**Ejemplo para identificar la API**

Ejemplos de rutas, métodos HTTP y parámetros comunes que podemos encontrar al interactuar con el sitio web:

1. **Inicio de sesión:**
   * Ruta: **/login**
   * Método HTTP: POST
   * Parámetros: Datos de formulario (nombre de usuario y contraseña), posiblemente encabezados para autenticación.
2. **Registro:**
   * Ruta: **/register**
   * Método HTTP: POST
   * Parámetros: Datos de formulario (nombre de usuario, contraseña, dirección de correo electrónico, etc.).
3. **Búsqueda de cursos:**
   * Ruta: **/courses/search**
   * Método HTTP: GET
   * Parámetros: Query string para los términos de búsqueda (p. ej., **?q=testing**).
4. **Visualización de un curso:**
   * Ruta: **/courses/{id}**
   * Método HTTP: GET
   * Parámetros: ID del curso en la URL.
5. **Actualización de perfil de usuario:**
   * Ruta: **/profile/{id}**
   * Método HTTP: PUT
   * Parámetros: ID de usuario en la URL, datos JSON en el cuerpo de la solicitud para actualizar el perfil.
6. **Eliminar cuenta de usuario:**
   * Ruta: **/profile/{id}**
   * Método HTTP: DELETE
   * Parámetros: ID de usuario en la URL, posiblemente confirmación en el cuerpo de la solicitud.

**Caso real**

1. **Abrir el sitio web:** Accede al sitio web <https://www.ministryoftesting.com/> en tu navegador.
2. **Utilizar el Inspector de Elementos:** Haz clic derecho en cualquier parte de la página web y selecciona "Inspeccionar" o "Inspeccionar elemento" en el menú contextual. Esto abrirá las herramientas de desarrollo de tu navegador.
3. **Navegar por el sitio:** Explora diferentes secciones del sitio web, realiza acciones como iniciar sesión, buscar contenido, ver perfiles de usuario, etc.
4. **Analizar las solicitudes de red:** En las herramientas de desarrollo, ve a la pestaña "Network" o "Red" (puede variar según el navegador). Aquí verás una lista de todas las solicitudes HTTP realizadas por el sitio web.
5. **Filtrar las solicitudes:** Puedes filtrar las solicitudes por tipo (XHR para solicitudes AJAX, Documentos para solicitudes de página, etc.). Esto te ayudará a centrarte en las solicitudes relevantes para la API o las interacciones que deseas analizar.
6. **Inspeccionar cada solicitud:** Haz clic en una solicitud específica para ver detalles como la ruta, el método HTTP utilizado y los parámetros. Esto te dará una idea de cómo se comunican con el servidor.
7. **Documentar las rutas y métodos:** A medida que identifiques las rutas y métodos utilizados, documenta esta información junto con cualquier parámetro relevante.
8. **Configurar tus pruebas de carga y rendimiento:** Con la información recopilada, podrás configurar tus escenarios de prueba en Plawright y Artillery para simular la carga en las rutas específicas de la API del sitio web.

Se crea un archivo: artillery-config.yml

config:

  target: 'https://www.ministryoftesting.com'

  phases:

    - duration: 60 # Duración de la prueba en segundos (1 minutos)

      arrivalRate: 10 # tasa para aumentar o reducir la carga en tu aplicación durante la prueba.

      rampTo: 5 # La tasa de llegada aumentará gradualmente hasta llegar a 5 durante la duración de la fase.

# ejemplo que muestra desde 6

scenarios:

  - flow:

      - post:

          url: 'https://cdn.getgist.com/translation\_files/en\_translation.json'

          json: '{}'

            userId: '{{ \_\_random(1, 10) }}'

            key1: 'value1'

            title: 'Carga de múltiples publicaciones' # modulo 3 despues del LOg in OK

            body: 'This is the body of the test post'

        think: 1

  - flow:

      - get:

          url: 'https://cdn.getgist.com/translation\_files/en\_translation.json'

            title: 'Carga de imágenes' # modulo 1 inicio

            body: 'This is the body of the test get'

        think: 2

  - flow:

      - get:

          url: 'https://messenger.getgist.com/projects/zui7pcpe/articles?random\_id=83b7f1b1-e720-ba58-3aa9-c9fe12d6ef6d&search\_text=&page=1'

            title: 'Carga de comentarios' #  modulo 4 despues del LOg in OK, iniciar chat

            body: 'This is the body of the test get'

        think: 3