

## Lab - Explore Rest APIs with API Simulator and Postman

### Tujuan

**Part 1: Menjalankan DEVASC VM**

**Part 2: Mengeksplor Dokumen API menggunakan Simulator API**

**Part 3: Menggunakan Postman untuk membuat API Calls ke Simulator API**

**Part 4: Menggunakan python untuk menambahkan data 100 Buku ke Simulator API.**

### Latar Belakang

DEVASC VM didalamnya sudah terdapat sebuah simulator dokumen API perpustakaan sekolah dan sudah terhubung dengan database. Nantinya kita dapat menggunakan simulator offline untuk mengeksplor dan melakukan test fungsionalitasnya.

Dalam kegiatan lab ini, kita akan mempelajari bagaimana untuk menggunakan Simulator API perpustakaan sekolah untuk membuat API call. API Call ini dapat digunakan untuk menampilkan daftar, menambahkan, dan menghapus buku. Nantinya kita akan menggunakan Postman untuk melakukan semua itu fungsionalitas tersebut.

### Resource yang dibutuhkan

- 1 PC / Laptop
- Virtual Box / VMWare
- DEVASC Virtual Machine

### Langkah-langkah

#### Part 1: Menjalankan DEVASC VM

Jalankan DEVASC yang ada di PC/Laptop anda.

#### Part 2: Mengeksplor Dokumentasi API Menggunakan Simulator API

Untuk memahami cara melakukan panggilan ke REST API, pengembang biasanya memulai dengan mempelajari dokumentasi API. Format untuk request, responses, headers, dan parameter untuk REST API biasanya didokumentasikan menggunakan Spesifikasi OpenAPI (sebelumnya Spesifikasi Swagger).

##### Step 1: Buka Web Browser Chromium

Double-click icon web browser Chromium yang ada di desktop.

##### Step 2: Connect ke web site Perpustakaan Sekolah (the School Library)

Jika browser tidak secara otomatis membuka situs web Perpustakaan Sekolah, ketikkan alamat **library.demo.local** kemudian enter agar menuju ke situs wes tersebut.

### Step 3: Menuju ke Halaman API docs

- a. Situs web defaultnya ke tab Buku Kami (**Our Books**) dan menampilkan daftar buku. Di sudut kanan atas yang menyatakan, **Click disini untuk API docs**, klik di sini untuk membuka halaman web dokumentasi API.

Anda sekarang akan melihat daftar API di **/api/v1 Default namespace**.

- b. Perhatikan panah bawah ke paling kanan. Mengklik di mana saja pada **/api/v1** akan meminimalkan daftar API dan mengarahkan panah ke kanan. Klik lagi pada “bar” yang sama untuk menampilkan kembali daftar API.

Perhatikan kunci di ujung kanan beberapa API. Kunci menunjukkan bahwa API ini memerlukan token untuk digunakan.

### Step 4: List Buku menggunakan GET /books API.

Klik di mana saja pada “bar” untuk **GET /books API**. API ini mengembalikan daftar buku di perpustakaan sekolah.

- **Parameters** - Ada beberapa parameter API opsional. Ini dapat digunakan untuk memfilter, mengurutkan, atau membuat paginasi output. Ini akan dirujuk nanti di lab ini.
- **Response content type** - Klik application/json untuk melihat daftar berbagai tipe format data yang informasinya dapat dilihat. Biarkan seleksi sebagai application/json.
- **Code** - Kode menampilkan 200 secara default, yang menunjukkan permintaan API dari server berhasil seperti yang ditampilkan di Deskripsi. (Anda belum mengirim permintaan API.)

### Step 5: Gunakan fitur Try it out di dokumentasi API.

Salah satu fitur yang lebih canggih dari Spesifikasi OpenAPI adalah kemampuan untuk menguji panggilan API untuk melihat apakah Anda membuatnya dengan benar. Anda juga dapat meninjau tanggapan untuk melihat apakah itu yang Anda harapkan. Anda akan melihat fitur pengujian yang sama ini dalam dokumentasi API untuk Cisco, MapQuest, dan organisasi lain yang menggunakan fitur Spesifikasi OpenAPI ini.

- a. Dalam dokumentasi **GET /books API**, klik tombol **Try it out**
- b. Perhatikan bahwa Anda sekarang memiliki opsi untuk memasukkan informasi untuk parameter opsional. Biarkan parameter kosong dan klik tombol **Execute**.

Di bagian **Responses** anda akan melihat:

- **Curl:** Perintah curl yang dapat anda gunakan untuk mengakses informasi yang sama untuk **/books API**.
- **Request URL:** URL ini digunakan dalam API request, yang dapat digunakan untuk meminta informasi yang sama menggunakan curl, Postman dan Python.
- **Code:** ini adalah Response code HTTP. Kode 200 menunjukkan panggilan yang berhasil.
- **Response body:** Daftar buku dalam format JSON.
- **Response headers:** Informasi tentang API yang dikembalikan dari server.

Di dalam **Response body** Anda akan melihat daftar buku dalam format JSON:

```
[
  {
    "id": 0,
    "title": "IP Routing Fundamentals",
    "author": "Mark A. Sportack"
  },
  {
```

```
[
  {
    "id": 1,
    "title": "Python for Dummies",
    "author": "Stef Maruch Aahz Maruch"
  },
  {
    "id": 2,
    "title": "Linux for Networkers",
    "author": "Cisco Systems Inc."
  },
  {
    "id": 3,
    "title": "NetAcad: 20 Years Of Online-Learning",
    "author": "Cisco Systems Inc."
  }
]
```

### Step 6: Gunakan perintah curl di jendela terminal.

**GET /books** API menyediakan informasi untuk mengakses konten yang ditampilkan di badan respons menggunakan **curl**. **curl** adalah alat command line untuk mentransfer data ke atau dari server, menggunakan salah satu protokol yang didukung termasuk HTTP dan HTTPS.

- Pilih perintah **curl**, klik kanan dan Salin ke clipboard Anda:

```
curl -X GET "http://library.demo.local/api/v1/books" -H "accept: application/json"
```

Buka jendela terminal. Klik kanan dan **Paste** konten dari clipboard ke terminal dan tekan Enter. Perhatikan bahwa ini memberikan informasi yang sama dengan antarmuka OpenAPI library.

```
devasc@labvm:~$ curl -X GET "http://library.demo.local/api/v1/books" -H
"accept: application/json"
[
  {
    "id": 0,
    "title": "IP Routing Fundamentals",
    "author": "Mark A. Sportack"
  },
  {
    "id": 1,
    "title": "Python for Dummies",
    "author": "Stef Maruch Aahz Maruch"
  },
  {
    "id": 2,
    "title": "Linux for Networkers",
    "author": "Cisco Systems Inc."
  },
  {
    "id": 3,
    "title": "NetAcad: 20 Years Of Online-Learning",
    "author": "Cisco Systems Inc."
  }
]
```

```
}  
]  
devasc@labvm:~$
```

### Step 7: Buat daftar buku dengan ISBN-nya menggunakan GET /books API.

- Kembali ke GET /books API situs web School Library API.
- Di bagian **Parameter**, pilih panah bawah di sebelah parameter **includeISBN** dan pilih **true**
- Klik **Execute**.

Perhatikan perubahan berikut dalam **Responses**:

- Curl** sekarang menyertakan parameter untuk ISBN.  

```
curl -X GET "http://library.demo.local/api/v1/books?includeISBN=true" -H  
"accept: application/json"
```
- Request URL** sekarang menyertakan parameter for ISBN.  

```
http://library.demo.local/api/v1/books?includeISBN=true
```
- Response body** memiliki daftar buku yang sama seperti yang ditunjukkan sebelumnya tetapi sekarang termasuk ISBN buku.

Untuk meminimalkan scrolling, ketika Anda selesai dengan API, Anda dapat menutup jendela API tertentu dengan mengklik di mana saja pada bar judul. Sekarang Anda dapat melihat semua API dengan lebih mudah.

### Step 8: Dapatkan token menggunakan POST /loginViaBasic API.

- Klik **POST API /loginViaBasic**.
- Perhatikan tidak ada parameter. Klik **Try it out**, lalu klik **Execute**.
- Bagian **Sign in** akan meminta Anda memasukkan **username** dan **password**. Masukkan informasi berikut dan klik **Sign in**:
  - Username**: cisco
  - Password**: Cisco123!
- Token akan ditampilkan di **Response body**. Pilih informasi di antara tanda kutip, klik kanan dan **Copy** informasi ke clipboard Anda. **Token Anda akan berbeda dari yang ditunjukkan di bawah ini**.

```
{  
  "token": "cisco|KZZzteQbC5iV3HKEzB7hCJ6qHQXen4rLGh72YJKeVfs"  
}
```

- Scroll ke atas halaman API Perpustakaan Sekolah dan klik tombol **Authorize** berwarna hijau. Kotak dialog **Available authorizations** akan muncul.
- Klik kanan dan **Paste** token setelah **Value** dan klik **Authorize**. Perhatikan **Name** adalah X-API-KEY. Informasi ini bersama dengan token akan digunakan nanti di Postman.
- Tutup kotak dialog **Available authorizations** dan kembali ke daftar API. Perhatikan kunci oleh beberapa API sekarang telah berubah. API ini sekarang tersedia untuk Anda gunakan.
- Klik bar untuk **POST API /loginViaBasic** untuk menutup jendela.

### Step 9: Tambahkan buku menggunakan POST /books API.

- Click the API **POST /books**.

- b. Perhatikan di bawah Parameter bahwa **payload** diperlukan. Ini berarti bahwa API ini memerlukan informasi untuk parameter ini dalam format yang ditentukan oleh **Parameter content type**, yaitu JSON.
- c. Click **Try it out**.
- d. Ubah **id**, **title** dan **author** dengan informasi yang ada di bawah ini.

```
{
  "id": 4,
  "title": "IPv6 Fundamentals",
  "author": "Rick Graziani"
}
```

- e. Click **Execute**.
- f. Verifikasi bahwa post berhasil dalam respons Server. Kode 200 berarti postingan berhasil. Anda akan melihat buku yang Anda tambahkan di **Response body** bersama dengan id baru. Anda juga akan melihat informasi terbaru untuk **curl** dan **Request URL**.
- g. Untuk menambahkan buku lain, ubah id, judul, dan penulis dengan informasi yang ditunjukkan di bawah ini.

```
{
  "id": 5,
  "title": "31 Days Before Your CCNA Exam",
  "author": "Allan Johnson"
}
```

- h. Click **Execute**.
- i. Verifikasi bahwa pos berhasil dalam **Server response**. Kode 200 berarti postingan berhasil. Anda akan melihat buku yang Anda tambahkan di **Response body** bersama dengan id baru. Anda juga akan melihat informasi terbaru untuk **curl** dan **Request URL**.

Catatan: Jika Anda mendapatkan **kode 401**, periksa teks **Response body**. Kemungkinan besar Anda menerima respon "**error**": "**Invalid API key**". Ini karena Anda tidak memasukkan semua karakter untuk **API Key** Anda. Atau mungkin, Anda menambahkan ruang yang tidak perlu. Kembali ke langkah sebelumnya dan ulangi proses otorisasi.

- j. Click pada bar API **POST /books** untuk menutup jendela/window.
- k. Anda dapat memverifikasi bahwa buku telah ditambahkan ke halaman **Our Books**. Kembali ke tab Perpustakaan Sekolah di browser Anda (<http://library.demo.local>) dan refresh halaman. Berhati-hatilah untuk tidak menutup tab API Perpustakaan Sekolah. Jika ya, maka Anda perlu mengautentikasi ulang.

### Step 10: List books using the GET /books API.

- a. Kembali ke tab browser **School Library API**. Klik **GET /books** API
- b. Klik **Try it out**. Jika Anda melihat tombol **Cancel** berwarna merah, maka Anda sudah dalam mode **Try it out**.
- c. Click **Execute**.
- d. Di bawah **Server response** di **Response body**, Anda sekarang akan melihat dua buku yang Anda tambahkan. Perhatikan mereka masing-masing memiliki **id** unik.

```
[
  {
    "id": 0,
    "title": "IP Routing Fundamentals",
    "author": "Mark A. Sportack"
  }
]
```

```
},
{
  "id": 1,
  "title": "Python for Dummies",
  "author": "Stef Maruch Aahz Maruch"
},
{
  "id": 2,
  "title": "Linux for Networkers",
  "author": "Cisco Systems Inc."
},
{
  "id": 3,
  "title": "NetAcad: 20 Years Of Online-Learning",
  "author": "Cisco Systems Inc."
},
{
  "id": 4,
  "title": "IPv6 Fundamentals",
  "author": "Rick Graziani"
},
{
  "id": 5,
  "title": "31 Days Before Your CCNA Exam",
  "author": "Allan Johnson"
}
]
```

- e. Klik bar **GET /books** API untuk menutup window.

### Step 11: Buat daftar buku tertentu menggunakan GET /books{id} API.

- Klik **GET /books{id}** API. Perhatikan bahwa API ini memerlukan **id** sebagai parameter.
- Di sebelah kanan **Parameter**, klik tombol **Try it out**.
- Di bawah **Parameter**, masukkan 4 untuk id yang diperlukan.
- Klik **Execute**. Perhatikan informasi yang diberikan oleh **Curl** dan **Request URL**.
  - Curl** - Ini adalah perintah curl untuk melakukan fungsi yang sama menggunakan curl.
  - Request URL** - Ini adalah URL yang dapat digunakan untuk mendapatkan informasi yang sama menggunakan Postman dan Python.

Verifikasi bahwa **get** berhasil dalam **Server response**. Kode 200 berarti postingan berhasil. Di **Response body** Anda akan melihat buku yang Anda minta dengan **id 4**.

```
{
  "id": 4,
  "title": "IPv6 Fundamentals",
  "author": "Rick Graziani"
}
```

- e. Click bar **GET /books{id}** API untuk menutup window.

**Step 12: Hapus buku tertentu menggunakan DELETE /books{id} API.**

- Klik **DELETE /books{id}** API. Perhatikan bahwa API ini memerlukan **id** sebagai parameter.
- Click **Try it out**.
- Di bawah Parameter, masukkan 4.
- Click **Execute**.
- Verifikasi bahwa penghapusan berhasil di **Server response**. Kode 200 berarti postingan berhasil. Di **Response body** Anda akan melihat buku yang Anda hapus dengan **id 4**.

```
{
  "id": 4,
  "title": "IPv6 Fundamentals",
  "author": "Rick Graziani"
}
```

- Click bar **DELETE /books{id}** API untuk menutup window.

**Step 13: Buat daftar buku menggunakan GET /books API.**

- Click **GET /books** API.
- Click **Try it out**. Jika anda melihat tombol **Cancel** berwarna merah, maka anda sudah berada di mode **Try it out**.
- Click **Execute**.
- Di bawah **Server response** di **Response body**, Anda tidak akan lagi melihat buku dengan **id** atau 4.

```
[
  {
    "id": 0,
    "title": "IP Routing Fundamentals",
    "author": "Mark A. Sportack"
  },
  {
    "id": 1,
    "title": "Python for Dummies",
    "author": "Stef Maruch Aahz Maruch"
  },
  {
    "id": 2,
    "title": "Linux for Networkers",
    "author": "Cisco Systems Inc."
  },
  {
    "id": 3,
    "title": "NetAcad: 20 Years Of Online-Learning",
    "author": "Cisco Systems Inc."
  },
  {
    "id": 5,
    "title": "31 Days Before Your CCNA Exam",
    "author": "Allan Johnson"
  }
]
```

]

**Catatan:** Jangan tutup tab **School Library API** di browser Chromium. Anda akan menggunakan dokumentasi API di bagian selanjutnya.

### Part 3: Gunakan Postman untuk Membuat Panggilan API ke Simulator API

Di Bagian ini, Anda akan menggunakan Postman untuk membuat panggilan API yang sama dengan yang Anda buat di dokumentasi API Perpustakaan Siswa. Postman adalah alat/tools yang berguna ketika situs web pengembang API tidak tersedia sambil memberikan kemampuan untuk menyimpan, mengatur, dan menggunakan kembali API dengan mudah.

#### Step 1: Buka Postman.

Klik dua kali ikon Postman di desktop. Biasanya, Anda akan masuk ke Postman. Namun, tidak perlu mendapatkan akun dan login ke Postman untuk lab dalam kursus ini.

#### Step 2: Buat daftar buku menggunakan GET /books API.

- Di jendela utama di sebelah tab **Launchpad**, klik ikon plus "+" untuk membuat Permintaan Tanpa Judul / **Untitled Request** Secara default, ini akan menjadi permintaan **GET**.
- Klik panah bawah di sebelah **GET** untuk melihat operasi API yang berbeda termasuk **GET, POST, dan DELETE**. Biarkan pilihan pada **GET**. Klik panah atas di sebelah GET untuk menutup daftar.
- Masukkan request URL.
  - Kembali ke tab **School Library API** di Chromium dan, jika perlu, perluas **GET /books** API.
  - Di bawah **Request URL**, pilih, klik kanan dan **Copy** URL ke clipboard Anda:  
`http://library.demo.local/api/v1/books`
  - Kembali ke **Postman** dan rekatkan URL di sebelah GET yang menyatakan, " Enter request URL ".  
**Catatan:** Jika menempel menambahkan baris di bawah URL, hapus baris tambahan.
- Klik **Send**. Untuk memverifikasi bahwa permintaan API berhasil, Anda sekarang akan melihat respons yang menyertakan kode **Status** 200 OK berwarna hijau. Scroll ke bawah ke bagian **Body** untuk melihat responsnya. Perhatikan bahwa defaultnya adalah **Pretty** dan **json**.

```
[
  {
    "id": 0,
    "title": "IP Routing Fundamentals",
    "author": "Mark A. Sportack"
  },
  {
    "id": 1,
    "title": "Python for Dummies",
    "author": "Stef Maruch Aahz Maruch"
  },
  {
    "id": 2,
    "title": "Linux for Networkers",
    "author": "Cisco Systems Inc."
  },
  {
    "id": 3,
```



```
    "title": "NetAcad: 20 Years Of Online-Learning",
    "author": "Cisco Systems Inc."
  },
  {
    "id": 5,
    "title": "31 Days Before Your CCNA Exam",
    "author": "Allan Johnson"
  }
]
```

**Catatan:** Anda dapat menyimpan output JSON ke file menggunakan tombol **Save Response** di atas output. Ini tidak diperlukan untuk lab ini.

### Step 3: Dapatkan Token menggunakan POST /loginViaBasic API.

- Di jendela utama, klik ikon **plus "+"** untuk membuat baru **Untitled Request**.
- Klik panah bawah di sebelah **GET** dan pilih **POST**.
- Masukkan request URL.
  - Kembali ke tab **School Library API** di Chromium dan perluas **POST /loginViaBasic** API, jika perlu.
  - Di bawah **Request URL**, pilih, klik kanan dan Copy URL ke clipboard Anda:

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

**Catatan:** Jika **Request URL** tidak lagi ditampilkan, Anda mungkin menutup dan membuka kembali halaman dokumentasi **School Library API** dan tidak lagi diautentikasi. Klik **Try it out**, lalu **Execute**, lalu autentikasi ulang dengan username **cisco** dan password **Cisco123!**.

- Kembali ke **Postman** dan Paste URL di sebelah POST yang menyatakan, " Enter request URL ".

**Note:** Jika akan melakukan Paste dengan menambahkan baris di bawah URL, hapus baris tambahan.
- Click **Authorization**. Dalam area ini, lengkapi hal berikut:
  - Dalam daftar drop-down untuk **Type**, pilih **Basic Auth**.
  - Untuk bagian **Username** and **Password**, isian dengan dibawah ini:
    - Username:** cisco
    - Password:** Cisco123!

- Click **Send**.
- Jika perlu, scroll ke bawah ke bagian **Body** untuk melihat token baru Anda. Token Anda akan berbeda dari yang ditampilkan di sini.

```
{
  "token": "cisco|5xSUHYFDvIAoCRv0LqWVSDcjJAwWjg18vMm16u2lm1I"
}
```

### Step 4: Tambahkan buku menggunakan POST /books API.

Sekarang Anda akan menambahkan buku *IPv6 Fundamentals* yang Anda hapus di Part 2 saat menggunakan fitur **Try it out** di dokumentasi **School Library API**.

- Di jendela utama, klik ikon plus "+" untuk membuat **Untitled Request**.
- Klik panah bawah di sebelah **GET** dan pilih **POST**.
- Masukkan request URL.

- 1) Kembali ke tab **School Library API** di Chromium and perluas **POST /books** API.
- 2) Dibawah **Request URL**, pilih, klik kanan dan **Copy** URL ke clipboard Anda::

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

**Catatan:** Jika **Request URL** tidak lagi ditampilkan, Anda mungkin membatalkan **Try it out**. Klik **Try it out**, lalu **Execute** untuk menampilkan **Request URL**.

- 3) Kembali ke **Postman** dan paste URL di sebelah POST yang menyatakan, " Enter request URL ".

**Note:** Jika akan melakukan Paste dengan menambahkan baris di bawah URL, hapus baris tambahan.

- d. Click **Authorization**. Dalam area ini, lengkapi hal berikut:

- 1) Dalam daftar drop-down untuk **Type**, pilih **API Key**.
- 2) Di dalam bagian **Key**, Masukkan **X-API-KEY**.

**Catatan:** Ingatlah bahwa Anda melihat **X-API-KEY** di halaman web School Library API saat Anda mendapatkan token dengan memilih tombol **Authorize** berwarna hijau.

- 3) Kembali ke tab **Post** di Postman dan copy token yang Anda terima di Langkah 3. Pastikan untuk memasukkan semua yang ada di dalam tanda kutip. Token Anda akan berbeda dari yang ditampilkan di sini.

Example: **cisco|5xSUHYFDvIAoCRv0LqWVSDcjJAwWjg18vMm16u21m1I**

- 4) Kembali ke tab **Post** kedua di Postman. Paste token di bagian **Value**.

- e. Di baris yang sama dengan tab **Authorization**, klik **Body**. Bagian ini akan memungkinkan Anda untuk memilih format input Anda.

- Click radio button **raw**.
- Click **Text** dan ubah option/pilihan ke **JSON**.

- f. Di area input Anda akan melihat angka 1, untuk " line 1". Masukkan objek JSON berikut.

```
{
  "id": 4,
  "title": "IPv6 Fundamentals",
  "author": "Rick Graziani",
  "isbn": "978 158144778"
}
```

- g. Click **Send**.

- h. Untuk memverifikasi bahwa API request berhasil, Anda sekarang akan melihat respons yang menyertakan kode **Status** 200 OK berwarna hijau.

### Step 5: Verifikasi buku tambahan dengan Get /books API.

- a. Kembali ke tab GET pertama. Seperti yang Anda lihat, Postman memudahkan untuk beralih di antara panggilan API yang berbeda.
- b. Click **Send**.
- c. Untuk memverifikasi bahwa API request berhasil, Anda sekarang akan melihat respons yang menyertakan kode **Status** 200 OK berwarna hijau.
- d. Click **Body** untuk melihat Response. Perhatikan bahwa defaultnya adalah **Pretty** dan **json**.

```
[
  {
```

```
    "id": 0,
    "title": "IP Routing Fundamentals",
    "author": "Mark A. Sportack"
  },
  {
    "id": 1,
    "title": "Python for Dummies",
    "author": "Stef Maruch Aahz Maruch"
  },
  {
    "id": 2,
    "title": "Linux for Networkers",
    "author": "Cisco Systems Inc."
  },
  {
    "id": 3,
    "title": "NetAcad: 20 Years Of Online-Learning",
    "author": "Cisco Systems Inc."
  },
  {
    "id": 4,
    "title": "IPv6 Fundamentals",
    "author": "Rick Graziani"
  },
  {
    "id": 5,
    "title": "31 Days Before Your CCNA Exam",
    "author": "Allan Johnson"
  },
]
```

### Step 6: Gunakan parameter tambahan dengan Get /books API.

- a. Buka situs web **School Library API**. Scroll ke atas ke **GET /books** API dan perluas, jika perlu. Perhatikan parameter yang tersedia:
  - **includeISBN**: Mencantumkan nomor ISBN dalam hasil. Default=false
  - **sortBy**: Mengurutkan hasil menggunakan parameter yang ditentukan. Default=id
  - **author**: Kembalikan hanya buku-buku dari Penulis yang bersangkutan..
  - **page**: Digunakan untuk menentukan nomor halaman.
- b. Click **Try it out**. Jika anda melihat tombol **Cancel** yang berwarna merah, maka Anda tidak perlu memilih tombol ini.
- c. Dibawah parameters:
  - Click **includeISBN** and pilih **true**
  - Click **sortBy** and pilih **author**
- d. Click **Execute**.

- e. Di **Response body** Anda akan melihat daftar buku yang sekarang diurutkan berdasarkan penulis dan termasuk ISBN.

```
[
  {
    "id": 5,
    "title": "31 Days Before Your CCNA Exam",
    "author": "Allan Johnson"
  },
  {
    "id": 2,
    "title": "Linux for Networkers",
    "author": "Cisco Systems Inc.",
    "isbn": "000-0000000123"
  },
  {
    "id": 3,
    "title": "NetAcad: 20 Years Of Online-Learning",
    "author": "Cisco Systems Inc.",
    "isbn": "000-0000001123"
  },
  {
    "id": 0,
    "title": "IP Routing Fundamentals",
    "author": "Mark A. Sportack",
    "isbn": "978-1578700714"
  },
  {
    "id": 4,
    "title": "IPv6 Fundamentals",
    "author": "Rick Graziani",
    "isbn": "978 1587144778"
  },
  {
    "id": 1,
    "title": "Python for Dummies",
    "author": "Stef Maruch Aahz Maruch",
    "isbn": "978-0471778646"
  }
]
```

Perhatikan bahwa **Request URL** sekarang menyertakan parameter. Anda akan melihat ini lagi di Postman.

<http://library.demo.local/api/v1/books?includeISBN=true&sortBy=author>

- f. Kembali ke **Postman** dan buka tab API pertama, DAPATKAN <http://library.demo.local/api/v1/books>. Anda sekarang akan menyertakan beberapa parameter dari situs web the School Library API.
- g. Click **Params**. Anda akan melihat di bawah kotak input **Query Params** untuk **KEY** dan **VALUE**. Masukkan informasi berikut:
- Dibawah **KEY**, masukkan **includeISBN** dan dibawah **Value** masukkan **true**

Perhatikan tanda centang akan secara otomatis disertakan di sebelah kiri value dan baris baru ditambahkan.

- Dibawah **KEY**, masukkan **sortBy** dan dibawah **Value** masukkan **author**

Perhatikan bahwa ketika memasukkan parameter queri ini, ia telah memperbarui URL asli di sebelah GET. Ini adalah **Request URL** yang sama yang Anda lihat di situs web School Library API untuk panggilan API yang sama ini. Ini adalah URL Postman yang akan digunakan, dengan parameter queri ini saat melakukan panggilan API.

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

- h. Click **Send**.

Perhatikan di **Body**, sekarang menunjukkan daftar buku yang sama, diurutkan berdasarkan penulis dan termasuk ISBN yang Anda lihat di situs web School Library API.

```
[
  {
    "id": 5,
    "title": "31 Days Before Your CCNA Exam",
    "author": "Allan Johnson"
  },
  {
    "id": 2,
    "title": "Linux for Networkers",
    "author": "Cisco Systems Inc.",
    "isbn": "000-0000000123"
  },
  {
    "id": 3,
    "title": "NetAcad: 20 Years Of Online-Learning",
    "author": "Cisco Systems Inc.",
    "isbn": "000-0000001123"
  },
  {
    "id": 0,
    "title": "IP Routing Fundamentals",
    "author": "Mark A. Sportack",
    "isbn": "978-1578700714"
  },
  {
    "id": 4,
    "title": "IPv6 Fundamentals",
    "author": "Rick Graziani",
    "isbn": "978 1587144778"
  },
  {
    "id": 1,
    "title": "Python for Dummies",
    "author": "Stef Maruch Aahz Maruch",
    "isbn": "978-0471778646"
  }
]
```

## Part 4: Gunakan Python untuk Menambahkan 100 Buku ke Simulator API.

Anda dapat menggunakan tool OpenAPI Specification Try atau Postman untuk menambahkan buku sebanyak yang Anda inginkan. Namun, Anda harus menambahkannya satu per satu. Solusi yang lebih baik adalah dengan menulis program untuk menambahkan buku. Di Bagian ini, Anda akan mensimulasikan proses penambahan 100 buku dengan menggunakan library **faker** di Python.

### Step 1: Buka Visual Studio Code (VS) dan arahkan ke direktori school-library.

- Buka **VS Code** dengan melakukan double klik icon yang ada di desktop.
- Click **File > Open Folder...**, arahkan ke direktori **labs/devnet-src/school-library**, dan klik **OK**.

### Step 2: Selidiki libraries yang digunakan oleh program add100RandomBooks.py.

- Di panel VS Code EXPLORER di sebelah kiri, klik add100RandomBooks.py untuk membukanya.
- Di bagian atas, perhatikan "shebang" yang mengatur penerjemah ke Python 3 dan kemudian tiga libraries yang diimpor

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

```
import requests
import json
from faker import Faker
```

- Anda akan menggunakan library **requests** Python selama kursus ini. Library Request Pustaka diperlukan jika Anda ingin menggunakan Python untuk membuat permintaan API menggunakan GET, POST, DELETE, dan metode HTTP lainnya.
- Faker adalah library Python yang menghasilkan data 'palsu/fake' untuk Anda. Program ini menggunakan library **faker** Python untuk menghasilkan judul buku, penulis, dan ISBN acak. Anda dapat mencari di internet untuk informasi lebih lanjut tentang library faker. Namun, selesaikan langkah-langkah berikut untuk melihat semua 252 metode untuk library faker.
  - Buka jendela terminal dan mulai Python 3.
  - Dari faker, impor modul Faker().
  - Assign modul **Faker()** ke **fake**
  - Untuk melihat semua metode, masukkan **fake**. lalu tekan tombol tab dua kali. Perhatikan metode **fake.name()**, yang akan Anda gunakan pada langkah berikutnya. Pada langkah berikutnya dan nanti di lab ini, Anda juga akan menggunakan tiga metode yang disorot (diawali dengan **fake.**): **catch\_phrase()**, **isbn13()**, dan **name()**.

```
devasc@labvm:~/labs/devnet-src/school-library$ python3
Python 3.8.2 (default, Apr 27 2020, 15:53:34)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from faker import Faker
>>> fake = Faker()
>>> fake.<press Tab key twice - be sure you include the period>
Display all 252 possibilities? (y or 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.chrome(
fake.city(
<output omitted>
fake.date_time_ad(
fake.date_time_between(
<output omitted>
fake.future_date(
fake.ipv4_public(
fake.isbn10(
fake.isbn13(
fake.random_element(
fake.random_int(
fake.random_letter(
fake.msisdn(
fake.name(
fake.texts(
fake.time(
fake.pydict(
fake.zipcode_plus4(
>>>
```

### Step 3: Berlatih menghasilkan data acak menggunakan library faker.

- a. Masukkan yang berikut ini untuk menghasilkan nama palsu. Output Anda akan menjadi nama palsu yang berbeda setiap kali Anda menjalankan perintah. Enter

```
>>> print('My name is {}'.format(fake.name()))
My name is Katherine Ross.
>>>
```

- b. Menggunakan tiga pesan yang disorot pada Langkah 2d di atas, masukkan perintah yang akan mencetak output palsu berikut.

```
>>> print('My name is {} and I wrote "{}" (ISBN
{}).'.format(fake.name(), fake.catch_phrase(), fake.isbn13()))
My name is Gary Castaneda and I wrote "Organic incremental neural-net" (ISBN 978-0-
669-01935-3).
>>>
```

- c. Kemudian dalam program, sebuah loop digunakan untuk mengulangi ketiga metode ini untuk membuat entri untuk School Library. Masukkan yang berikut untuk menghasilkan 10 nama acak. Setelah "..." Anda harus menekan kembali untuk kedua kalinya.

```
>>> for i in range(10):
...     print(fake.name())
...
Kevin Moyer
Mr. Christopher Green MD
Spencer Jensen
Whitney Guzman
Nicole Scott
Tammy Lewis
Craig Edwards
Michael Diaz
Ryan McCoy
Terry Rocha
>>>
```

- d. Keluar dari interpreter Python setelah selesai menyelidiki library **faker**.

```
>>> quit()
devasc@labvm:~/labs/devnet-src/school-library$
```

### Step 4: Review variabel fungsi/function.

Kedua function dalam program menggunakan tiga variabel untuk mendapatkan token otorisasi dari layanan School Library API.

```
APIHOST = "http://library.demo.local"
LOGIN = "cisco"
PASSWORD = "Cisco123!"
```

### Step 5: Review function getAuthToken.

- a. Fungsi **getAuthToken** menggunakan tiga variabel untuk membuat permintaan/request. Variabel **r** memanggil method POST untuk API **loginViaBasic** dan menyimpan nilai token jika panggilan berhasil. Perhatikan penggunaan f-string untuk membangun URL permintaan.

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

- b. Jika panggilan tidak berhasil (kode status HTTP tidak sama dengan 200), pengecualian muncul dan dicetak ke jendela terminal. Sekali lagi, perhatikan penggunaan f-string untuk membuat pesan pengecualian. Anda dapat menguji kode pengecualian dengan mengubah salah satu variabel di Langkah 4..

```
if r.status_code == 200:
    return r.json()["token"]
else:
    raise Exception(f"Status code {r.status_code} and text {r.text}, while trying to Auth.")
```

### Step 6: Review function addBook.

- a. Mirip dengan function **addAuthToken**, function **addBook** menggunakan tiga variabel yang ditunjukkan pada Langkah 4 untuk membuat permintaan/request. Variabel **r** memanggil metode POST untuk **books** API. Data berasal dari variabel yang disebut **book**, yang ditentukan di bagian akhir program.

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

- b. Jika panggilan tidak berhasil, pengecualian dimunculkan dan dicetak ke jendela terminal. Anda dapat mengujinya dengan mengubah salah satu variabel di Langkah 4.

```
if r.status_code == 200:
    print(f"Book {book} added.")
else:
    raise Exception(f"Error code {r.status_code} and text {r.text}, while trying to add book {book}.")
```



### Step 7: Review kode yang memanggil dua function.

- a. Fungsi **addAuthToken** dipanggil dan hasilnya disimpan dalam variabel **apiKey**

```
apiKey = getAuthToken()
```

- b. **Modul Faker()** diset ke variabel bernama **fake**. Ada loop/perulangan kemudian iterasi 100 kali. Variabel **i** digunakan kemudian dalam loop untuk mengatur nilai kunci **id** untuk setiap buku baru dari 4 hingga dan tidak termasuk 104.

**Catatan:** Jika Anda ingin menyimpan dua buku sebelumnya yang ditambahkan sebelumnya di lab ini, ubah rentang ke (6, 106).

Catatan untuk pengulas LA: Versi skrip Anda mungkin memiliki rentang dari 4 hingga 105, yang akan menambah 101 buku. Ubah rentang dan simpan skrip untuk hanya menambahkan 100 buku. Ini akan diperbarui dalam rilis GA dari DEVASC VM.

```
fake = Faker()
for i in range(4, 104):
```

- c. Selanjutnya, tiga variabel menyimpan nilai metode yang dipanggil dari modul **Faker()**: **catch\_phrase()**, **name()**, dan **isbn13()**.

```
fakeTitle = fake.catch_phrase()
fakeAuthor = fake.name()
fakeISBN = fake.isbn13()
```

- d. Ingatlah bahwa parameter **payload** untuk API **books** memerlukan JSON dalam format berikut:

```
{
  "id": 0,
  "title": "string",
  "author": "string"
}
```

Variabel **book** dibangun menggunakan tiga kunci yang diperlukan untuk parameter payload dan nilai dari tiga variabel fake.

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

- e. Terakhir, fungsi **addBook** dipanggil dengan meneruskan variabel **book** dan **apiKey**. Karena **addBook** adalah bagian dari loop, maka akan dipanggil 101 kali, masing-masing satu kali untuk ID buku 4 hingga 105.

### Step 8: Jalankan dan verifikasi program **add100RandomBooks.py**.

- a. Masukkan perintah Python 3 untuk menjalankan program **add100RandomBooks.py**. Anda akan mendapatkan output yang mirip dengan berikut ini meskipun judul buku dan nomor ISBN Anda akan berbeda nilai fakenya.

```
devasc@labvm:~/labs/devnet-src/school-library$ python3 add100RandomBooks.py
Book {'id': 4, 'title': 'Assimilated client-server frame', 'author': 'Chelsea Mitchell', 'isbn': '978-0-411-83123-3'} added.
Book {'id': 5, 'title': 'Adaptive tangible conglomeration', 'author': 'Edward Ryan', 'isbn': '978-1-64406-014-8'} added.
<output omitted>
Book {'id': 103, 'title': 'Fundamental uniform data-warehouse', 'author': 'Dennis David', 'isbn': '978-1-68465-896-1'} added.
Book {'id': 104, 'title': 'Organic 4thgeneration functionalities', 'author': 'Nicole Gilbert', 'isbn': '978-0-13-176202-2'} added.
devasc@labvm:~/labs/devnet-src/school-library$
```

- b. Kembali ke browser Chromium dan refresh halaman web <http://library.demo.local/>. Anda sekarang akan melihat 100 buku baru yang Anda ditambahkan.

**Catatan:** Jika Anda membuka halaman dokumentasi API sebagai ganti halaman utama (<http://library.demo.local/api/v1/docs>) dan menggunakan **Try It out**, Anda hanya akan mendapatkan daftar 10 buku pertama. Anda dapat memasukkan nilai dari 2 hingga 10 parameter page untuk melihat buku lainnya.

How would you add another 100 books?

**Change the for loop to have a range of 104 to 204.**