<u>HTTP Methods Best Practices</u>

The Hypertext Transfer Protocol (HTTP) is designed to enable communications between clients and servers.

HTTP works as a request-response protocol between a client and server.

HTTP Methods

- GET
- POST
- PUT
- HEAD
- DELETE
- PATCH
- OPTIONS
- CONNECT
- TRACE

HTTP specifies a collection of request methods to specify what action is to be performed on a particular resource. The most commonly used HTTP request methods are **GET, POST, PUT, PATCH, and DELETE**. These are equivalent to the **CRUD operations (create, read, update, and delete).**

GET: GET request is used to read/retrieve data from a web server. GET returns an HTTP status code of **200 (OK)** if the data is successfully retrieved from the server.

We need to know__

a. Create simple request

b. Simple get request with URL parameter.

c. Catch request header simply get method.

We can create a simple get request using express js__

```
let express = require('express');
app = express ();
app.get('/', function (req, res) {
   res.send("Hello Express Js");
})
app.listen(5050);
console.log ("server run successfully");
```

The program run on browser with 5050 port.

POST: POST request is used to send data (file, form data, etc.) to the server. On successful creation, it returns an HTTP status code of **201**.

We need to know__

a. Simple post request.

b. Post request with URL parameter.

c. Request Header.

d. Request JSON body

e. Request multipart form data

f. Request file upload

JavaScript HTTP POST Request Example:

```
var url = "https://reqbin.com/echo/post/json";

var xhr = new XMLHttpRequest();
xhr.open("POST", url);
xhr.setRequestHeader("Accept", "application/json");
xhr.setRequestHeader("Content-Type", "application/json");

xhr.onreadystatechange = function () {
  if (xhr.readyState === 4) {
    console.log(xhr.status);
    console.log(xhr.responseText);
  }};

xhr.send('{"Id": 78912, "Customer": "Jason Sweet"}');
```

PUT: A PUT request is used to modify the data on the server. It replaces the entire content at a particular location with data that is passed in the body payload. If there are no resources that match the request, it will generate one.

We can create a simple put request using express js__

```
let express = require('express');
app = express ();
app.put('/', function (req, res) {
   res.send("Hello Express Js");
})
app.listen(5050);
console.log ("server run successfully");
```

The program run on browser with 5050 port.

PATCH: PATCH is similar to PUT request, but the only difference is, it modifies a part of the data. It will only replace the content that you want to update.

We can create a simple patch request using express js__

```
let express = require('express');
app = express ();
app.patch('/', function (req, res) {
   res.send("Hello Express Js");
})
app.listen(5050);
console.log ("server run successfully");
```

The program run on browser with 5050 port.

DELETE: A DELETE request is used to delete the data on the server at a specified location.

We can create a simple delete request using express js__

```
let express = require('express');
app = express ();
app.delete('/', function (req, res) {
   res.send("Hello Express Js");
})
app.listen(5050);
console.log ("server run successfully");
```