peligro de extinción (título, imagen, fecha de publicidad, url).

C. Descripción del código y lógica implementada

Dependencias

El archivo requirements.txt contiene todas las librerías utilizadas en el entorno del proyecto, como:

- requests
- beautifulsoup4
- pandas
- matplotlib
- folium
- plotly
- webdriver-manager
- Selenium

Código Documentado

Los scripts se encuentran en la carpeta scripts/ y están **modularizados** por tarea:

Script	Función
acceder_articulos.py	Extrae noticias relacionadas a especies en peligro de extinción desde la web.
traduccion_especies.py	Traduce los nombres de los animales de especies_animales.json por medio de coincidencias del nombre científico.
animales_lista.py hasta animales_listunica.py	Procesan diferentes fuentes para crear un único listado unificado de animales en peligro de extinción.
animales_nomcord.py	Añade coordenadas geográficas de cada animal en peligro de extinción.
animales_peligro.py	Consulta en wikipedia el estado de conservaciónde cada especie.
creando_graficos_por_region.py	Gráfica barras para cada región, coloreadas por zona (Costa/Sierra/Selva), mostrando cuántas especies en peligro han sido registradas por departamento.
Estadisticas_especies.py	Genera estadisticas_sudamerica.js con la estructura JSON final.
Estadisticas_regiones.py	Genera un .js con las estadísticas por departamento en Perú
eventos_naturales.py	Recopila eventos naturales recientes en el Perú

Cada script tiene comentarios que explican paso a paso su funcionamiento.

D. Automatización con GitHub Actions

En .github/workflows/ se incluyen dos archivos YML para automatizar:

Archivo	Descripción
acceder_articulos.yml	Ejecuta el scraping utilizado en acceder_articulos.py y actualiza los últimos artículos.
actualizar_evento.yml	Actualiza el archivo eventos_naturales.json con nuevos eventos.

Esto permite mantener los datos actualizados automáticamente cada cierto tiempo.

IV. Resultados y visualización

A. Visualización de Resultados

Los gráficos y mapas están en la carpeta graficos/. Entre ellos:

- grafico_top_especies.html: Muestra las 15 especies más avistadas en riesgo.
- mapa_calor_sudamerica.html: Mapa de calor por país según número de registros.
- tabla_estadisticas_paises.html: Tabla con resumen por país.
- grafico_por_grupo_horizontal.html: Visualización por tipo de especie (aves, mamíferos, etc.).

Todos estos archivos se visualizan desde las páginas web del proyecto.

B. Interfaz para Html

Script que se utilizaron para una mejor visualización de los datos en la web. Los archivos se encuentran dentro script/.

Archivo JS	Contenido
script.js	Muestra un carrusel de tarjetas con especies en peligro, permite buscarlas por nombre o descripción, corrige descripciones genéricas en inglés y permite navegar lateralmente con botones.
script_articulo.js	Muestra dinámicamente una lista de artículos en una página web y permite buscarlos por título.
script_extintos.js	Muestra un carrusel de animales extintos usando datos del archivo animales_extintos.js, creando una

	tarjeta por cada especie e incluyendo navegación con botones laterales.

C. Sitio Web y Visualización final

El sitio web está compuesto por:

- index.html: Página principal (buscador de animales en peligro de extinción y datos de animales extintos).
- estadisticas.html: Tabla y gráficos por país.
- mapa.html: Mapa interactivo de especies.
- articulos.html: Noticias relacionadas a animales en peligro de extinción.

V. Conclusiones

- Se logró integrar información de múltiples fuentes (Wikipedia, GBIF, iNaturalist, NASA FIRMS, USGS) para construir un panorama claro sobre la situación de las especies en peligro de extinción en Sudamérica y Perú.
- El proyecto combina scraping, APIs, procesamiento de datos y visualizaciones interactivas.
- El uso de GitHub Actions permite mantener los datos actualizados sin intervención manual.
- La entrega final incluye visualización web, códigos documentados, datos estructurados y análisis gráfico.