Reporte de Desarrollo

Introducción

Este reporte describe el proceso de desarrollo de un script en Python que lee y valida un archivo JSON que contiene los tweets. El objetivo del script es verificar que el contenido del archivo siga una estructura adecuada y tambien leer los tweets almacenados en el archivo

Estructura del script

El desarrollo del script se divide en cuatro partes principales:

- apertura del archivo

```
with open('reducido.json', 'r') as archivo:
    datos = json.load(archivo)
```

- definición del esquema json

```
esquema = {
 "$schema": "http://json-schema.org/draft-07/schema#",
 "type": "array",
 "items": {
   "type": "object",
```

```
  "properties": {
    "id": {
      "type": "string"
    },
    "texto": {
      "type": "string"
    },
    "usuario": {
      "type": "string"
    },
    "hashtags": {
      "type": "array",
      "items": {
        "type": "string"
      }
    },
    "fecha": {
      "type": "string",
      "format": "date-time"
```

```
      },
      "retweets": {
        "type": "number"
      },
      "favoritos": {
        "type": "number"
      }
    },
    "required": [
      "id",
      "texto",
      "usuario",
      "hashtags",
      "fecha",
      "retweets",
      "favoritos"
    ]
  }
}
```

```

```

- validación de los datos con el esquema definido

```
try:

    jsonschema.validate(instance=datos, schema=esquema)
    print("El JSON cumple con el esquema")
except jsonschema.exceptions.ValidationError as e:
    print(f"Error de validación: {e.message}")
```

- impresión del contenido por consola

```
for item in datos:
```

```
    for clave, valor in
item.items():
        print(f"Clave: {clave},
Tipo de valor: {type(valor)}")
```

Conclusiones

El script desarrollado permite leer y validar un archivo JSON de manera eficiente utilizando la librería jsonschema, manejando los errores comunes que podrían ocurrir en el proceso de validación mediante el uso de excepciones.