HTTP Methods :

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

**GET :**

Get is used to request some data from a specified resource.

URL : [https://www.google.com/search?q=hi&nk](https://www.google.com/search?q=hi&nkj)

* GET requests can be cached
* GET requests remain in the browser history
* GET requests can be bookmarked
* GET requests should never be used when dealing with sensitive data
* GET requests have length restrictions
* GET requests are only used to request data (not modify)

**POST :**

Post Method is used to post some data to the servers to create or update the data.

POST requests are never cached

POST requests do not remain in the browser history

POST requests cannot be bookmarked

POST requests have no restrictions on data length

**PUT :**

PUT is used to send data to a server to create/update a resource.

The difference between POST and PUT is that PUT requests are idempotent. That is, calling the same PUT request multiple times will always produce the same result. In contrast, calling a POST request repeatedly has side effects of creating the same resource multiple times.

**HEAD :**

HEAD is almost identical to GET, but without the response body.

In other words, if GET /users returns a list of users, then HEAD /users will make the same request but will not return the list of users.

A HEAD request is useful for checking what a GET request will return before actually making a GET request - a HEAD request can read the Content-Length header to check the size of the file, without actually downloading the file.

**DELETE :**

The DELETE method deletes the specified resource.

**PATCH :**   
The PATCH method is used to apply partial modifications to a resource.

**OPTIONS :**

The OPTIONS method describes the communication options for the target resource.

**CONNECT :**

The CONNECT method is used to start two-way communications (a tunnel) with the requested resource.

**TRACE :**

The TRACE method is used to perform a message loop-back test that tests the path for the target resource (useful for debugging purposes).

**HTTP Status Codes :**

**Informational responses**

**100 Continue**

This code response indicates that the client should continue the request or ignore the response if the request is already finished.

**103 Early Hints**

This status code is primarily intended to be used with the Link header, letting the user agent start preloading resources while the server prepares a response or preconnect to an origin from which the page will need resources.

**Successful responses**

**200 OK**

The request succeeded. The result and meaning of "success" depends on the HTTP methods.

**201 Created**

The request succeeded, and a new resource was created as a result. This is typically the response sent after POST requests, or some PUT requests.

**202 Accepted**

The request has been received but not yet acted upon. It is noncommittal, since there is no way in HTTP to later send an asynchronous response indicating the outcome of the request. It is intended for cases where another process or server handles the request, or for batch processing.

**203 Non-Authoritative Information**

This response code means the returned metadata is not exactly the same as is available from the origin server, but is collected from a local or a third-party copy. This is mostly used for mirrors or backups of another resource. Except for that specific case, the 200 OK response is preferred to this status.

**204 No Content**

There is no content to send for this request, but the headers are useful. The user agent may update its cached headers for this resource with the new ones.

**205 Reset Content**

Tells the user agent to reset the document which sent this request.

**206 Partial Content**

This response code is used in response to a range request when the client has requested a part or parts of a resource.

**Redirection messages**

**300 Multiple Choices**

In agent-driven content negotiation, the request has more than one possible response and the user agent or user should choose one of them. There is no standardized way for clients to automatically choose one of the responses, so this is rarely used.

**301 Moved Permanently**

The URL of the requested resource has been changed permanently. The new URL is given in the response.

**302 Found**

This response code means that the URI of requested resource has been changed temporarily. Further changes in the URI might be made in the future, so the same URI should be used by the client in future requests.

**303 See Other**

The server sent this response to direct the client to get the requested resource at another URI with a GET request.

**304 Not Modified**

This is used for caching purposes. It tells the client that the response has not been modified, so the client can continue to use the same cached version of the response.

**Client error responses**

**400 Bad Request**

The server cannot or will not process the request due to something that is perceived to be a client error (e.g., malformed request syntax, invalid request message framing, or deceptive request routing).

**401 Unauthorized**

Although the HTTP standard specifies "unauthorized", semantically this response means "unauthenticated". That is, the client must authenticate itself to get the requested response.

**402 Payment Required**

The initial purpose of this code was for digital payment systems, however this status code is rarely used and no standard convention exists.

**403 Forbidden**

The client does not have access rights to the content; that is, it is unauthorized, so the server is refusing to give the requested resource. Unlike 401 Unauthorized, the client's identity is known to the server.

**404 Not Found**

The server cannot find the requested resource. In the browser, this means the URL is not recognized. In an API, this can also mean that the endpoint is valid but the resource itself does not exist. Servers may also send this response instead of 403 Forbidden to hide the existence of a resource from an unauthorized client. This response code is probably the most well known due to its frequent occurrence on the web.

**405 Method Not Allowed**

The request method is known by the server but is not supported by the target resource. For example, an API may not allow DELETE on a resource, or the TRACE method entirely.

**406 Not Acceptable**

This response is sent when the web server, after performing server-driven content negotiation, doesn't find any content that conforms to the criteria given by the user agent.

**408 Request Timeout**

This response is sent on an idle connection by some servers, even without any previous request by the client. It means that the server would like to shut down this unused connection. This response is used much more since some browsers use HTTP pre-connection mechanisms to speed up browsing. Some servers may shut down a connection without sending this message.

**409 Conflict**

This response is sent when a request conflicts with the current state of the server. In WebDAV remote web authoring, 409 responses are errors sent to the client so that a user might be able to resolve a conflict and resubmit the request.

**410 Gone**

This response is sent when the requested content has been permanently deleted from server, with no forwarding address. Clients are expected to remove their caches and links to the resource. The HTTP specification intends this status code to be used for "limited-time, promotional services". APIs should not feel compelled to indicate resources that have been deleted with this status code.

**413 Content Too Large**

The request body is larger than limits defined by the server. The server might close the connection or return an Retry-After header field.

**414 URI Too Long**

The URI requested by the client is longer than the server is willing to interpret.

**415 Unsupported Media Type**

The media format of the requested data is not supported by the server, so the server is rejecting the request.

**417 Expectation Failed**

This response code means the expectation indicated by the Expect request header field cannot be met by the server.

**421 Misdirected Request**

The request was directed at a server that is not able to produce a response. This can be sent by a server that is not configured to produce responses for the combination of scheme and authority that are included in the request URI.

**426 Upgrade Required**

The server refuses to perform the request using the current protocol but might be willing to do so after the client upgrades to a different protocol. The server sends an Upgrade header in a 426 response to indicate the required protocol(s).

**428 Precondition Required**

The origin server requires the request to be conditional. This response is intended to prevent the 'lost update' problem, where a client GETs a resource's state, modifies it and PUTs it back to the server, when meanwhile a third party has modified the state on the server, leading to a conflict.

**429 Too Many Requests**

The user has sent too many requests in a given amount of time (rate limiting).

**431 Request Header Fields Too Large**

The server is unwilling to process the request because its header fields are too large. The request may be resubmitted after reducing the size of the request header fields.

**451 Unavailable For Legal Reasons**

The user agent requested a resource that cannot legally be provided, such as a web page censored by a government.

**Server error responses**

**500 Internal Server Error**

The server has encountered a situation it does not know how to handle. This error is generic, indicating that the server cannot find a more appropriate 5XX status code to respond with.

**501 Not Implemented**

The request method is not supported by the server and cannot be handled. The only methods that servers are required to support (and therefore that must not return this code) are GET and HEAD.

**502 Bad Gateway**

This error response means that the server, while working as a gateway to get a response needed to handle the request, got an invalid response.

**503 Service Unavailable**

The server is not ready to handle the request. Common causes are a server that is down for maintenance or that is overloaded. Note that together with this response, a user-friendly page explaining the problem should be sent. This response should be used for temporary conditions and the Retry-After HTTP header should, if possible, contain the estimated time before the recovery of the service. The webmaster must also take care about the caching-related headers that are sent along with this response, as these temporary condition responses should usually not be cached.

**504 Gateway Timeout**

This error response is given when the server is acting as a gateway and cannot get a response in time.

**508 Loop Detected (WebDAV)**

The server detected an infinite loop while processing the request.

**510 Not Extended**

The client request declares an HTTP Extension (RFC 2774) that should be used to process the request, but the extension is not supported.

**511 Network Authentication Required**

Indicates that the client needs to authenticate to gain network access.