



*Cisco
Developers
Connect
SuperCAT*

May 23rd 2024, Seoul Korea

Coding & APIs Fundamentals

김기동 프로

Agenda

- REST API
- cURL 실습
- Postman 실습
- Python 실습



왜 API를 사용해야 되는지?

사용 중인 모든 스위치에서 down 상태인 인터페이스를 shutdown해서 administratively down 상태로 전환하려고 하는 경우



일련의 작업 절차를 코드로 작성하여 자동화하고 재사용 가능

```
for switch in my_network:
    for interface in switch:
        if interface.is_down():
            interface.shutdown()
            interface.set_description("Interface disabled per policy")
```

Interface

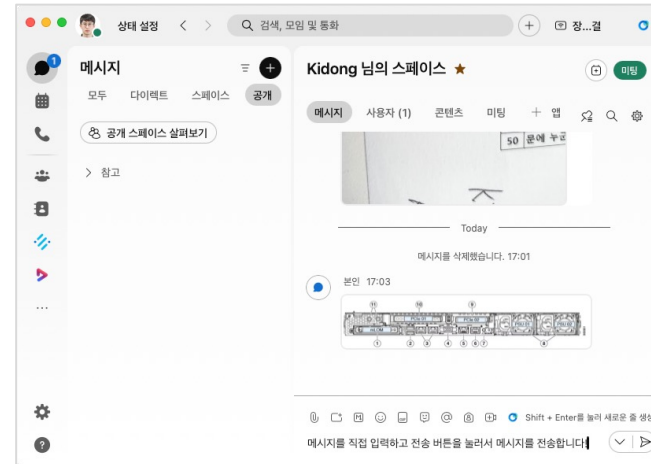
- 2개 이상의 사람, 장치, 소프트웨어가 서로 통신할 때 사용하는 것

- CLI (Command Line Interface)

```
ssh
TS#
TS#
TS#show run
Building configuration...

Current configuration : 6579 bytes
!
! Last configuration change at 02:07:10 UTC Mon Feb 5 2024 by admin
!
version 17.12
service timestamps debug datetime msec
service timestamps log datetime msec
platform qfp utilization monitor load 80
platform punt-keepalive disable-kernel-core
!
hostname TS
!
boot-start-marker
boot-end-marker
!
!
no aaa new-model
```

- GUI (Graphical User Interface)

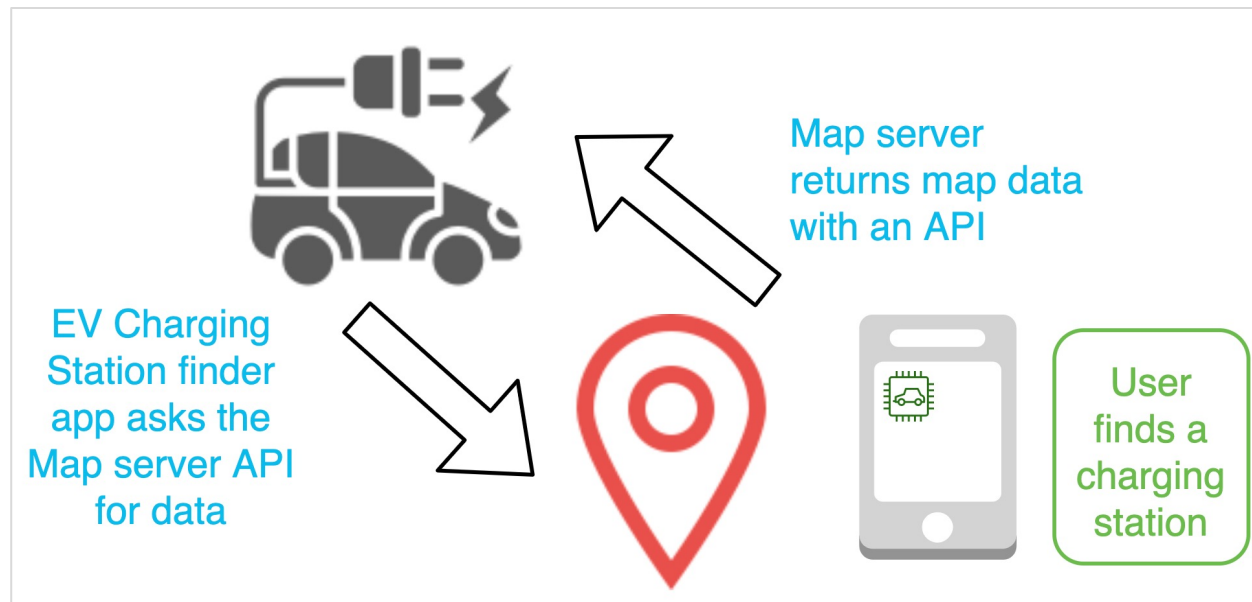


- SW와 SW 간에는?



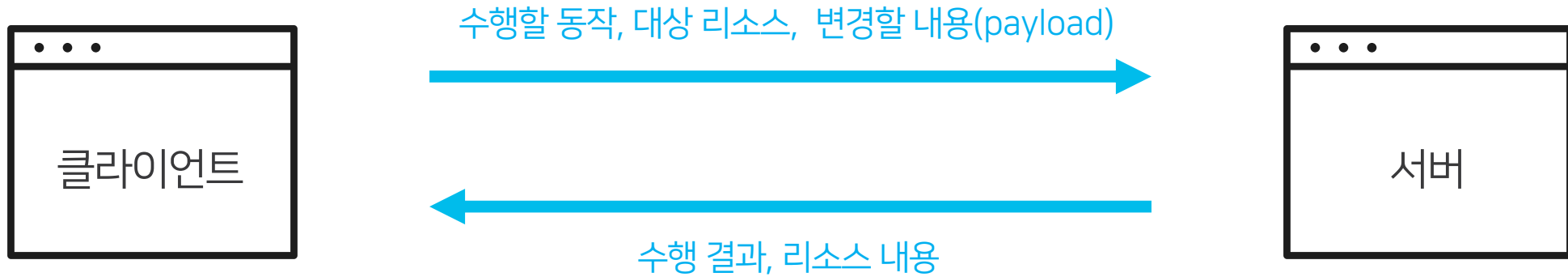
API (Application Programming Interface)

- 소프트웨어 간의 통신을 지원
- 모듈화: 명확하게 정의되고 재사용 가능한 모듈을 활용하여 애플리케이션을 구축
- 추상화: 세부 구현 사항을 추상화하여 API를 호출하는 상위 로직에 제공
- 안정성: 안정적이고 일관된 인터페이스를 제공



REST API

- REpresentational State Transfer API
- HTTP 기반으로 구축된 API 프레임워크



REST API: URI(Uniform Resource Identifier)

- 리소스 이름을 식별하는 데 사용되는 문자열



- Protocol: 사용할 프로토콜
- Server or Host: 웹 서비스를 제공하는 서버 또는 호스트의 도메인 이름
- Resource: 리소스 경로
- Query Parameter: 응답 데이터를 필터링하거나 범위를 지정하기 위한 세부 정보 (선택사항)

REST API: HTTP method

- 서버가 수행할 동작을 지정

HTTP method	CRUD	동작
POST	Create	새로운 리소스 생성
GET	Read	리소스 조회
PUT	Update	리소스 수정 (부분 수정 불가)
PATCH	Update	리소스의 일부 수정
DELETE	Delete	리소스 삭제

REST API: 데이터 포맷

- Header 에서 Content-Type 을 통해서 데이터 포맷 확인

데이터 포맷 확인 →

```
-zsh
kidkim@KIDKIM-M-G9YJ ~ % curl -i https://deckofcardsapi.com/api/deck/new/
HTTP/2 200
date: Tue, 21 May 2024 10:16:58 GMT
content-type: application/json
content-length: 80
access-control-allow-origin: *
x-content-type-options: nosniff
referrer-policy: same-origin
x-frame-options: DENY
vary: Origin
cf-cache-status: DYNAMIC
report-to: {"endpoints":[{"url":"https://a.nel.cloudflare.com/report/v4?s=WXcmY2%2Bil8FrWCakyqTfsbpuBuRWryaFoasDQaPFfNY%2FbCcLQvAVfHpxUPxcadNT3NRjQHhbEn0fbT58sbMyjo6geo6rdppk7mmJB70dDEXCHLgo8uGeZqSasmOUsWdvCrv8fDY%3D"}],"group":"cf-nel","max_age":604800}
nel: {"success_fraction":0,"report_to":"cf-nel","max_age":604800}
server: cloudflare
cf-ray: 8873d0632b5196b6-KIX
alt-svc: h3=":443"; ma=86400

{"success": true, "deck_id": "k6a6dqszi19y", "remaining": 52, "shuffled": false}

kidkim@KIDKIM-M-G9YJ ~ %
```

JSON 형식 →

REST API: 데이터 포맷

- JSON (JavaScript Object Notification)

- 속성-값 쌍(attribute-value pairs) 사용

```
{
  "zzzObject" : {
    "attributes" : {
      "property1" : "value1",
      "property2" : "value2",
      "property3" : "value3"
    },
    "children" :
    [{
      "zzzChild1" : {
        "attributes" : {
          "childProperty1" : "childValue1",
          "childProperty2" : "childValue1"
        },
        "children" : []
      }
    ]
  }
}
```

- XML (eXtensible Markup Language)

- 자체적으로 정의한 태그 사용

```
<zzzObject
  property1 = "value1",
  property2 = "value2",
  property3 = "value3">
  <zzzChild1
    childProperty1 = "childValue1",
    childProperty2 = "childValue1">
  </zzzChild1>
</zzzObject>
```

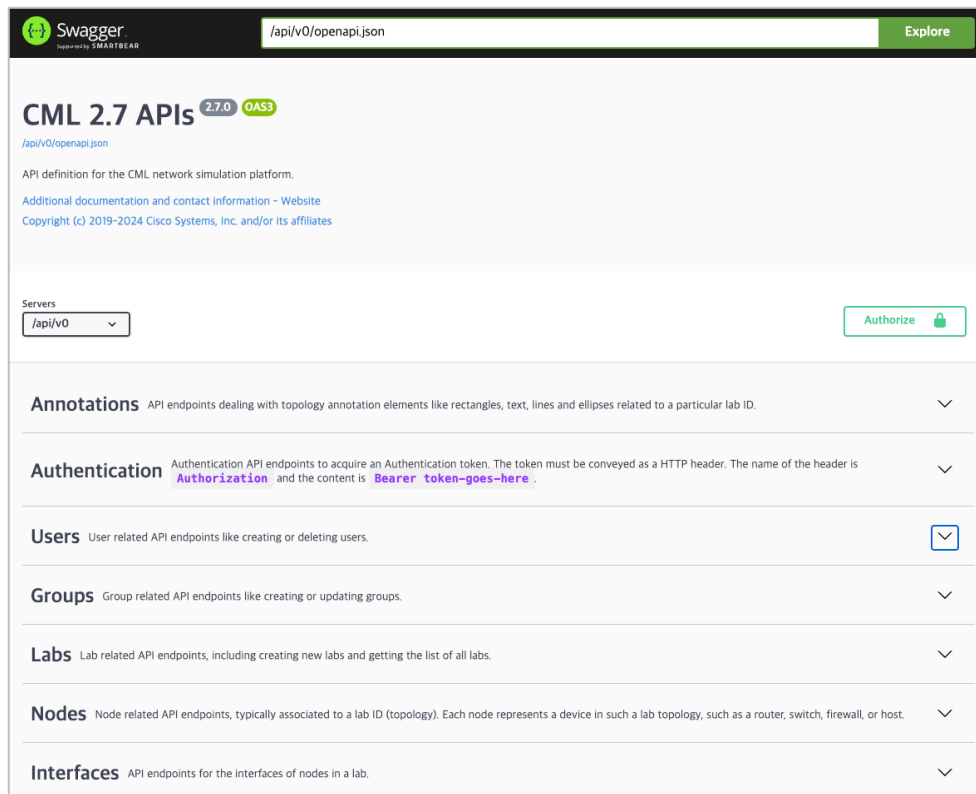
REST API: HTTP 상태 코드

- 응답의 결과를 코드로 나타냄

Status Code	Status Message	Meaning
200	OK	All looks good
201	Created	New resource created
400	Bad Request	Request was invalid
401	Unauthorized	Authentication missing or incorrect
403	Forbidden	Request was understood but not allowed
404	Not Found	Resource not found
500	Internal Server Error	Something wrong with the server
503	Service Unavailable	Server is unable to complete request

REST API 문서

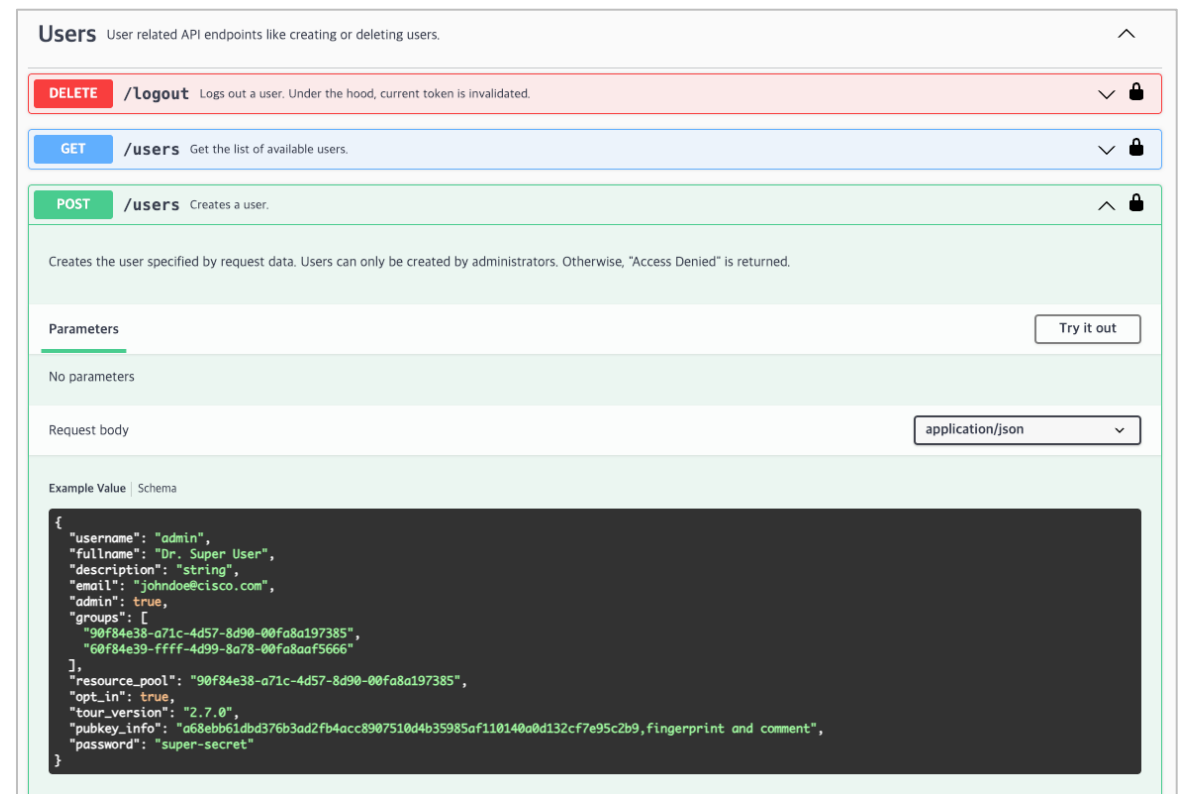
- developer.cisco.com 또는 운용 중인 솔루션 웹페이지에서 확인



Swagger UI interface for CML 2.7.0 APIs. The interface shows the API definition for the CML network simulation platform. The URL bar displays `/api/v0/openapi.json` with an **Explore** button. The main content area lists various API endpoints categorized by function:

- Annotations**: API endpoints dealing with topology annotation elements like rectangles, text, lines and ellipses related to a particular lab ID.
- Authentication**: Authentication API endpoints to acquire an Authentication token. The token must be conveyed as a HTTP header. The name of the header is `Authorization` and the content is `Bearer token-goes-here`.
- Users**: User related API endpoints like creating or deleting users. (Selected)
- Groups**: Group related API endpoints like creating or updating groups.
- Labs**: Lab related API endpoints, including creating new labs and getting the list of all labs.
- Nodes**: Node related API endpoints, typically associated to a lab ID (topology). Each node represents a device in such a lab topology, such as a router, switch, firewall, or host.
- Interfaces**: API endpoints for the interfaces of nodes in a lab.

The interface also includes a **Servers** dropdown menu set to `/api/v0` and an **Authorize** button.



Swagger UI interface for the **Users** API endpoint. The endpoint is `/users` with a **POST** method. The description states: "Creates a user. Under the hood, current token is invalidated." The interface shows the request body and an example value.

DELETE `/logout` Logs out a user. Under the hood, current token is invalidated.

GET `/users` Get the list of available users.

POST `/users` Creates a user.

Creates the user specified by request data. Users can only be created by administrators. Otherwise, "Access Denied" is returned.

Parameters: No parameters.

Request body: application/json

Example Value

```
{
  "username": "admin",
  "fullname": "Dr. Super User",
  "description": "string",
  "email": "johndoe@cisco.com",
  "admin": true,
  "groups": [
    "90f84e38-a71c-4d57-8d90-00fa8a197385",
    "60f84e39-ffff-4d99-8a78-00fa8aaf5666"
  ],
  "resource_pool": "90f84e38-a71c-4d57-8d90-00fa8a197385",
  "opt_in": true,
  "tour_version": "2.7.0",
  "pubkey_info": "a68ebb61dbd376b3ad2fb4acc8907510d4b35985af110140a0d132cf7e95c2b9,fingerprint and comment",
  "password": "super-secret"
}
```

REST API 실습

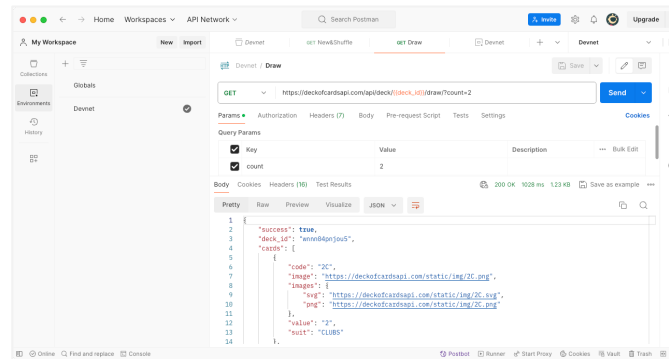


REST API 실습 도구

1. cURL

```
kidkim@KIDKIM-M-69YJ ~ % curl https://deckofcardsapi.com/api/deck/new/
{"success": true, "deck_id": "n2ijhc141hm5", "remaining": 52, "shuffled": false}
kidkim@KIDKIM-M-69YJ ~ %
```

2. Postman

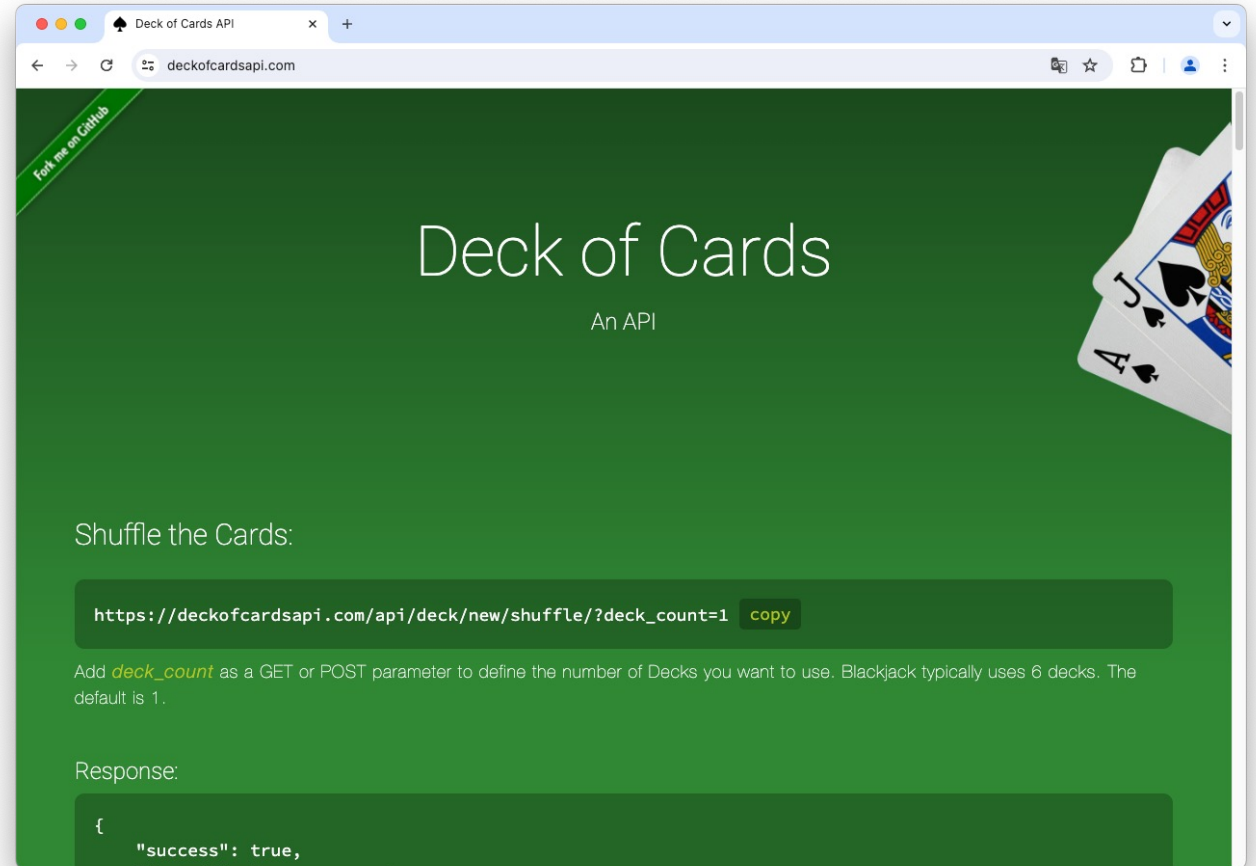


3. Python

```
main.py
1 import requests
2
3 url = "https://deckofcardsapi.com/api/deck/wnnn04pnjou5/draw/?count=5"
4
5 payload = {}
6 headers = {}
7
8 response = requests.request("GET", url, headers=headers, data=payload)
9
10 print(response.text)
11
```

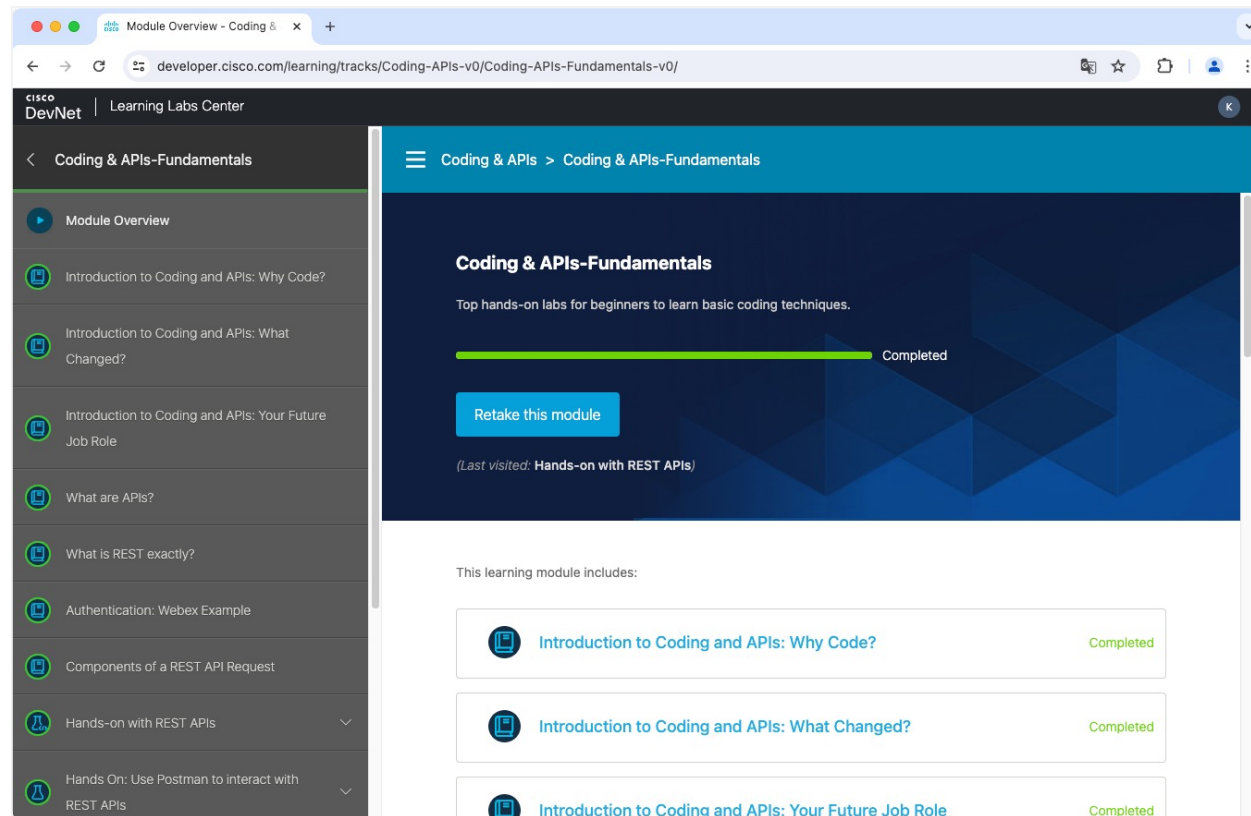
실습 API

- 카드 덱 생성, 카드 섞기, 카드 뽑기
- 링크: <https://deckofcardsapi.com/>



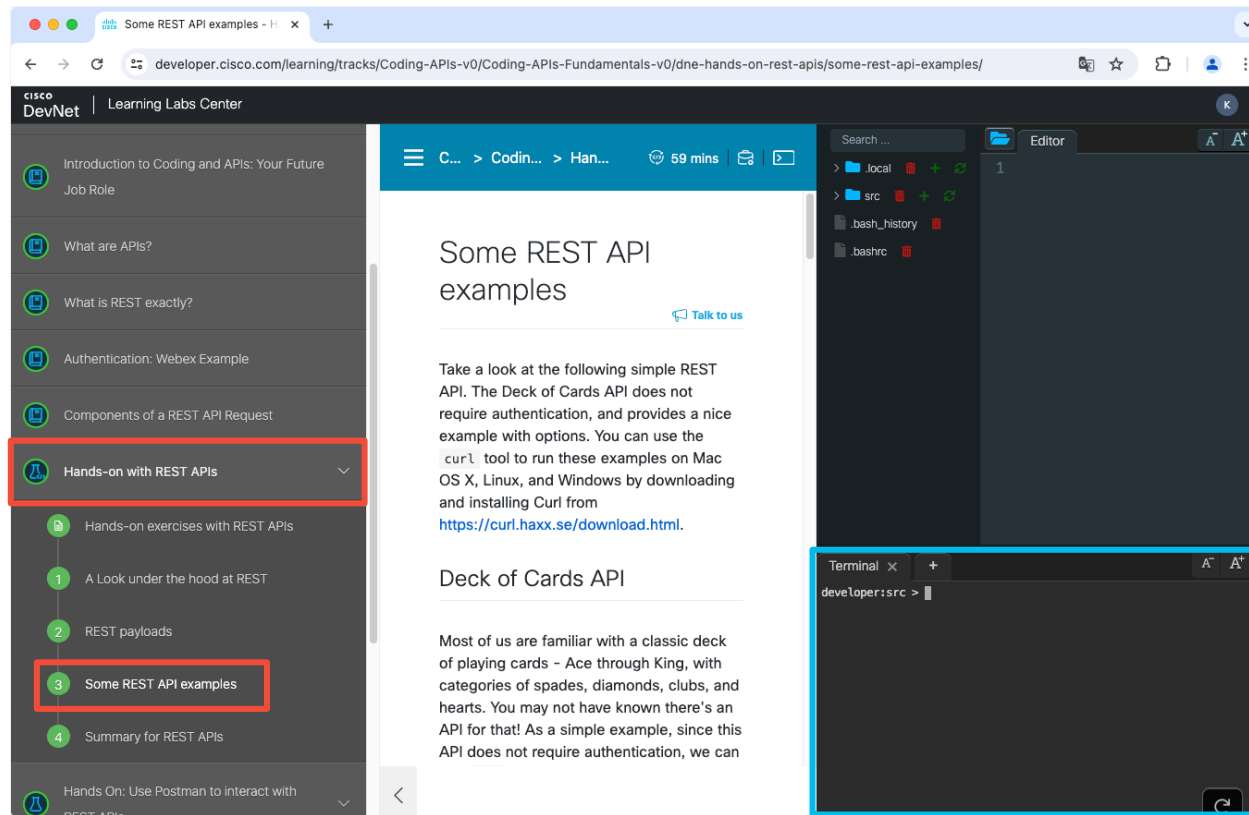
실습 환경: Devnet Learning Lab

- <https://developer.cisco.com/learning/tracks/Coding-APIs-v0/Coding-APIs-Fundamentals-v0/>



cURL 실습환경

- 우측 목록에서 'Hands-on with REST APIs' - 'Some Rest API examples' 로 이동




← 이곳에 커맨드를 입력

cURL

- URL을 사용하여 데이터를 전송하는 오픈 소스 도구




Download Documentation libcurl Get Help Development News

Please donate a few minutes of your time to [the curl user survey 2024](#)



command line tool and library
for transferring data with URLs
(since 1998)

Supports...
DICT, FILE, FTP, FTPS, GOPHER, GOPHERS, HTTP, HTTPS, IMAP, IMAPS, LDAP, LDAPS, MQTT, POP3, POP3S, RTMP, RTMPS, RTSP, SCP, SFTP, SMB, SMBS, SMTP, SMTPS, TELNET, TFTP, WS and WSS. curl supports TLS certificates, HTTP POST, HTTP PUT, FTP uploading, HTTP form based upload, proxies (SOCKS4, SOCKS5, HTTP and HTTPS), HTTP/2, HTTP/3, cookies, user+password authentication (Basic, Plain, Digest, CRAM-MD5, SCRAM-SHA, NTLM, Negotiate, Kerberos, Bearer tokens and AWS Sigv4), file transfer resume, proxy tunneling, HSTS, Alt-Svc, unix domain sockets, HTTP compression (gzip, brotli and zstd), etags, parallel transfers, DNS-over-HTTPS and more.

Top Sponsors




<https://curl.se/>

cURL 실습: 새 카드 덱 요청하기

- curl <https://deckofcardsapi.com/api/deck/new/>

A Brand New Deck:

<https://deckofcardsapi.com/api/deck/new/> [copy](#)

Open a brand new deck of cards.

A-spades, 2-spades, 3-spades... followed by diamonds, clubs, then hearts.

Hot Tip: Add `jokers_enabled=true` as a GET or POST parameter to your request to include two Jokers in the deck.

Response:

```
{
  "success": true,
  "deck_id": "3p40paa87x90",
  "shuffled": false,
  "remaining": 52
}
```

deck_id 중요!



cURL 실습: 새 카드 덱 요청하기

- 미션에서 사용하게 될 [deck_id](#) 확인!
- curl <https://deckofcardsapi.com/api/deck/new/>

```
Terminal x + A- A+
developer:src > curl https://deckofcardsapi.com/api/deck/new/
{"success": true, "deck_id": "w6fb1k0032nf", "remaining": 52,
"shuffled": false}developer:src >
```

- curl <https://deckofcardsapi.com/api/deck/new/> | python -m json.tool

```
Terminal x + A- A+
developer:src > curl https://deckofcardsapi.com/api/deck/new/
| python -m json.tool
% Total % Total % Received % Xferd Average Speed Time Time
Time Current
Dload Upload Total Spent
t Left Speed
0 0 0 0 0 0 0 0 --:--:-- --:--:--
100 80 100 80 0 0 228 0 --:--:-- --:--:--
-- --:--:-- 228
{
  "success": true,
  "deck_id": "81bfqj4ysoa1",
  "remaining": 52,
  "shuffled": false
}
developer:src >
```

cURL 미션: 카드 2장 뽑기



- 카드 2장 뽑는 요청을 실행합니다.

Draw a Card:

```
https://deckofcardsapi.com/api/deck/⟨deck_id⟩/draw/?count=2
```

copy

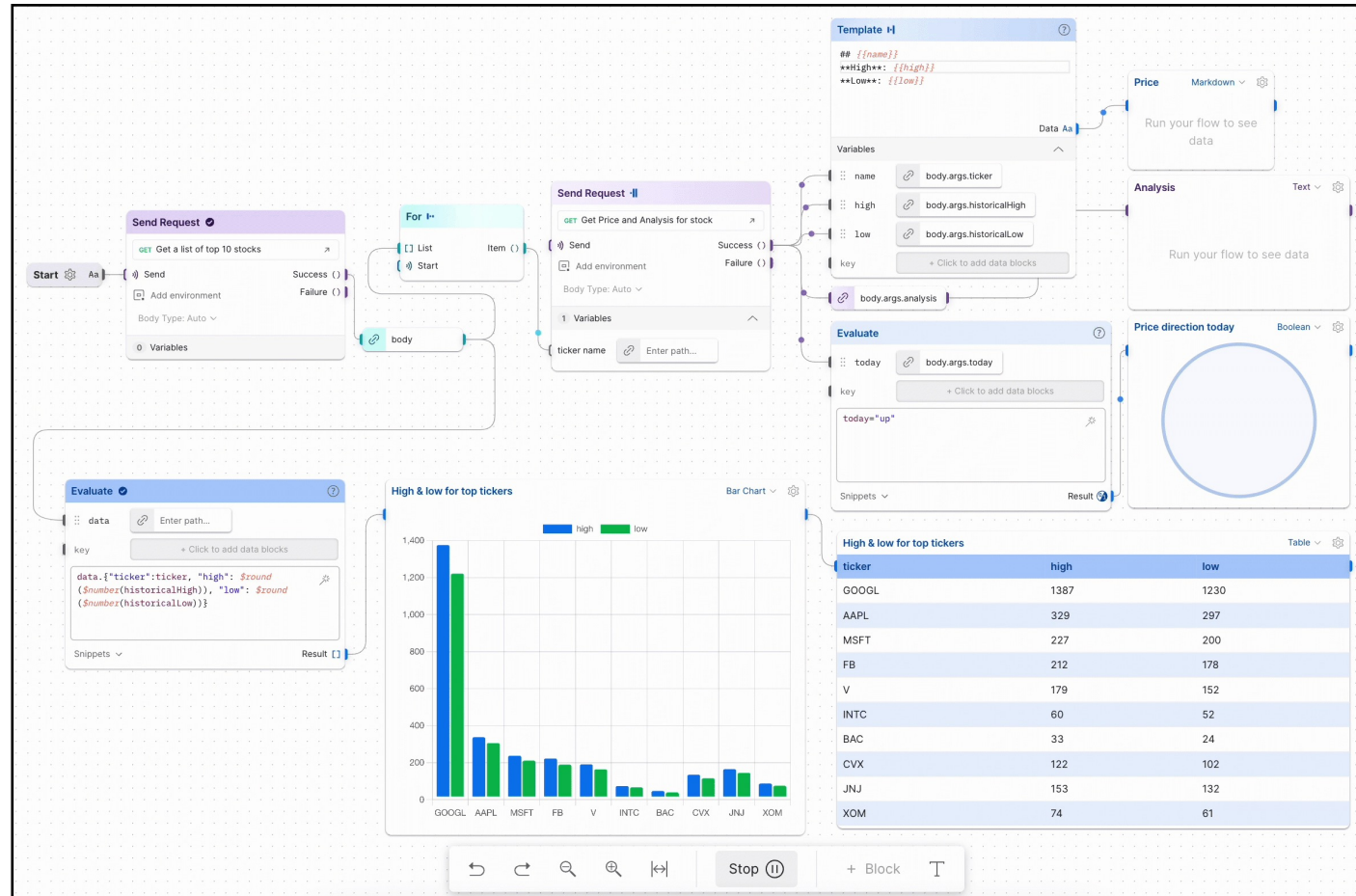
The count variable defines how many cards to draw from the deck. Be sure to replace *deck_id* with a valid deck_id. We use the deck_id as an identifier so we know who is playing with what deck. After two weeks, if no actions have been made on the deck then we throw it away.

TIP: replace *deck_id* with "new" to create a shuffled deck and draw cards from that deck in the same request.

- https://deckofcardsapi.com/api/deck/⟨deck_id⟩/draw/?count=2 사용
- <deck_id> 부분을 이전 실습에서 얻은 실제 deck_id로 교체

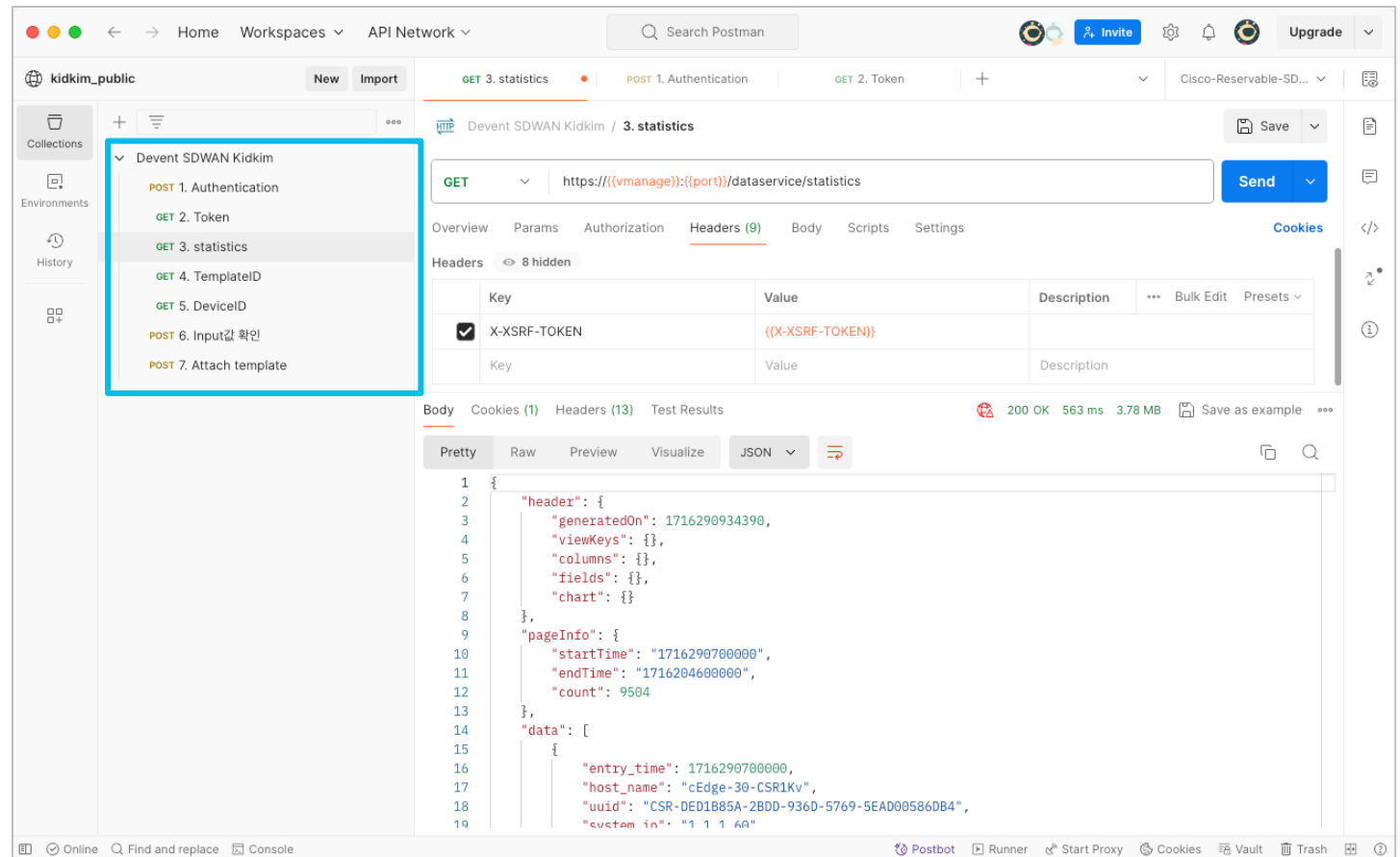
Postman

- API를 효과적으로 구축, 관리, 사용할 수 있도록 지원하는 통합 소프트웨어



Postman 실습: Collection

- 반복적인 테스트 수행에 용이
- 다양한 요청을 그룹화하고 폴더로 구성하여 관리
- 다른 사용자와 공유 가능



Postman 실습: Environment

- request에서 사용할 변수를 설정

Environment configuration for **Cisco-Reservable-SD-WAN-Environment**

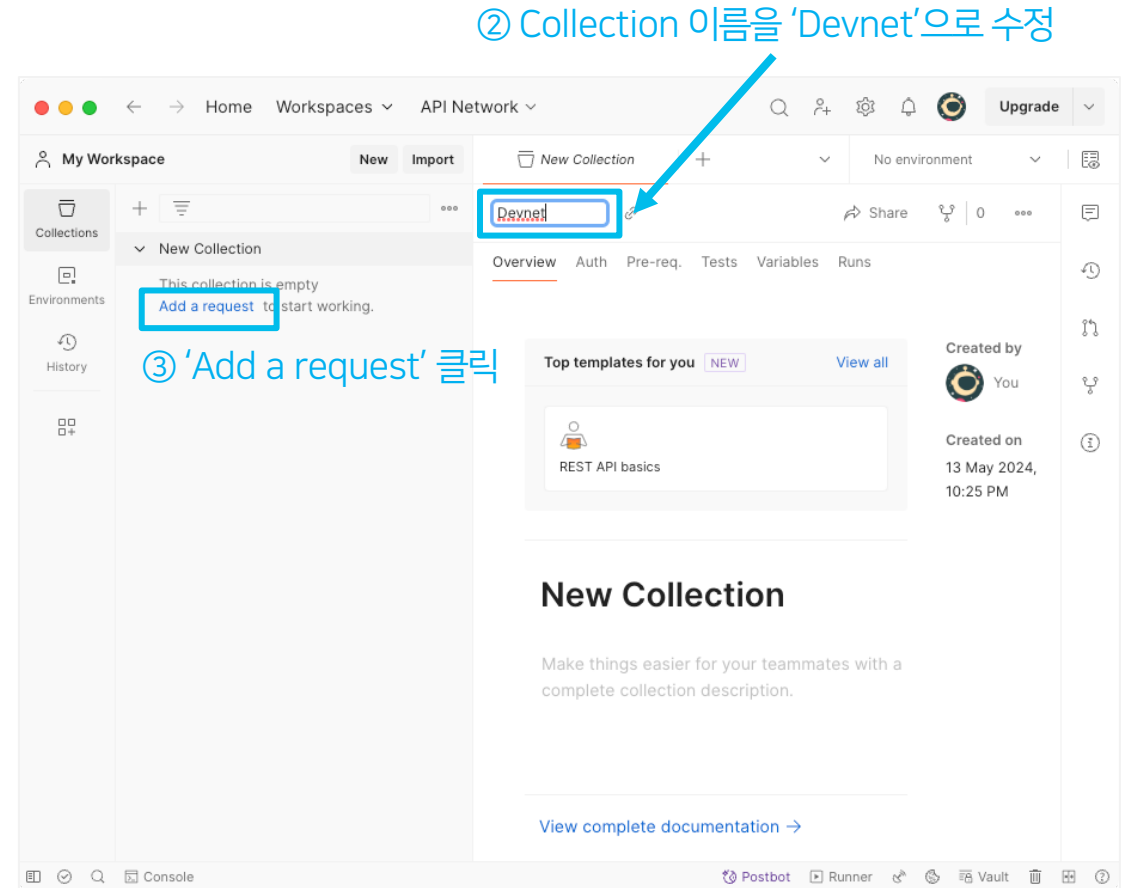
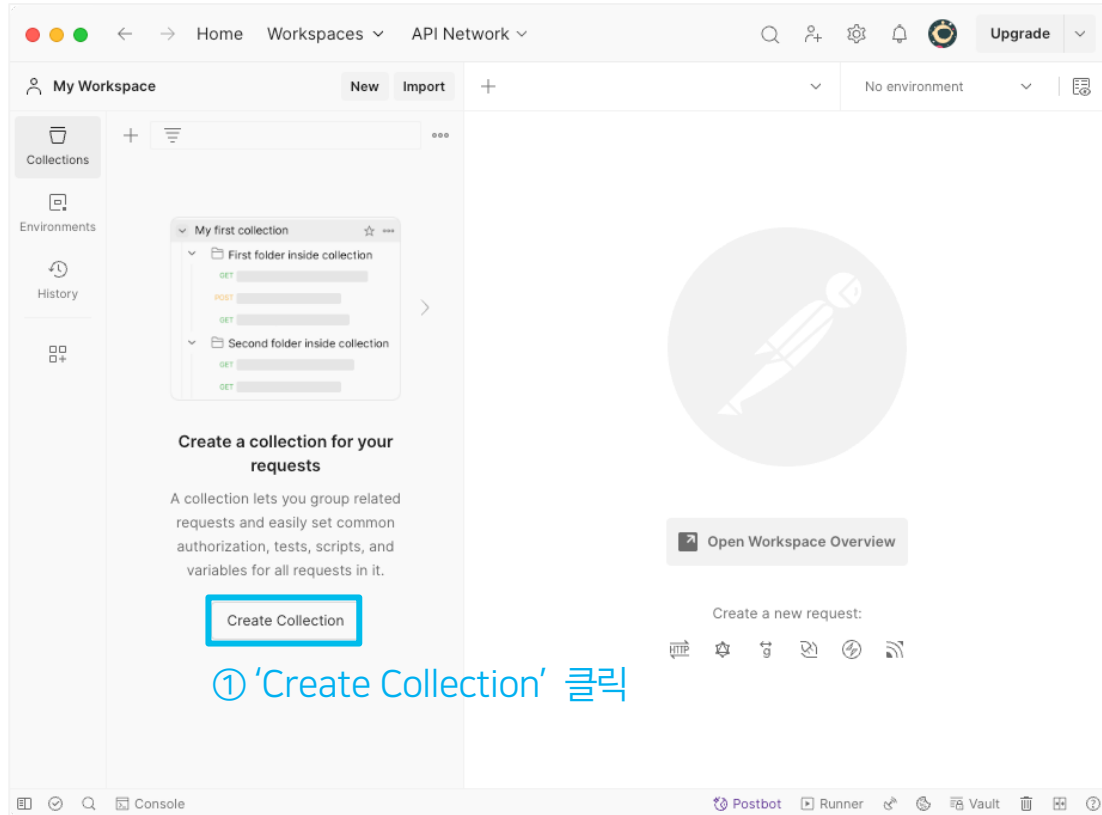
Variable	Type	Initial value	Current value
<input checked="" type="checkbox"/> vmanage	default	192.168.13.21	192.168.13.21
<input checked="" type="checkbox"/> j_username	default	admin	admin
<input checked="" type="checkbox"/> j_password	secret
<input checked="" type="checkbox"/> port	default	8443	8443
<input checked="" type="checkbox"/> X-XSRF-TOKEN	default		AD32A6DC7F0AD913C561AF1049B9AA...
<input checked="" type="checkbox"/> hostname	default	cEdge-00-C8Kv	cEdge-00-C8Kv
<input checked="" type="checkbox"/> deviceIP	default	192.168.13.	192.168.13.
<input checked="" type="checkbox"/> deviceId	default		
<input checked="" type="checkbox"/> templateId	default		
Add new variable			

Use variables to reuse values and protect sensitive data

Store sensitive data in variable type secret to keep its values masked on the screen. Learn more about [variable type](#).
Work with the current value of a variable to prevent sharing sensitive values with your team. Learn more about [variable values](#).

중요한 정보 마스킹

Postman 실습: Collection 초기 설정



Postman 실습: 카드가 섞인 덱을 요청하기 - 1

- https://deckofcardsapi.com/api/deck/new/shuffle/?deck_count=1

Shuffle the Cards:

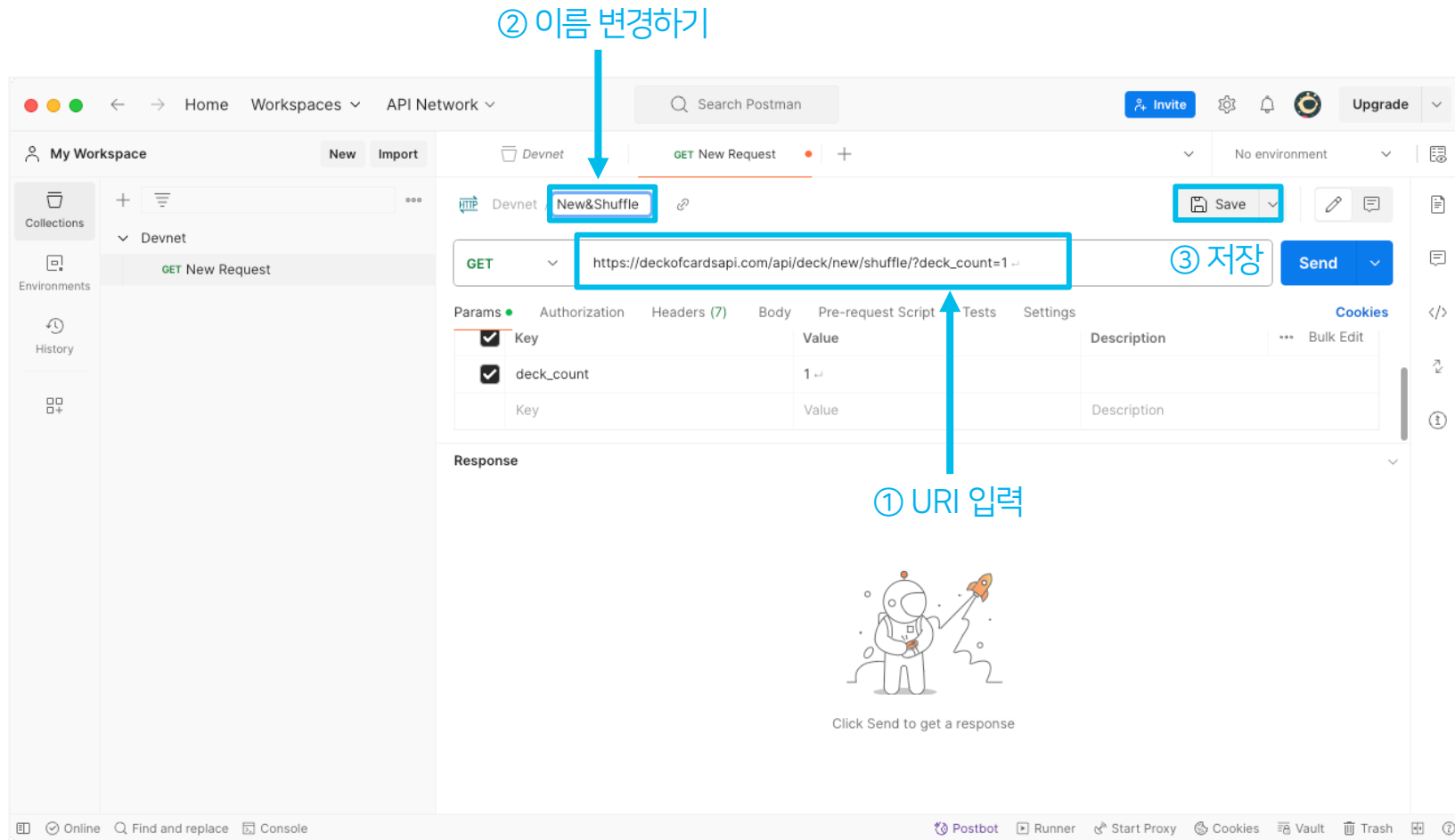
https://deckofcardsapi.com/api/deck/new/shuffle/?deck_count=1 [copy](#)

Add **deck_count** as a GET or POST parameter to define the number of Decks you want to use. Blackjack typically uses 6 decks. The default is 1.

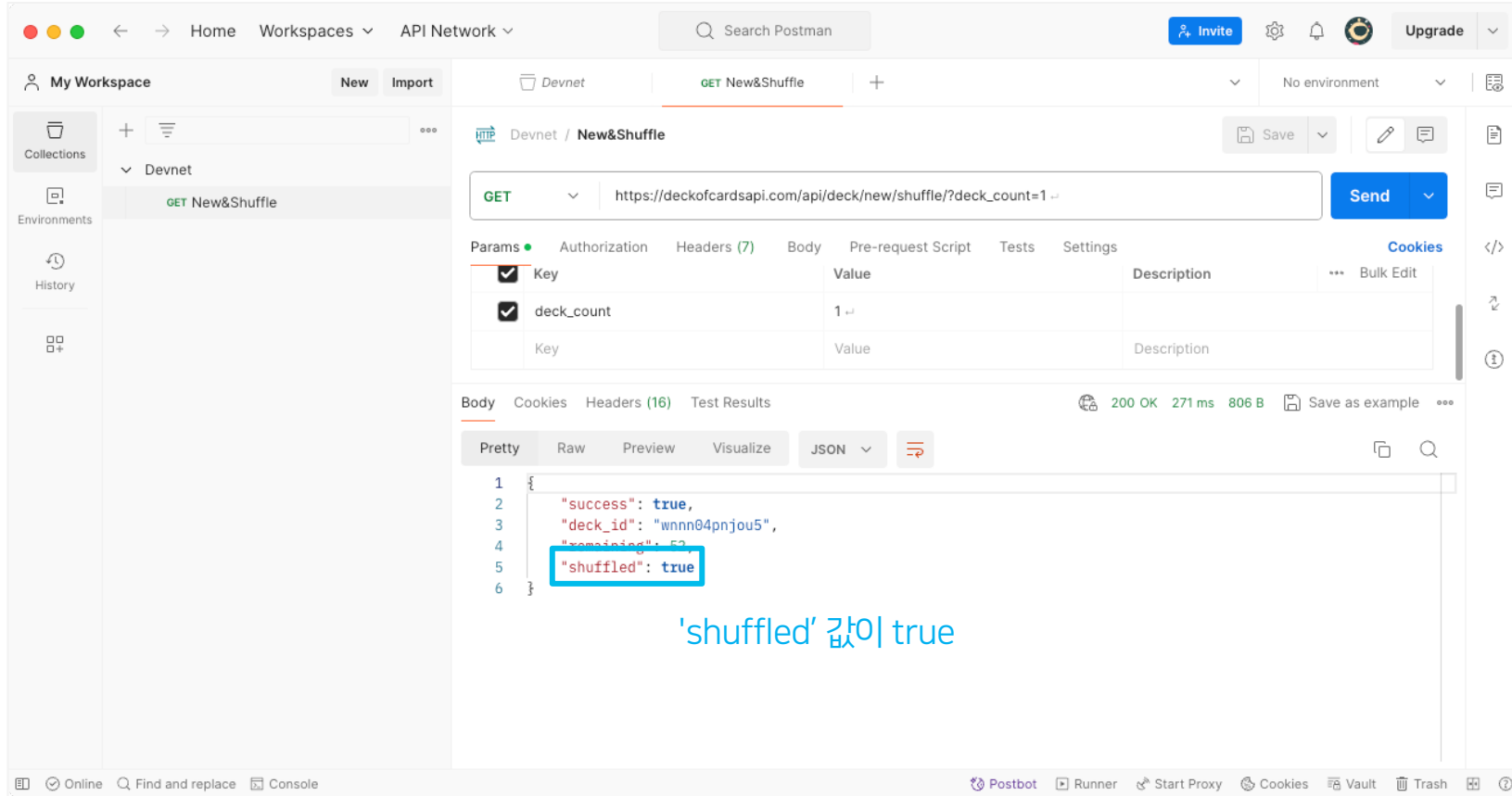
Response:

```
{
  "success": true,
  "deck_id": "3p40paa87x90",
  "shuffled": true,
  "remaining": 52
}
```

Postman 실습: 카드가 섞인 덱을 요청하기 - 2



Postman 실습: 카드가 섞인 덱을 요청하기 - 3



The screenshot shows the Postman interface with a GET request to `https://deckofcardsapi.com/api/deck/new/shuffle?deck_count=1`. The response status is 200 OK, 271 ms, 806 B. The response body is JSON, and the `"shuffled": true` field is highlighted with a red box.

Params

Key	Value	Description
deck_count	1	

Body

```
1 {
2   "success": true,
3   "deck_id": "wnnn04pnjou5",
4   "remaining": 52,
5   "shuffled": true
6 }
```

'shuffled' 값이 true

Postman 실습: Environment 설정 - 1

- 카드 2장 뽑는 Request를 생성
- https://deckofcardsapi.com/api/deck/%%deck_id%%/draw/?count=2 사용

Draw a Card:

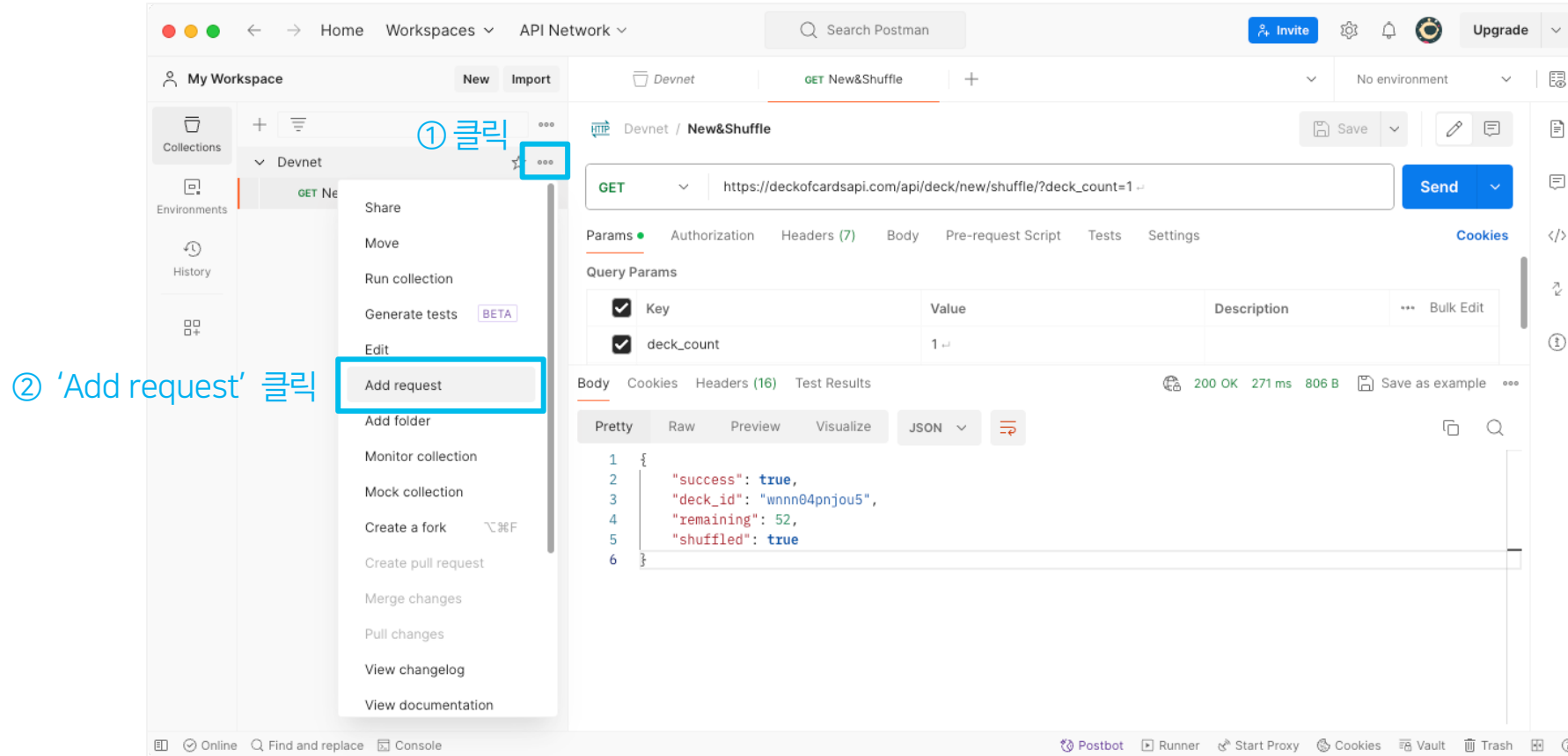
```
https://deckofcardsapi.com/api/deck/%%deck_id%%/draw/?count=2
```

copy

The count variable defines how many cards to draw from the deck. Be sure to replace `deck_id` with a valid deck_id. We use the deck_id as an identifier so we know who is playing with what deck. After two weeks, if no actions have been made on the deck then we throw it away.

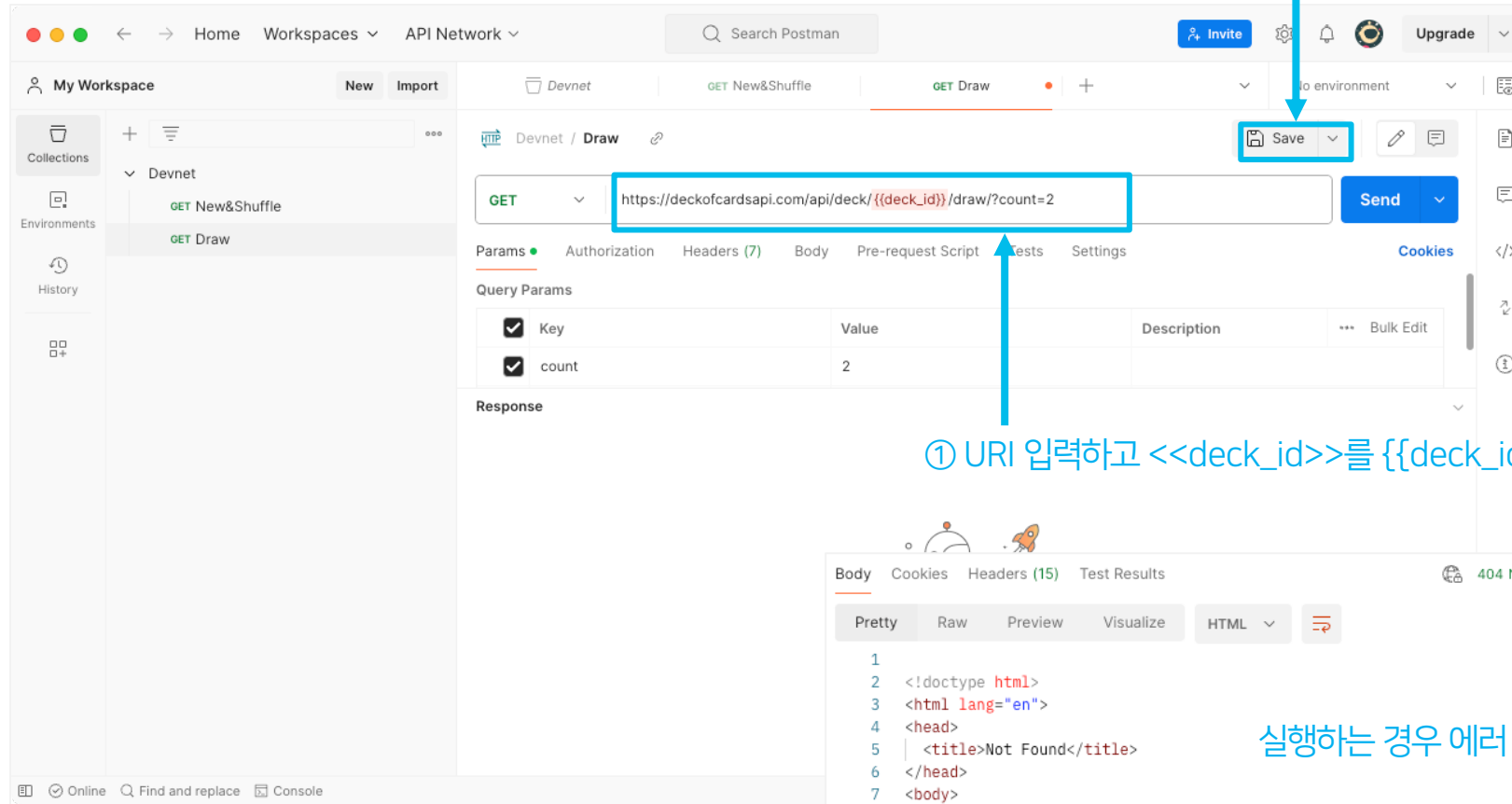
TIP: replace `%%deck_id%%` with `"new"` to create a shuffled deck and draw cards from that deck in the same request.

Postman 실습: Environment 설정 - 2



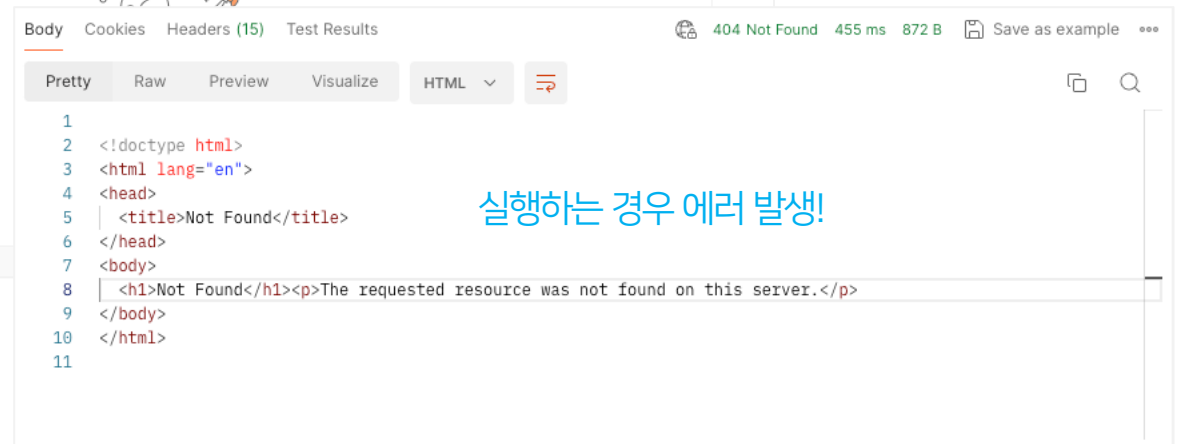
Postman 실습: Environment 설정 -3

② 이름 변경 후 저장



① URI 입력하고 <<deck_id>>를 {{deck_id}}로 변경

실행하는 경우 에러 발생!



Postman 실습: Environment 설정 - 4

① 'Environments' 클릭

My Workspace

Environments

You don't have any environments.
An environment is a set of variables that allows you to switch the context of your requests.

Create Environment

GET Draw

https://deckofcardsapi.com/api/deck/((deck_id))/draw/?count=2

Send

Params

Query Params

Key	Value	Description
count	2	

Body

404 Not Found 455 ms 872 B

Save as example

```
1 <!doctype html>
2 <html lang="en">
3 <head>
4   <title>Not Found</title>
5 </head>
6 <body>
7   <h1>Not Found</h1><p>The requested resource was not found on this server.</p>
8 </body>
9 </html>
10
11
```

② 'Create Environment' 클릭

Postman 실습: Environment 설정 - 5

The screenshot shows the Postman interface with the 'Devnet' environment selected. The left sidebar shows 'My Workspace' with 'Environments' expanded. The main panel displays the 'Devnet' environment settings. The 'Variable' column has a checkbox next to 'deck_id'. The 'Initial value' column has the value 'wnnn04pnjou5'. The 'Current value' column also shows 'wnnn04pnjou5'. The 'Save' button is visible in the top right. Annotations in blue text and boxes highlight specific elements: ③ 이름 변경 (Name change) points to the 'Devnet' tab; ④ 저장 (Save) points to the 'Save' button; ① 'device_id' 입력 (Input 'device_id') points to the 'deck_id' checkbox; ② 실제 device_id 입력 (Input actual device_id) points to the 'wnnn04pnjou5' value in the 'Initial value' column.

Variable	Type	Initial value	Current value
<input checked="" type="checkbox"/> deck_id	default	wnnn04pnjou5	wnnn04pnjou5

Postman 실습: Environment 설정 - 6

The screenshot shows the Postman interface with the 'Devnet' environment selected. The left sidebar shows 'My Workspace' with 'Environments' selected. The 'Devnet' environment is highlighted with a blue box and a checkmark icon. A blue arrow points to the 'Devnet' dropdown menu in the top right, labeled '활성화 확인' (Check activation). The main panel shows the 'Devnet' environment variables table.

Variable	Type	Initial value	Current value
<input checked="" type="checkbox"/> deck_id	default	wnnn04pnjou5	wnnn04pnjou5
Add new variable			

① 클릭하여 활성화

Postman 실습: Environment 설정 - 7

The screenshot shows the Postman application interface. On the left, the 'My Workspace' sidebar is visible with 'Environments' selected, showing a 'Devnet' environment. The main panel displays a 'GET' request to the URL 'https://deckofcardsapi.com/api/deck/((deck_id))/draw?count=2'. The 'Query Params' section shows a table with one parameter: 'count' with a value of '2'. The 'Body' tab is selected, showing a JSON response in 'Pretty' format. The response is a successful draw of two cards from a deck.

Key	Value	Description
count	2	

```
1 {
2   "success": true,
3   "deck_id": "wnnn04pnjou5",
4   "cards": [
5     {
6       "code": "2C",
7       "image": "https://deckofcardsapi.com/static/img/2C.png",
8       "images": {
9         "svg": "https://deckofcardsapi.com/static/img/2C.svg",
10        "png": "https://deckofcardsapi.com/static/img/2C.png"
11      },
12       "value": "2",
13       "suit": "CLUBS"
14     }
15   ]
16 }
```

결과 확인

Postman 미션 : 변수 추가



- Environment에서 뽑는 개수를 변수 이름 count로 생성하고, 5로 설정

Draw a Card:

```
https://deckofcardsapi.com/api/deck/ deck_id /draw/?count=2
```

The count variable defines how many cards to draw from the deck. Be sure to replace `deck_id` with a valid deck_id. We use the deck_id as an identifier so we know who is playing with what deck. After two weeks, if no actions have been made on the deck then we throw it away.

TIP: replace `deck_id` with "new" to create a shuffled deck and draw cards from that deck in the same request.

- Request의 '`?count=2`' 부분을 '`?count={{count}}`'로 변경

Python 실습 1: Postman에서 코드 복사

① 클릭

② cURL 을 Python - Requests로 변경

The screenshot shows the Postman interface with a GET request to `https://deckofcardsapi.com/api/deck/wnnn04pnjou5/draw/?count=5`. The response is a JSON object indicating success and providing card details. A 'Code snippet' panel on the right shows the equivalent Python code using the `requests` library. Two blue arrows point to the code snippet panel: one to the code editor icon and another to the 'Python - Requests' dropdown menu.

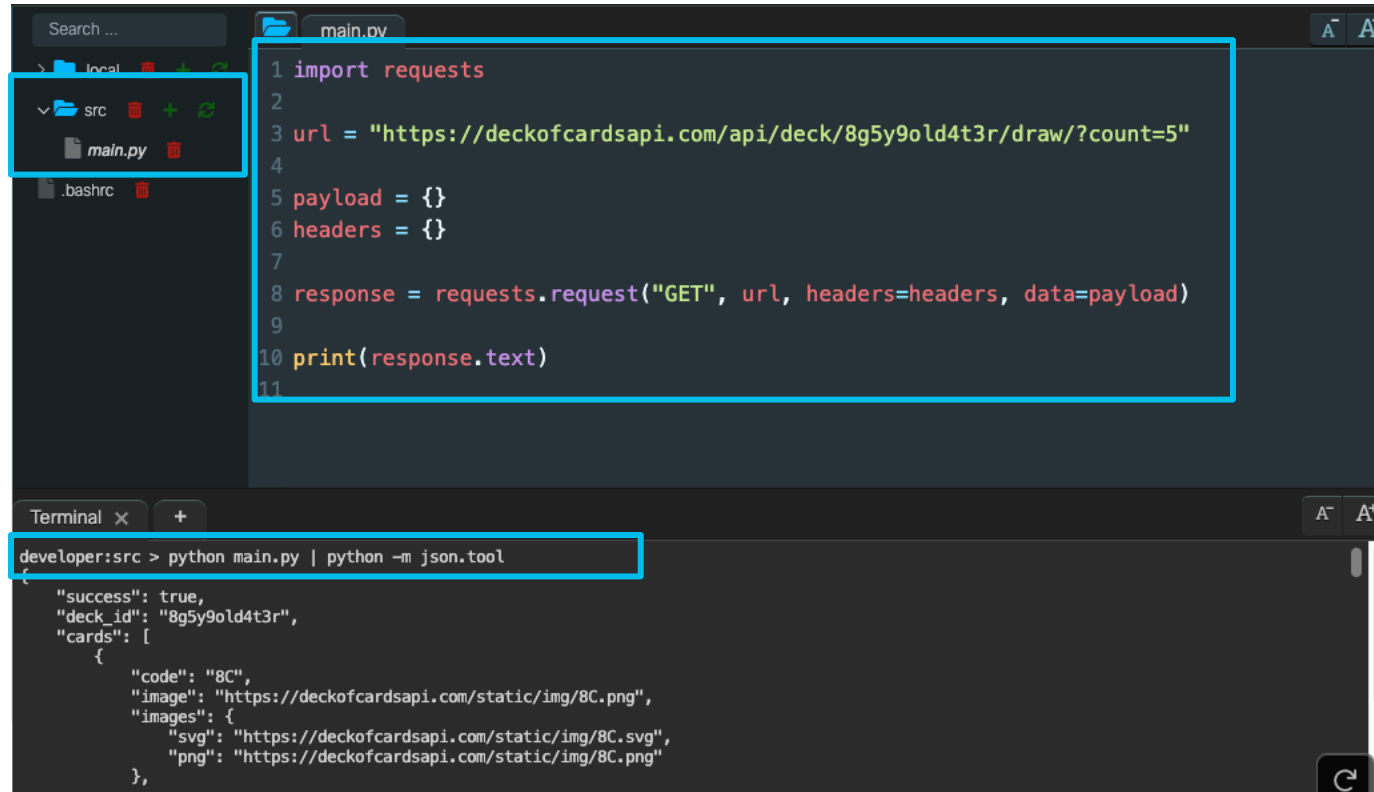
```
1 import requests
2
3 url = "https://deckofcardsapi.com/api/deck/wnnn04pnjou5/draw/?count=5"
4
5 payload = {}
6 headers = {}
7
8 response = requests.request("GET", url, headers=headers, data=payload)
9
10 print(response.text)
11
```

Python 실습: 코드 실행

② Postman에서 복사한 코드 붙여넣기(자동저장)

① src폴더의 main.py 클릭

③ 실행



The screenshot shows a code editor with a file named `main.py` open. The code in the editor is as follows:

```
1 import requests
2
3 url = "https://deckofcardsapi.com/api/deck/8g5y9old4t3r/draw/?count=5"
4
5 payload = {}
6 headers = {}
7
8 response = requests.request("GET", url, headers=headers, data=payload)
9
10 print(response.text)
11
```

Below the code editor is a terminal window. The command entered in the terminal is:

```
developer:src > python main.py | python -m json.tool
```

The output of the command is a JSON object:

```
{
  "success": true,
  "deck_id": "8g5y9old4t3r",
  "cards": [
    {
      "code": "8C",
      "image": "https://deckofcardsapi.com/static/img/8C.png",
      "images": {
        "svg": "https://deckofcardsapi.com/static/img/8C.svg",
        "png": "https://deckofcardsapi.com/static/img/8C.png"
      }
    }
  ]
}
```

Python 실습: 변수 설정

① deckID를 변수로 생성

② URL에 변수 사용

```
1 import requests
2
3 deckID = "8g5y9old4t3r"
4
5 url = "https://deckofcardsapi.com/api/deck/" + deckID + "/draw/?count=5"
6
7 payload = {}
8 headers = {}
9
10 response = requests.request("GET", url, headers=headers, data=payload)
11
12 print(response.text)
```


Python 실습: 새 덱을 요청하고, 5장 뽑기

새로 추가

```
1 import requests
2
3 url = "https://deckofcardsapi.com/api/deck/new"
4
5 payload = {}
6 headers = {}
7
8 response = requests.request("GET", url, headers=headers, data=payload)
9
10 deckID = response.json()["deck_id"]
11
12 url = "https://deckofcardsapi.com/api/deck/" + deckID + "/draw/?count=5"
13
14 payload = {}
15 headers = {}
16
17 response = requests.request("GET", url, headers=headers, data=payload)
18
19 print(response.text)
```

응답 결과의 'deck_id' 값을 변수로 지정

```
{
  "success": true,
  "deck_id": "od9dt6xuuzcj",
  "remaining": 52,
  "shuffled": false
}
```

Python 실습: 새 덱을 요청하고, 5장 뽑기

- 실행 결과

```
developer:src > python main.py | python -m json.tool
{
  "success": true,
  "deck_id": "99og3sxl0pus",
  "cards": [
    {
      "code": "AS",
      "image": "https://deckofcardsapi.com/static/img/AS.png",
      "images": {
        "svg": "https://deckofcardsapi.com/static/img/AS.svg",
        "png": "https://deckofcardsapi.com/static/img/AS.png"
      },
      "value": "ACE",
      "suit": "SPADES"
    },
    {
      "code": "2S",
      "image": "https://deckofcardsapi.com/static/img/2S.png",
      "images": {
        "svg": "https://deckofcardsapi.com/static/img/2S.svg",
        "png": "https://deckofcardsapi.com/static/img/2S.png"
      },
      "value": "2",
      "suit": "SPADES"
    },
    {
      "code": "3S",
      "image": "https://deckofcardsapi.com/static/img/3S.png",
      "images": {
        "svg": "https://deckofcardsapi.com/static/img/3S.svg",
        "png": "https://deckofcardsapi.com/static/img/3S.png"
      },
      "value": "3",
      "suit": "SPADES"
    },
    {
      "code": "4S",
```



Thank you