APIs (Application Programming Interface)

Verb	Endpoint	Parameters	Response
GET	/user/[username: String]	{ }	200 OK, User
DELETE	/user/[username: String]	{ }	200 OK, Result
PATCH	/user/[username: String]	{"email": String}	200 OK, User

Application Programming Interfaces are structured methods for devices to communicate with a web application's backend

APIs (Application Programming Interface)

Verb	Endpoint	Parameters	Response
GET	/user/[username: String]	{ }	200 OK, User
DELETE	/user/[username: String]	{ }	200 OK, Result
PATCH	/user/[username: String]	{"email": String}	200 OK, User

Put in plain language, we can say that APIs allow devices to communicate and perform actions on web apps

APIs (Application Programming Interface)

```
Method  Endpoint

GET     /user/[username]
```

In the example above, we see that sending a GET request to the /user/<username> endpoint will retrieve user data

APIs (Application Programming Interface)

```
Method  Endpoint

DELETE  /user/[username]
```

But a request to the same endpoint, but with the DELETE method will delete the specified user from the system

APIs (Application Programming Interface)

```
Method  Endpoint              Parameters

PATCH   /user/[username]  {"email": String}
```

One last action we can do with this API is use the PATCH method to the same endpoint to change the user's associated email address

APIs (Application Programming Interface)

```
Method  Endpoint              Parameters

PATCH   /user/[username]  {"email": String}
```

With few exceptions, all data sent to API endpoints will be in JSON format

APIs Documentation

Web apps often host a page with documentation for their APIs, which can be used to better understand how to enumerate them

API Valid Method Enumeration

In cases where we can identify a valid API endpoint, it's often beneficial to try different HTTP methods with it, to see if you can find different valid responses

```
Method  Endpoint

GET     /user/[username]
```

```
Method  Endpoint

DELETE  /user/[username]
```

API Mass Assignment

```
{
    "id": 123,
    "name": "John Doe",
    "email": "john@example.com",
    "isAdmin": "false"
}
```

In certain APIs, it is possible to assign values to parameters that are not specified in the original request

API Mass Assignment

```
{
    "id": 123,
    "name": "John Doe",
    "email": "john@example.com",
    "isAdmin": "false"
}
```

For example, an API GET request may retrieve the following information about a user on the system. This lets us know there is an **isAdmin** parameter for each user

API Mass Assignment

```
{
   "name": "HackerFrogs",
   "email": "hacker@hackerfrogs.com"
}
```

If, during the user creation process, we see an API POST request that includes the above data, we could try to give our additional privileges...

API Mass Assignment

```
{
   "name": "HackerFrogs",
   "email": "hacker@hackerfrogs.com",
   "isAdmin": "true"
}
```

By adding an **isAdmin** parameter with our user creation post request, and if successful, would be a classic example of mass assignment vulnerability