Laboratorio - Explore las API de descanso con el Simulador API y Postman

Objetivos

Parte 1: Inicie el DevNet VM

Parte 2: Explore la documentación de API usando el simulador de API

Parte 3: Use el Postman para realizar llamadas API al simulador de API

Parte 4: Use Python para agregar 100 libros para el simulador de API

Aspectos básicos/Situación

La máquina virtual de DevNet VM incluye un simulador de API de biblioteca escolar con documentación de API y una base de datos asociada. Usted puede utilizar el simulador sin conexión para explorar las API y probar su funcionalidad.

En este laboratorio, usted aprenderá a usar el simulador de API de la biblioteca escolar para realizar llamadas a la API para enumerar, agregar y eliminar libros. Más tarde, usarás Postman para realizar estas mismas llamadas a la API.

Recursos necesarios

- 1 PC con sistema operativo de su elección
- Virtual Box o VMWare
- Máquina virtual (Virtual Machine) DEVASC

Instrucciones

Parte 1: Iniciar la Máquina virtual de DEVASC

Si usted no ha completado el laboratorio **- Instale el Entorno de Laboratorio de la Máquina Virtual**, hágalo ahora. Si usted lo ha completado ya, inicie la máquina virtual DEVASC.

Parte 2: Explore la documentación API usando el simulador de API

Para entender cómo realizar llamadas a una API REST, los desarrolladores suelen empezar estudiando la documentación de la API. El formato para las solicitudes, respuestas, encabezados y parámetros para las API REST se documentan normalmente mediante la especificación OpenAPI (anteriormente la especificación Swagger).

Paso 1: Abrir el Web Browser Chromium.

Haga doble clic al ícono de Web Browser Chromium en el escritorio

Paso 2: Conéctese al sitio Web de la Biblioteca de la Escuela.

Si el navegador no ha abierto automáticamente el sitio web de la Biblioteca de la Escuela, en la barra de direcciones escriba: **library.demo.local** y presione volver para ir allí.

Paso 3: Vaya a la página de documentos de API.

a. El sitio web se establece de forma predeterminada en la pestaña **Nuestros libros** y muestra una lista de libros. En la esquina superior derecha donde indica, **Haga clic aquí para ver documentos de API**, haga clic en **aquí** para ir a la página web de documentación de API.

   Ahora usted verá una lista de APIs en el espacio de **nombre predeterminado /api/v.1**.

b. Observe la flecha hacia abajo a la derecha. Al hacer clic en cualquier parte de la barra **/api/v1** se minimiza la lista de API y se girará la flecha hacia la derecha. Haga clic de nuevo en la misma barra para volver a mostrar la lista de API.

   Observe el bloqueo a la derecha de varias de las API. El bloqueo indica que estas API puede requerir un token para ser utilizado.

Paso 4: Enumera libros con la API GET /books.

Haga clic en cualquier parte de la barra de la API **GET /books** Esta API vuelve a una lista de libros de la biblioteca escolar.

- **Parámetros** - Hay varios parámetros de API opcionales. Estos se pueden usar para filtrar, ordenar o paginar la salida. Estos serán referidos más adelante en este laboratorio.
- **Tipo de contenido de respuesta** - clic en **aplicación/JSON** para ver una lista de los diferentes tipos de formatos de datos que se puede visualizar en la información. Deje la selección como **aplicación/JSON**.
- **Código** - El código muestra 200 de forma predeterminada, lo que indica que la solicitud de API del servidor fue un éxito como se muestra en la **descripción**. (Usted todavía no ha enviado una solicitud de API).

Paso 5: Utilice la función Probar en la documentación de la API.

Una de las características más potentes de la especificación API Abierta (OpenAPI) es la capacidad de probar una llamada API para ver si la construyó correctamente. Usted también puede revisar la respuesta para ver si es lo que esperaba. Usted verá esta misma característica de prueba en la documentación de API para Cisco, MapQuest y otras organizaciones que utilizan esta característica de especificación OpenAPI.

a. En la documentación de la API **GET /books** haga clic en el botón **Probar**

b. Observe que ahora tiene la opción de introducir información para los parámetros opcionales. Deje los parámetros en blanco y haga clic en el botón **Ejecutar**

   En la sección **Respuestas** usted verá:

   - **CURL:** El comando CURL que puede utilizar para tener acceso a la misma información para la API /books.
   - **URL de solicitud:** esta URL es usada en la solicitud de API, que se puede utilizar para solicitar la misma información usando CURL, Postman y Python.
   - **Código**: Este es el código de respuesta HTTP. 200 indica una llamada exitosa.
   - **Cuerpo de respuesta**: Lista de libros en formato JSON.
   - **Encabezados de respuesta**: información sobre la API devuelta desde el servidor.

   En el **cuerpo de la respuesta** usted verá una lista de libros en formato JSON:

```
[
    {
        «ID":0,
        "Título": "Fundamentos del enrutamiento IP",
        "author": «Mark A. Sportack»
```

```
    },
    {
        "ID":1
        «Título» «Python para tontos»
        «author» «Stef Maruch Aahz Maruch»
    },
    {
        "id"2
        «Título» «Linux para Trabajadores de NET»
        «author» «Cisco Systems Inc»
    },
    {
        «id» 3,
        «Título» «NetACad: 20 años de aprendizaje en línea»
        «author» «Cisco Systems Inc»
    }
]
```

Paso 6: Utilice el comando CURL en una ventana terminal.

La API **GET /books** proporciona información para acceder al contenido mostrado en el cuerpo de respuesta mediante CURL. CURL es una herramienta de línea de comandos para transferir datos hacia o desde un servidor, utilizando cualquiera de los protocolos admitidos, incluidos HTTP y HTTPS.

a. Seleccione el comando CURL, haga clic con el botón derecho y **cópielo** en su porta papeles:

```
Curl -X GET "http://library.demo.local/api/v.1/books" -H «accept: application/JSON»
```

Abran una ventana del terminal. Haga clic con el botón derecho y **Pegue** los contenidos del porta papeles en el terminal y pulse Intro. Observe que esto proporciona la misma información que la interfaz OpenAPI de la biblioteca.

```
devasc @labvm: ~$ CURL -X GET "http://library.demo.local/api/v1/books" -H «aceppt: application/JSON»
[
    {
        «ID» 0,
        "Título": "Fundamentos del enrutamiento IP",
        «author",: «Mark A. Sportack»
    },
    {
        "id":1
        «Título» «Python para tontos»
        «author» «Stef Maruch Aahz Maruch»
    },
    {
        "id"2
        «título» «Linux para Trabajadores de NET»
        «author» «Cisco Systems Inc»
    },
    {
        «id» 3,
```

```
        «título» «NetAcad: 20 años de aprendizaje en línea»
        «author» «Cisco Systems Inc»
    }
]
devasc@labvm:~$
```

Paso 7: Enumera libros con su ISBN utilizando la API GET /books.

a. Vuelva a la API GET /books del sitio web de la biblioteca escolar.

b. En la sección **Parámetros**, seleccione la flecha abajo junto al parámetro **include ISBN** parámetro y seleccione **verdad (true)**

c. Haga clic en **Ejecutar**.

Observe los siguientes cambios en **Respuestas**:

- **CURL** ahora incluye el parámetro para ISBN.

```
curl -X GET "http://library.demo.local/api/v1/books?includeISBN=true"-H
«accept: application/json»
```

- **La URL de solicitud** ahora incluye el parámetro para ISBN.

```
http://library.demo.local/api/v1/books?IncludeIsBN=True
```

- **El cuerpo** de la respuesta tiene la misma lista de libros que se muestra anteriormente, pero ahora incluye el ISBN del libro.

Para minimizar el desplazamiento, cuando haya terminado con una API, puede cerrar esa ventana específica de API haciendo clic en cualquier parte de la barra de título. Ahora puedes ver todas las API más fácilmente.

Paso 8: Obtener un token usando la API POST /LoginViaBasic.

a. Haga clic en la **API POST /LoginViaBasic**.

b. Observe que no hay parámetros. Haga clic en **Probar** y, a continuación, haga clic en **Ejecutar**.

c. Un cuadro de **Iniciar sesión** le pedirá a usted un **nombre de usuario** y **contraseña** Ingrese la siguiente información y haga clic en **Iniciar sesión**

- **Nombre de Usuario**: cisco
- **Contraseña**: Cisco123!

d. El token se mostrará en el **cuerpo de la respuesta**. Seleccione la información entre las comillas, haga clic con el botón derecho y **Copie** la información en su portapapeles. **Su token será diferente de la que se muestra a continuación**.

```
{
  ''token»: "cisco|Kzzzteqbc5iv3hkezb7hcj6qhqxen4rlgh72yjkevfs"
}
```

e. Desplázate hacia arriba hasta la parte superior de la página API de la biblioteca escolar y haz clic en el botón verde **Autorizar**. Aparecerá el cuadro de diálogo **Autorizaciones disponibles**.

f. Haga clic con el botón derecho y **pegue** el token después de **Valor** y haga clic en **Autorizar** Observe que el **nombre** es X-API-KEY. Esta información junto con el token se usará más adelante en Postman.

g. Cierre el cuadro de diálogo **Autorizaciones disponibles** y vuelva a la lista de API. Observe que los bloqueos de varias de las API han cambiado ahora. Estas API ya están disponibles para su uso.

h. Haga clic en la barra de API **POST /LoginViaBasic** para cerrar la ventana.

Paso 9: Añada libros mediante la API POST /books.

a. Haga clic en la API **POST /books**.

b. Observe en **Parámetros** que la **carga útil** es requerida. Esto significa que esta API debe requerir información para este parámetro en el formato especificado por el **tipo de contenido Parameter**, que es JSON.

c. Haga clic en **Probar**.

d. Modifique el **ID**, el **título** y el **autor** con la información que se muestra a continuación.

```
{
  «id»: 4,
  «título»: "Fundamentos de IPv6«,
  «author»: "Rick Graziani»
}
```

e. Haga clic en **Ejecutar**.

f. Compruebe que la publicación se ha realizado exitosamente en la respuesta del servidor. Un **código** de 200 significa que la publicación fue un éxito. Usted debería ver el libro que acaba de agregar en el **cuerpo de la respuesta** junto con un nuevo **ID**. También verá información actualizada sobre **CURL** y la **URL de solicitud**.

g. Para añadir otro libro, modifique el **ID**, el **título** y el **autor** con la información que se muestra a continuación.

```
{
  "ID": 5.
  «título»: "31 días antes de su examen CCNA«,
  «author»: "Allan Johnson»
}
```

h. Haga clic en **Ejecutar**.

i. Verifique que la publicación se ha realizado exitosamente en la **respuesta del servidor**. Un **código** de 200 significa que la publicación fue un éxito. Usted debería ver el libro que acaba de agregar en el **cuerpo de la respuesta** junto con un nuevo **id**. También verá información actualizada sobre **CURL** y la **URL de solicitud**.

Nota: Si tienes un código **401**,comprueba el texto del cuerpo de la **respuesta. Lo más probable es que haya recibido una respuesta de «error»: «Clave API inválida».** Esto se debe a que usted no ingresó todos los caracteres para su clave API. O posiblemente, agregar un espacio innecesario. Vuelva al paso anterior y repita el proceso de autorización.

j. Haga clic en la barra de **API POST /books** para cerrar la ventana.

k. Usted puede verificar los libros que se agregaron a la página de **Nuestros libros**. Vuelva a la pestaña **Biblioteca escolar** en su navegador (http://library.demo.local) y actualice la página. Tenga cuidado de no cerrar la pestaña API de la biblioteca escolar. Si lo hace, tendrá que volver a autenticarse.

Paso 10: Enumera libros con la API GET /books.

a. Vuelva a la pestaña **API de la biblioteca escolar** en el navegador. Haga clic en la API **GET /books**.

b. Haga clic en **Probar** Si usted ve el botón **Cancelar** en rojo, entonces ya está en el modo **Prueba**.

c. Haga clic en **Ejecutar**

d. Debajo de **Respuesta del servidor** en el **cuerpo de la respuesta**, ahora verá los dos libros que agregó. Observe que cada uno tiene una **identificación**única.

```
[
  {
    «ID»: 0,
    "Título": "Fundamentos del enrutamiento IP",
    «author",: «Mark A. Sportack»
  },
  {
    "id":1
    «Título» «Python para tontos»
    «author» «Stef Maruch Aahz Maruch»
  },
  {
    "ID"2
    «título»: «Linux para Trabajadores de NET»
    «author» «Cisco Systems Inc.»
  },
  {
    «id» 3,
    "título» «NetAcad: 20 años de aprendizaje en línea"
    «author» «Cisco Systems Inc.»
  },
  {
    «id»: 4,
    "título»: "Fundamentos de IPv6",
    "author»: "Rick Graziani"
  },
  {
    "id": 5.
    "título»: "31 días antes de su examen CCNA",
    «author»: "Allan Johnson»
  }
]
```


e. Haga clic en la barra de la API **GET /books** para cerrar la ventana.

Paso 11: Enumera un libro específico utilizando la API GET /books {id}.

a. Haga clic en la API **GET /books {id}**. Observe que esta API requiere el **id** como parámetro.

b. A la derecha de Parámetros, haga clic en el botón **Probar**.

c. En **Parámetros**, escriba 4 para el identificador requerido.

d. Haga clic en **Ejecutar** Observe la información proporcionada por **CURL** y **URL de solicitud**.

- **CURL** - Este es el comando CURL para realizar la misma función usando CURL.
- **URL de solicitud** - Esta es la URL que se puede usar para obtener la misma información usando Postman y Python.

Compruebe que el resultado fue correcto en la **respuesta del servidor**. Un **código** de 200 significa que la publicación fue un éxito. En el **cuerpo de la respuesta** verá el libro que solicitó con el **id** de 4.


```
{
  «id»: 4,
  «título»: "Fundamentos de IPv6«,
```

```
  «author»: "Rick Graziani»
}
```

e. Haga clic en la barra de la API **GET /books {id}** para cerrar la ventana.

Paso 12: Elimine un libro específico con la API DELETE /books {id}.

a. Haga clic en la API **DELETE /books {id}**. Observe que esta API requiere el **id** como parámetro.

b. Haga clic en **Probar**

c. En **Parámetros**, escriba 4.

d. Haga clic en **Ejecutar**

e. Verifique que la eliminación se realizó correctamente en la **respuesta del servidor.** Un **código** de 200 significa que la publicación fue un éxito. En el **cuerpo de la respuesta** usted verá el libro que eliminó con el **id** de 4.

```
{
  «id»: 4,
  «título»: "Fundamentos de IPv6«,
  «author»: "Rick Graziani»
}
```

f. Haga clic en la barra de la API **DELETE /books {id}** para cerrar la ventana.

Paso 13: Enumera libros con la API GET /books.

a. Haga clic en el API **GET /books**

b. Haga clic en **Probar**. Si usted ve el botón **Cancelar** en rojo, entonces ya estás en el modo **Prueba**.

c. Haga clic en **Ejecutar**.

d. En **Respuesta del servidor** en el **cuerpo de la respuesta**, ya no verá el libro con **id** o 4.

```
[
  {
    «id»: 0,
    "Título": "Fundamentos del enrutamiento IP",
    «author",: «Mark A. Sportack»
  },
  {
    "id":1
    «Título» «Python para tontos»
    «author» «Stef Maruch Aahz Maruch»
  },
  {
    "id"2
    «título»: «Linux para Networkers»
    «author» «Cisco Systems Inc.»
  },
  {
    «id» 3,
    «título» «NetAcad: 20 años de aprendizaje en línea»
    «author» «Cisco Systems Inc.»
  },
  {
```

```
    "id": 5.
    «título»: "31 días antes de su examen CCNA«,
    «author»: "Allan Johnson»
  }
]
```

Nota: No cerrar la pestaña **API de la biblioteca escolar** en el navegador Chromium. Usted utilizará la documentación de la API en la siguiente parte.

Parte 3: Usar Postman para realizar llamadas API al Simulador de API

En esta parte, usted usará Postman para realizar las mismas llamadas API que realizó en la documentación de la API de la biblioteca de estudiantes. Postman es una herramienta útil cuando un sitio web de desarrolladores de API no está disponible, al tiempo que proporciona la habilidad de guardar, organizar y reutilizar API fácilmente.

Paso 1: Abra Postman.

Haga doble clic en el icono de Postman en el escritorio normalmente, iniciaría sesión en Postman. Sin embargo, no es necesario obtener una cuenta e iniciar sesión en Postman para laboratorios en este curso.

Paso 2: Enumere los libros utilizando la API GET /books.

a. En la ventana principal junto a la pestaña **Launchpad**, haga clic en el icono más «+» para crear una **solicitud sin título**. De forma predeterminada, esta será una solicitud **GET**.

b. Haga clic en la flecha abajo situada junto a **GET** para ver las diferentes operaciones de API, incluidas GET, POST y ELIMINAR (DELETE). Deje la selección en **GET**. Haga clic en la flecha hacia arriba situada junto a **Obtener** para cerrar la lista.

c. Introduzca la URL solicitada.

1) Vuelva a la pestaña **API de la biblioteca escolar** en Chromium y, si es necesario, expanda la API **GET /books**.

2) En **Solicitar URL**, seleccione, haga clic con el botón derecho y **Copiar** la URL en el portapapeles:

`http://library.demo.local/api/v1/books`

3) Vuelva a **Postman** y pegue la URL junto a GET donde indica, «Introducir URL de solicitud».

Nota: Si pegando agrega una línea debajo de la URL, elimine la línea adicional.

d. Haga clic en **Send** (Enviar). Para verificar que la solicitud de API fue correcta, ahora verá una respuesta que incluye el código de **estado** 200 OK en verde. Desplácese hacia abajo hasta la sección **Cuerpo** para ver la respuesta. Observe que el valor predeterminado es **Pretty** y **JSON**.

```
[
    {
        «ID»: 0,
        "Título": "Fundamentos del enrutamiento IP",
        «author",: «Mark A. Sportack»
    },
    {
        "ID":1
        «Título» «Python para principiantes»
        "author» «Stef Maruch Aahz Maruch"
    },
    {
```

```
        "ID"2
        «título»: «Linux para Trabajadores de NET»
        «author» «Cisco Systems Inc.»
    },
    {
        «id» 3,
        «título» «NetAcad: 20 años de aprendizaje en línea»
        «author» «Cisco Systems Inc.»
    },
    {
        "ID": 5.
        «título»: "31 días antes de su examen CCNA«,
        «author»: "Allan Johnson»
    }
]
```

Nota: Puede guardar la salida JSON en un archivo usando el botón **Guardar** respuesta encima de la salida. Esto no es requerido para este laboratorio.

Paso 3: Obtenga un token usando la API POST /LoginViaBasic.

a. En la ventana principal, haga clic en el icono más «+» para crear una nueva **solicitud sin título**.

b. Haga clic en la flecha hacia abajo junto a **GET** y seleccione **POST**.

c. Introduzca la URL requerida

1) Vuelva a la pestaña **API de la biblioteca escolar** en Chromium y expanda la API **POST /LoginViaBasic**, si es necesario.

2) En **Solicitar URL**, seleccione, haga clic con el botón derecho y **Copiar** la URL en el portapapeles:

`http://library.demo.local/api/v1/loginViaBasic`

Nota: Si la **URL de solicitud** ya no se muestra, es probable que haya cerrado y vuelto a abrir la **página** de documentación de la API de la biblioteca escolar y ya no esté autenticada. Haga clic en **Probar**, luego **Ejecutar**, y luego volver a autenticar con el nombre de usuario **cisco** y contraseña **Cisco123!**.

3) Vuelva a **Postman** y pegue la URL junto a POST donde dice «Introducir URL de solicitud».

Nota: Si pegando agrega una línea debajo de la URL, elimine la línea adicional.

d. Haga clic en **Autorización** Dentro de esta área, complete lo siguiente:

1) En la lista desplegable de **Tipo**, elija **Autenticación (Auth) básica**.

2) Para los campos **Nombre de usuario** y **Contraseña**, rellene lo siguiente:

- **Nombre de usuario:** cisco
- **Contraseña:** Cisco123!

e. Haga clic en **Send** (Enviar).

f. Si es necesario, desplácese hacia abajo hasta la sección **Cuerpo** para ver su nuevo token token. **Su token será diferente de la que se muestra aquí.**

```
{
    «token»: «Cisco|5xSUHYFDVIAOCRV0LQWVSDCJJAWWJG18VMML6U2LM1i»
}
```

Paso 4: Añadir un libro mediante la API POST /books.

Ahora debe agregar el libro *Fundamentos de IPv6* que eliminó en la Parte 2 cuando utilice la función **Probar** en la documentación de la **API de la Biblioteca escolar**.

a. En la ventana principal, haga clic en el icono más «+» para crear una **solicitud sin título**.

b. Haga clic en la flecha hacia abajo junto a **GET** y seleccione **POST**.

c. Introduzca la URL solicitada.

1) Vuelva a la pestaña **API de la biblioteca escolar** en Chromium y expanda la API **POST /books**.

2) En **Solicitar URL**, seleccione, haga clic con el botón derecho y **Copiar** la URL en el portapapeles:

`http://library.demo.local/api/v1/books`

Nota: Si la **URL de solicitud** ya no se muestra, probablemente canceló **Pruébalo**. Haga clic en **Probar** y, a continuación, Ejecutar para mostrar la **URL de solicitud**.

3) Vuelva a **Postman** y pegue la URL junto a POST donde dice «Introducir URL de solicitud».

Nota: Si pegando agrega una línea debajo de la URL, elimine la línea adicional.

d. Haga clic en **Autorización** Dentro de esta área, complete lo siguiente:

1) En la lista desplegable **Tipo**, elija **Clave API**.

2) En el campo **Clave**, escriba **X-API-KEY**.

Nota: Recuerde que vio **X-API-KEY** en la página web de la API de la biblioteca escolar cuando obtuvo un token seleccionando el botón verde **Autorizar**.

3) Vuelva a la pestaña **Post** en Postman y copie el token que recibió en el paso 3. Asegúrese de incluir todo dentro de las comillas. Su token será diferente de la que se muestra aquí.

Ejemplo: `Cisco|5xSUHYFDVIAOCRV0LQWVSDCJJAWWJG18VMML6U2LM1i`

4) Vuelve a la segunda pestaña **Post** en Postman. Pegar el token en el campo **Valor**

e. En la misma fila con la **Autorización**, haga clic en **Cuerpo**. Esta sección le permitirá a usted elegir el formato de su entrada.

- Haga clic en el botón de radio **RAW**.
- Haga clic en **Texto** y cambie esta opción a **JSON**.

f. En el área de entrada verá el número 1, para “línea 1”. Ingrese el siguiente objeto JSON.

```
{
  «id»: 4,
  «título»: "Fundamentos de IPv6«,
  "author»: «Rick Graziani",
  "isbn»: «978 158144778"
}
```

g. Haga clic en **Send** (Enviar).

h. Para verificar que la solicitud de API fue correcta, ahora verá una respuesta que incluye el código de **estado** 200 OK en verde.

Paso 5: Verifique el libro adicional con la API Get /books.

a. Vuelva a la primera pestaña GET. Como puede ver, Postman hace que sea fácil cambiar entre diferentes llamadas de API.

b. Haga clic en **Send** (Enviar).

c. Para verificar que la solicitud de API fue correcta, ahora usted verá una respuesta que incluye el código de **estado** 200 OK en verde.

d. Haga clic en **Cuerpo** para ver la respuesta. Observe que el predeterminado es **Pretty** y **JSON**.

```
[
    {
        «ID»: 0,
        "Título": "Fundamentos del enrutamiento IP",
        «author",: «Mark A. Sportack»
    },
    {
        "ID":1
        «Título» «Python para tontos»
        «author» «Stef Maruch Aahz Maruch»
    },
    {
        "ID"2
        «título»: «Linux para Trabajadores de NET»
        «author» «Cisco Systems Inc.»
    },
    {
        «id» 3,
        «título» «NetAcad: 20 años de aprendizaje en línea»
        "author» «Cisco Systems Inc."
    },
    {
        «id»: 4,
        «título»: "Fundamentos de IPv6«,
        «author»: "Rick Graziani»
    }
    {
        "id": 5.
        "título»: "31 días antes de su examen CCNA",
        "author»: "Allan Johnson"
    },
]
```

Paso 6: Utilice parámetros adicionales con el API Get /books.

a. Vaya al sitio web de la **API de la biblioteca escolar** . Desplácese hacia arriba hasta **GET /books** API y expandirlo, si es necesario.

   Observe los parámetros que están disponibles:

   - **IncludeISBN**: Incluye en los resultados los números ISBN. Predeterminado (Default)=Falso
   - **SortBy**: Ordena los resultados utilizando el parámetro especificado. Predeterminado (Default)=ID
   - **Autor**: Devolver sólo los libros del Autor dado.
   - **Page** : Se utiliza para especificar un número de página.

b. Haga clic en **Probar**. Si ve un botón **Cancelar** en rojo, entonces no necesita seleccionar este botón.

c. Bajo parámetros:

- Haga clic en **IncludeISBN** y seleccione **true**
- Haga clic en **OrdenarPor** y seleccione **autor**

d. Haga clic en **Ejecutar**.

e. En el **cuerpo de la respuesta** verá la lista de libros ahora ordenados por autor e incluyendo los ISBN.


```
[
    {
        "id": 5.
        "título": "31 días antes de su examen CCNA",
        "author": "Allan Johnson"
    },
    {
        "id"2
        "título": "Linux para Trabajadores de Net"
        "author": Cisco Systems Inc.",
        "isbn": "000-0000000123"
    },
    {
        «id» 3,
        "título" "NetAcad: 20 años de aprendizaje en línea"
        "author": "Cisco Systems Inc.",
        "isbn": «000-0000001123"
    },
    {
        "ID": 0,
        "Título": "Fundamentos del enrutamiento IP",
        "author": «Mark A. Sportack",
        "isbn": "978-1578700714"
    },
    {
        "id": 4,
        "título": "Fundamentos de IPv6",
        "author": "Rick Graziani",
        "isbn": "978 1587144778"
    },
    {
        "ID":1
        "Título": "Python para principiantes"
        "author": "Stef Maruch Aahz Maruch",
        "isbn": "978-0471778646"
    }
]
```


Observe que la **URL de solicitud** ahora incluye los parámetros. Verás esto de nuevo en Postman.

`http://library.demo.local/api/v1/books?includeIsBN=true&sortBy=Autor`

f. Vuelva a **Postman** y vaya a la primera pestaña API, GET http://library.demo.local/api/v1/books. Ahora incluirá algunos de los parámetros del sitio web de Librería Escolar de API.

g. Haga clic en **Params**. Verá en los cuadros de entrada de **parámetros de consulta** para **KEY** y **VALUE**. Ingrese la siguiente información:

- En **KEY**, escriba **IncludeISBN** y en **Valor** ingrese **Cierto (true)**

  Observe que una marca de verificación se incluirá automáticamente a la izquierda del valor y se agregará una nueva fila.

- En **KEY**, escriba **SortBy** y en **Valor** escriba **autor**

Observe que al introducir estos parámetros de consulta, ha actualizado la URL original junto a GET. Esta es la misma **URL de solicitud** que vio en el sitio web de School Library API para esta misma llamada de API. Esta es la URL que usará Postman, con estos parámetros de consulta al realizar la llamada API.

`http://library.demo.local/api/v1/books?includeIsBN=true&sortBy=Autor`

h. Haga clic en **Send** (Enviar).

Nota en el **cuerpo**, ahora muestra la misma lista de libros, ordenados por autor e incluyendo los ISBN que vió en el sitio web de la API de la biblioteca escolar.

```
[
    {
        "ID": 5.
        "título": "31 días antes de su examen CCNA",
        "author": "Allan Johnson"
    },
    {
        "id"2
        "título": "Linux para Trabajadore de Net"
        "author": "Cisco Systems Inc.",
        "isbn": "000-0000000123"
    },
    {
        "id" 3,
        "título" "NetAcad: 20 años de aprendizaje en línea"
        "author": "Cisco Systems Inc.",
        "isbn": "000-0000001123"
    },
    {
        "ID": 0,
        "Título": "Fundamentos del enrutamiento IP",
        "author": "Mark A. Sportack",
        "isbn": "978-1578700714"
    },
    {
        "id": 4,
        "título": "Fundamentos de IPv6",
        "author": "Rick Graziani",
        "isbn": "978 1587144778"
    },
    {
        "ID":1
        "Título" "Python para tontos"
        "author": "Stef Maruch Aahz Maruch",
        "isbn": "978-0471778646"
    }
```

```
]
```

Parte 4: Usar Python para agregar 100 libros al simulador de API

Usted puede usar la herramienta Prueba de especificación OpenAPI o Postman para agregar tantos libros como desee. Sin embargo, usted tiene que agregarlos uno a la vez. Una mejor solución sería escribir un programa para agregar los libros. En esta parte, simulará el proceso de agregar 100 libros utilizando la biblioteca de **falsificadores** de Python.

Paso 1: Abra el Estudio Visual (Visual Studio VS Code) y navegue hasta el directorio de la biblioteca escolar.

a. Abra **VS Code** desde el botón **Menú** o haciendo doble clic en el icono del escritorio.

b. Haga clic en **Archivo > Abrir carpeta...**, vaya a la carpeta **labs/devnet-src/school-library** y haga clic en **Aceptar**.

Paso 2: Investigue las bibliotecas utilizadas por el programa add100RandomBooks.py.

a. En el panel VS Code **EXPLORER** de la izquierda, haga clic en **add100RandomBooks.py** para abrirlo, si es necesario.

b. En la parte superior, observe el «shebang» que establece al intérprete en Python 3 y luego las tres bibliotecas que se importan.

```
#!/usr/bin/env python3


solicitudes de importación
Importar JSON
de la importación de falsificador (faker) a Falsificador (Faker)
```

c. Usará la biblioteca de **solicitudes** de Python a lo largo de este curso. La biblioteca de **peticiones** es necesaria si desea utilizar Python para realizar solicitudes API utilizando GET, POST, DELETE y otros métodos HTTP.

d. Falsificador (Faker) es una biblioteca de Python que genera datos 'falsos' para usted. Este programa utiliza la biblioteca de **falsificaciones** de Python para generar títulos de libros aleatorios, authores e ISBN. Puede buscar en Internet más información sobre la biblioteca de **Falsificador (faker)**. Sin embargo, complete los pasos siguientes para ver todos los 252 métodos de la biblioteca de falsificadores.

1) Abra una ventana terminal e inicie Python 3

2) Desde **Falsificador (faker)** importar el módulo **Falsificador () (Faker ())**.

3) Asigne el módulo **Falsificador () (Faker ())** a **falso**.

4) Para ver todos los métodos, ingrese **falso.** y, a continuación, presione la tecla tabulación dos veces. Observe el método **Nombre.Falso() (fake.name ())**, que utilizará en el siguiente paso. En el siguiente paso y más adelante en este laboratorio, también usará los tres métodos resaltados (premarcados con **falso.**).: **catch_phrase ()**, **isbn13 ()**y **name ()**.

```
devasc @labvm: ~/labs/devnet-src/school-library$ python3
Python 3.8.2 (predeterminado, 27 de abril de 2020, 15:53:34)
[GCC 9.3.0] en Linux
Escriba "Ayuda" (Help), "Derechos de autor" (Copyright) o "Licencia"
(License) para más información
>>> de la importación de falsificador Faker
>>> falso = Faker ()
```

```
>>> falso.<press Tab key twice – be sure you include the period>
Mostrar todas las 252 posibilidades? (s o n) (y o n)
fake.add_provider (fake.future_datetime (fake.pyfloat (
fake.address (fake.generator_attrs fake.pyint (
fake.am_pm (fake.get_formatter (fake.pyiterable (
<output omitted>
fake.catch_phrase ( fake.ipv4_public (fake.random_element (
fake.chrome (fake.isbn10 (fake.random_int (
fake.city ( fake.isbn13 ( fake.random_letter (
<output omitted>
fake.date_time_ad (fake.msisdn (fake.texts (
fake.date_time_between ( fake.name ( fake.time (
<output omitted>
fake.future_date (fake.pydict (fake.zipcode_plus4 (
>>>
```

Paso 3: Practique generar datos aleatorios usando la biblioteca de falsificadores.

a. Introduzca lo siguiente para generar un nombre falso. Su salida será un nombre falso diferente cada vez que ejecute el comando.

```
>>> print ('Mi nombre es {} .'.format (fake.name ()))
Mi nombre es Katherine Ross.
>>>
```

b. Utilizando los tres mensajes resaltados en el paso 2d anterior, introduzca el comando que imprimiría la siguiente salida falsa.

```
>>> print ('Mi nombre es {} y escribí «{}» (ISBN {}) .'.format (fake.name (),
fake.catch_phrase (), fake.isbn13 ()))
Mi nombre es Gary Castaneda y escribí «Organic incremental neural-net» (ISBN 978-0-
669-01935-3).
>>>
```

c. Más adelante en el programa, se utiliza un bucle para iterar a través de estos tres métodos para crear entradas para la Biblioteca de la Escuela. Introduzca lo siguiente para generar 10 nombres aleatorios. Después del "..." deberá presionar Volver (Return) por segunda vez.

```
>>> para i en el rango (10):
... print (fake.name ())
...
Kevin Moyer
Sr. Christopher Green MD
Spencer Jensen
Whitney Guzmán
Nicole Scott
Tammy Lewis
Craig Edwards
Michael Díaz
Ryan Mccoy
Terry Rocha
>>>
```

d. Salga del intérprete de Python cuando termine de investigar la biblioteca de **falsificadores**.

```
>>> saltar()
devasc @labvm: ~/labs/devnet-src/librería$
```

Paso 4: Revise las variables de función.

Las dos funciones del programa utilizan tres variables para obtener un token de autorización del servicio API de la biblioteca escolar.

```
APIHOST = "http://library.demo.local»
INICIAR SESIÓN = "cisco"
CONTRASEÑA = «Cisco123!«
```

Paso 5: Revise la función Obtener FichaAutenticadora (GetAuthToken).

a. La función **getAuthToken** utiliza tres variables para generar una solicitud. La variable **r** invoca el método POST para la API **LoginViaBasic** y almacena el valor del token si la llamada es exitosa. Observe el uso de una cadena f para crear la URL de solicitud.

```
def getAuthToken ():
    AuthCreds = (LOGIN, PASSWORD)
    r = requests.post (
        f"{APIHOST}/api/v1/loginViaBasic",
        auth = AuthCreds
    ).
```

b. Si la llamada no se realiza correctamente (el código de estado HTTP no es igual a 200), se genera una excepción e imprime en la ventana de terminal. Una vez más, observe el uso de una cadena f para generar el mensaje de excepción. Puede probar el código de excepción cambiando una de las variables en el paso 4.

```
    si r.status_code == 200:
        return r.json () ["token"]
    else:
        Elevar Excepción (F"Código de estado {r.status_code} y texto {r.text},
mientras intenta Auth.")
```

Paso 6: Revise la función AñadirLibro (AddBook).

a. Al igual que la función **AddAuthToken**, la función **AddBook** utiliza las tres variables que se muestran en el paso 4 para generar una solicitud. La variable **r** invoca el método POST para la API de **libros**. Los datos provienen de una variable llamada **libro**, que se especifica en la parte final del programa.

```
def addBook (libro, ApiKey):
    r = requests.post (
        f"{APIHOST}/api/v1/books",
        headers = (
            "Content-type". "application/JSON",
            "X-API-key": apiKey
            },
        data = json.dumps (book)
    ).
```

b. Si la llamada no tiene éxito, se genera una excepción e imprime en la ventana de terminal. Puede probarlo cambiando una de las variables en el paso 4.

```
    si r.status_code == 200:
        print (f"book {book} añadido.")
```

```
    else:
        Elevar Excepción (f"Código de error {r.status_code} y texto {r.text}, mientras
intenta agregar libro {book}.«")
```

Paso 7: Revise el código que invoca las dos funciones.

a. Se invoca la función **AddAuthToken** y los resultados se almacenan en la variable ApiKey.

```
apiKey = getAuthToken ()
```

b. El módulo **Faker ()** se establece en una variable llamada **fake**. A **un** bucle luego itera 100 veces. La variable **i** se utiliza más adelante en el bucle para establecer el valor de la clave **id** para cada nuevo libro desde 4 hasta y sin incluir 104.

Nota: Si usted desea mantener los dos libros anteriores agregados anteriormente en este laboratorio, cambie el **rango** a **(6, 106)**.

```
falso = Faker ()
para i en el intervalo (4, 104):
```

c. A continuación, tres variables contienen el valor de los métodos invocados desde el módulo **Faker ()** : **catch_phrase ()**, **name ()**e **isbn13 ()**.

```
    FakeTitle = fake.catch_phrase ()
    FakeAuthor = fake.name ()
    FakeISBN = fake.isbn13 ()
```

d. Recuerde que el parámetro de **carga útil** para la API de **libros** requiere JSON en el siguiente formato:

```
{
  "ID": 0,
  "título": "string",
  "author": "cadena"
}
```

La variable **libro** se construye utilizando las tres claves necesarias para el parámetro de carga útil y los valores de las tres variables falsas.

```
    book = {"id» :i, "title": fakeTitle, "author": fakeAuthor, "isbn": fakeISBN}
```

e. Finalmente, la función **AddBook** se llama pasando el **libro** y las variables **APIKey**. Debido a que **AddBook** es parte del bucle, se llamará 101 veces, una vez cada uno para el libro ID 4 a 105.

Paso 8: Ejecute y verifique el programa add100RandomBooks.py.

a. Introduzca el comando Python 3 para ejecutar el programa **add100RandomBooks.py**. Debe obtener una salida similar a la siguiente, aunque los títulos de su libro y los números ISBN serán diferentes valores falsos.

```
devasc @labvm: ~/labs/devnet-src/school-library$ python3 add100RandomBooks.py
Libro {'id': 4, 'title': 'Bastidor cliente-servidor asimilado', 'autor': 'Chelsea
Mitchell', 'isbn': '978-0-411-83123-3'} added.
Libro {'id': 5, 'title': 'Conglomeración tangible adaptativa', 'autor': 'Edward Ryan',
'isbn': '978-1-64406-014-8'} added.
<output omitted>
Libro {'id': 103, 'title': 'Data-warehouse' uniforme fundamental, 'autor': 'Dennis
David', 'isbn': '978-1-68465-896-1'} added.
Libro {'id': 104, 'title': 'Funcionalidades orgánicas de 4ª generación', 'autor':
'Nicole Gilbert', 'isbn': '978-0-13-176202-2'} added.
devasc @labvm: ~/labs/devnet-src/librería$
```

b. Vuelva al navegador Chromium y actualice la página web **http://library.demo.local/**. Ahora deberías ver tus 100 libros nuevos añadidos.

Nota: Si llegas a la página de documentación de la API en lugar de a la página principal (**http://library.demo.local/api/v1/docs**) y usa **Try It out, solo obtendrá una lista de los primeros 10 libros. Puede introducir un valor de 2 a 10 el** parámetro de página para ver los otros libros.

¿Cómo añadirías otros 100 libros?

Cambie el bucle for para tener un rango de 104 a 204.