HTTP Requests

HTTP works as a request - response protocol between a client and a server. So to complete this two way relationship the client side has to produce the requests to server to which the server is going to respond. However this request can be produced in a variety of forms that acts differently in the nature of what the client is requesting or what information client is ready to share with the server.

The requests are:-

1. **GET** :- It requests the representation of the specified resources. Using this method will only retrieve data. It can be cached, bookmarked and have length restrictions. It remains in browser history and should never be used while sharing sensitive information. Most commonly used http methods.
   Example-

2. **POST** :- Used to submit an entity to the specified resource, often causing change in state or side effects. It is in many ways just the opposite of get method. It is never cached, nor bookmarked, don't have length restrictions and don't come up in browsers history. Always trusted for sharing sensitive information like passwords and credentials.

3. **HEAD** :- Identical to GET but without the response body. Example- if get returns the list of users, the head will make the same request but without returning the list of users.

4. **PUT** :- Replaces all current representation of target resource with request payload. The difference in post and put is that calling the post multiple times will have repeated side effects of creating the same resource multiple times, whereas the put method no matter the number of times it is called it will always produce the same result.Helps in complete replacement of a resource.

5. **DELETE** :- It deletes the specified resources.

6. **OPTIONS** :- Describes the communication option for the target resource.
   OPTIONS /index.html HTTP/1.1
   OPTIONS * HTTP/1.1
   An asterisk (*) is used to refer to an entire server.

7. **PATCH** :- Used to apply partial modifications to a resource. Not idempotent.

8. **CONNECT** :- Used to start a two way communication with requested resource. Its opens a tunnel.
   For example, the CONNECT method can be used to access websites that use SSL (HTTPS). The client asks an HTTP Proxy server to tunnel the TCP connection to the desired destination.
   The server then proceeds to make the connection on behalf of the client. Once the connection has been established by the server, the Proxy server continues to proxy the TCP stream to and from the clients.

9. **TRACE** :- Performs a message loopback test along the path to the target resource, providing a debugging mechanism.