**1. Introducción**

Este proyecto consiste en una aplicación de escritorio desarrollada en Python, utilizando **Tkinter** para la interfaz gráfica y **PLY (Python Lex-Yacc)** para la generación del **analizador léxico**. Su objetivo es validar archivos JSON estructurados que describen equipos de trabajo, sus integrantes, proyectos, y otras características.

**2. Objetivos del Sistema**

* Permitir la carga de archivos JSON mediante interfaz gráfica.
* Analizar léxicamente el contenido del archivo, extrayendo tokens válidos.
* Identificar errores léxicos y señalarlos al usuario.
* Mostrar el resultado del análisis de forma clara y organizada.

**3. Componentes del Sistema**

**3.1. Librerías Utilizadas**

* tkinter: Para construir la interfaz gráfica.
* ply.lex: Para definir el analizador léxico (scanner).
* filedialog, scrolledtext, messagebox: Widgets auxiliares de Tkinter.
* decimal: Para el redondeo de números flotantes.

**4. Tokens Definidos**

El lexer reconoce **más de 60 tokens** relacionados con estructuras específicas de un archivo JSON semánticamente controlado. Entre los más importantes se encuentran:

* **Delimitadores**: {, }, [, ], ,
* **Claves JSON**: "equipos":, "nombre\_equipo":, "integrantes":, "email":, etc.
* **Valores especiales**: true, false, null, "" (VACÍO)
* **Tipos de datos**:
  + DATE (Formato YYYY-MM-DD)
  + EMAIL (Formato válido de email)
  + URL (Protocolos http o https)
  + STRING (Cualquier texto entre comillas dobles)
  + NUMERO (Enteros)
  + FLOAT (Decimales con dos dígitos)
* **Valores controlados**: "To Do", "In Progress", "Canceled", "Done", "On Hold", "Project Manager", "UX Designer", etc.

**5. Manejo de Errores**

* Se implementa una función t\_error() que captura cualquier carácter no reconocido.
* Los errores se almacenan en una lista errores[] y se marcan visualmente en el área de texto.
* El sistema resalta las líneas con errores con fondo color **misty rose**.
* Se notifica al usuario mediante un messagebox si hubo errores o si el análisis fue exitoso.

**6. Interfaz Gráfica**

**6.1. Ventana Principal**

* Título: "Analizador Léxico"
* Tamaño: 1000x650 píxeles

**6.2. Componentes Principales**

|  |  |
| --- | --- |
| **Componente** | **Función** |
| **Botón "Cargar JSON"** | Permite seleccionar un archivo .json desde el sistema. |
| **Área de entrada** | Muestra el contenido del archivo cargado. Permite editarlo. |
| **Botón "Procesar JSON"** | Ejecuta el análisis léxico del contenido. |
| **Área de salida** | Muestra los tokens reconocidos. |
| **Área de errores** | Muestra errores léxicos detectados. |

**7. Funciones Clave**

**7.1. cargar\_archivo()**

* Abre un diálogo para seleccionar un archivo JSON.
* Carga su contenido en el área de texto principal.

**7.2. obtener\_json()**

* Ejecuta el análisis léxico sobre el contenido.
* Muestra los tokens encontrados.
* Si hay errores, los muestra en un panel inferior.

**7.3. marcar\_linea\_error(linea)**

* Marca visualmente en el área de texto la línea donde ocurrió un error.

**8. Flujo del Programa**

1. El usuario abre la app.

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

1. Carga un archivo JSON.

Interfaz de usuario gráfica, Aplicación, Word

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

1. Presiona "Procesar JSON".

Texto

El contenido generado por IA puede ser incorrecto.

1. El sistema ejecuta el lexer:
   * Si no hay errores → muestra mensaje de éxito.

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

* + Si hay errores → se listan en la parte inferior y se marca la línea correspondiente.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

* + En ambos casos muestra los tokens y sus valores.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

1. El usuario puede corregir y volver a procesar.

**8. Flujo del Programa:**

Si un atributo entre comillas en el JSON **no coincide exactamente** con ninguno de los tokens definidos por expresiones regulares, pero **está correctamente entre comillas** y **no contiene caracteres ilegales**, entonces el analizador léxico lo interpreta como un token de tipo STRING.  
**Esto no se considera un error léxico**, ya que STRING es un token válido.





En cambio, **si aparece un carácter que no forma parte de ningún token definido** (por ejemplo, una comilla sin cerrar, un símbolo especial no esperado, o una palabra clave mal escrita y sin comillas), entonces se considera un **carácter ilegal**, y se reporta como error.

**Ejemplo:**  
Si el JSON contiene "identidad": " https://i.postimg.cc/zvQcPBms/LogoSSL.jpg"  
y no existe un token definido como t\_TOK\_IDENTIDAD, entonces:

* "identidad" se interpreta como un STRING. No lo va a tomar como error porque STRING es un token válido.
* Los dos puntos se interpretan como un Carácter ilegal.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.