Task 1 - Http Session, Http Methods, Http Status codes

1. List a few differences between HTTP and HTTPS.

SR.No.	HTTP	HTTPS
1	HTTP stands for HyperText Transfer Protocol.	HTTPS for HyperText Transfer Protocol Secure.
2.	In HTTP, URL begins with “http://”.	In HTTPs, URL starts with “https://”.
3.	HTTP uses port number 80 for communication.	HTTPs uses 443 port number for communication.
4.	HTTP is considered to be unsecure.	HTTPs are considered secure.
5.	In HTTP, Encryption is absent.	Encryption is present in HTTPS.
6.	HTTP does not require any certificates.	HTTPS needs SSL Certificates.
7.	HTTP does not improve search ranking	HTTPS helps to improve search ranking
8.	HTTP is faster than HTTPS.	HTTPS is slower than HTTP.
9.	HTTP Should be avoided.	HTTPS Should be preferred.
10.	In HTTP Data is transfer in plaintext.	In HTTPS Data transfer in ciphertext.

Reference:

a. Difference between HTTP and HTTPS - GeeksforGeeks
b. HTTP vs HTTPS: Comparison, Pros and Cons, and More (hostinger.com)

2. List down various HTTP Methods with short description (GET, POST, PUT, DELETE)
    a. **GET**:
        i. The GET method is used to retrieve data from a server. It requests a representation of the specified resource and does not modify the resource on the server. It is primarily used to read data from a server.
        ii. The GET method requests a representation of the specified resource. Requests using GET should only retrieve data.
    b. **POST**:
        i. The POST method is used to send data to the server to create a new resource. It submits data to be processed to a specified resource, often resulting in a new record being created on the server.
        ii. The POST method submits an entity to the specified resource, often causing a change in state or side effects on the server.
    c. **PUT**:
        i. The PUT method is used to update a resource on the server. It sends data to replace or update the current representation of the resource at the given URL. If the resource does not exist, PUT may create it.
        ii. The PUT method replaces all current representations of the target resource with the request payload.
    d. **DELETE**:
        i. The DELETE method is used to request the removal of a resource from the server. It instructs the server to delete the resource at the specified URL.
        ii. The DELETE method deletes the specified resource.

References:

a. HTTP request methods - HTTP | MDN (mozilla.org)
b. HTTP Request Methods (w3schools.in)

3. List down the various status (response) codes of each series. (at least 4 from each)
    a. Information responses
        i. **100 Continue**
            i. This interim response indicates that the client should continue the request or ignore the response if the request is already finished.
        ii. **101 Switching Protocols**
            i. This code is sent in response to an Upgrade request header from the client and indicates the protocol the server is switching to.
        iii. **102 Processing (WebDAV)**
            i. This code indicates that the server has received and is processing the request, but no response is available yet.
        iv. **103 Early Hints Experimental**
            i. 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.
    b. Successful responses:
        i. **201 Created**
            i. 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.
        ii. **202 Accepted**
            i. The request has been received but not yet acted upon. It is noncommittal, since there is no way for 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.
        iii. **203 Non-Authoritative Information**
            i. This response code means the returned metadata is not exactly the same as is available from the original 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.

iv. **204 No Content**
    i. There is no content to send for this request, but the headers may be useful. The user agent may update its cached headers for this resource with the new ones.

c. Redirection messages:
    i. **300 Multiple Choices**
        i. The request has more than one possible response. The user agent or user should choose one of them. (There is no standardized way of choosing one of the responses, but HTML links to the possibilities are recommended so the user can pick.)
    ii. **301 Moved Permanently**
        i. The URL of the requested resource has been changed permanently. The new URL is given in the response.
    iii. **302 Found**
        i. 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. Therefore, this same URI should be used by the client in future requests.
    iv. **303 See Other**
        i. The server sent this response to direct the client to get the requested resource at another URI with a GET request.

d. Client error responses
    i. **400 Bad Request**
        i. 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).
    ii. **401 Unauthorized**
        i. Although the HTTP standard specifies "unauthorized", semantically this response means "unauthenticated". That is, the client must authenticate itself to get the requested response.
    iii. **403 Forbidden**
        i. The client does not have access rights to the content; that is, it is unauthorized, so the server is refusing to give the

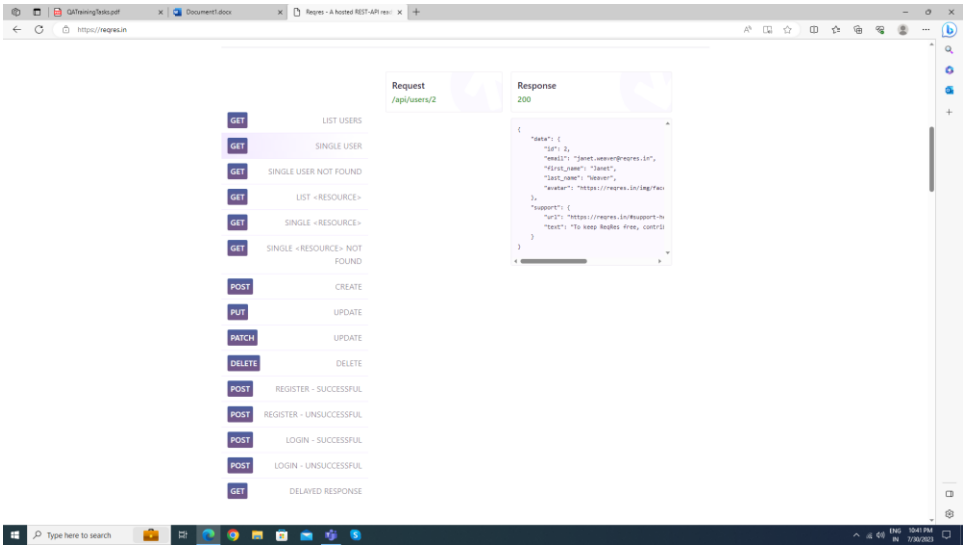
requested resource. Unlike 401 Unauthorized, the client's identity is known to the server.

iv. **404 Not Found**
    i. 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.

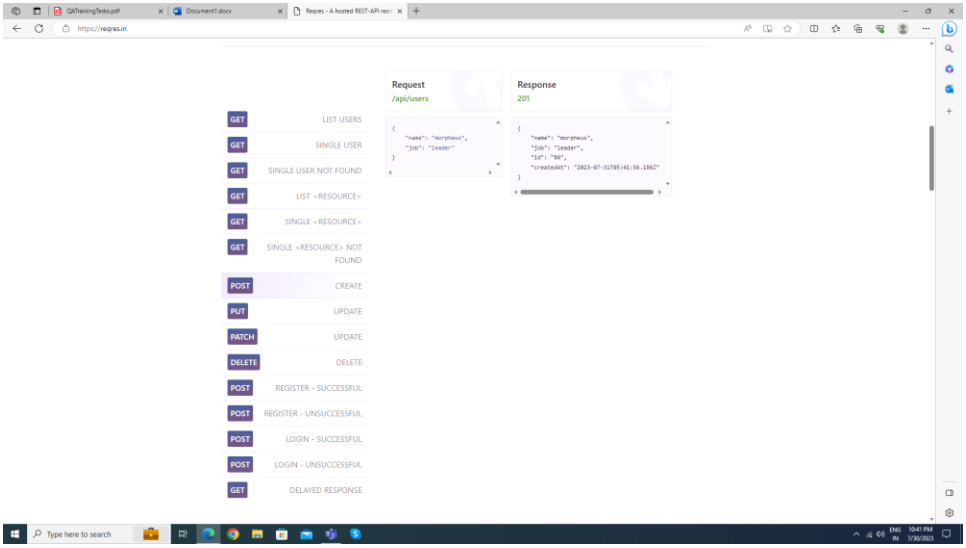
e. Server error responses
    i. **500 Internal Server Error**
        i. The server has encountered a situation it does not know how to handle.
    ii. **501 Not Implemented**
        i. The request method is not supported by the server and cannot be handled. The only methods that servers are required to support (and therefore that must not return this code) are GET and HEAD.
    iii. **502 Bad Gateway**
        i. 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.
    iv. **503 Service Unavailable**
        i. 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

References: HTTP response status codes - HTTP | MDN (mozilla.org)

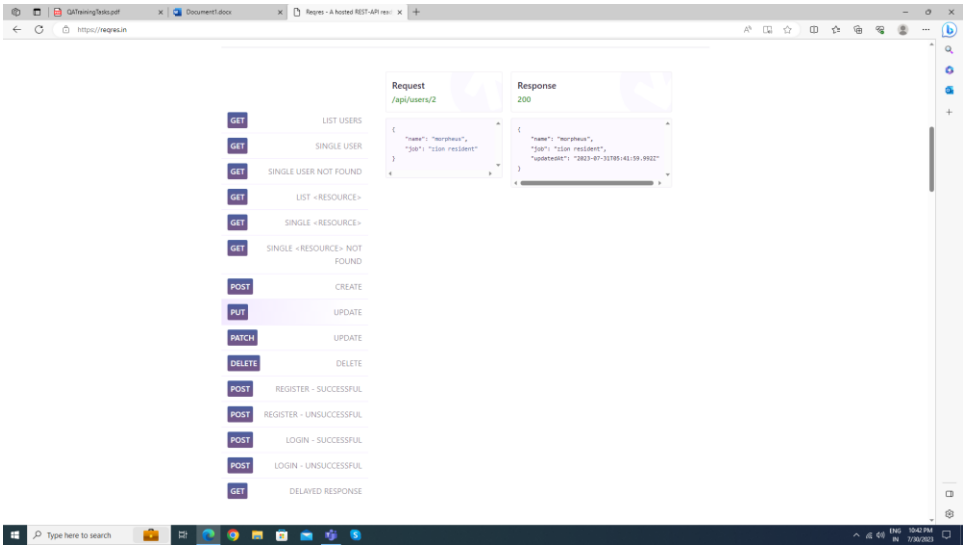
4. Perform GET, POST, PUT, PATCH and DELETE http methods from https://reqres.in/ and add screenshot.
    a. GET :



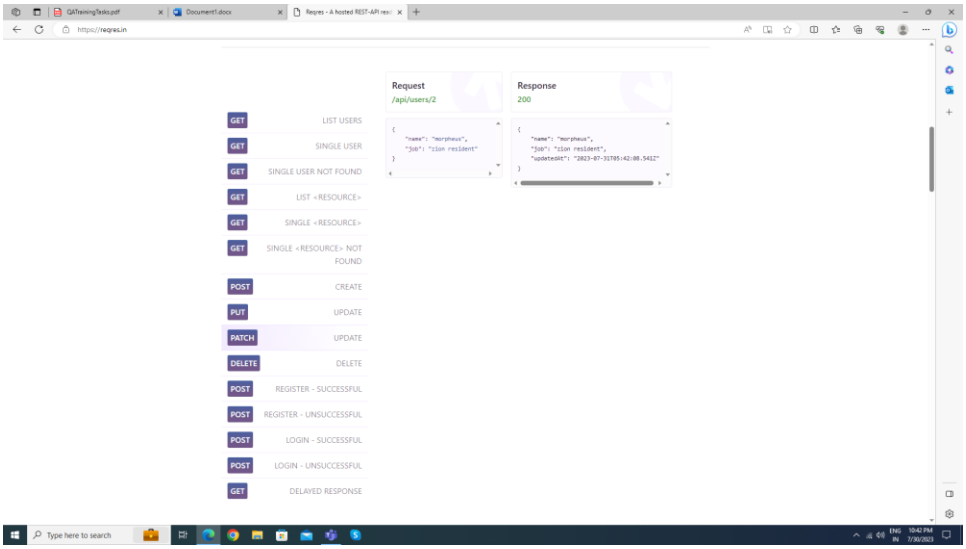
    b. POST:



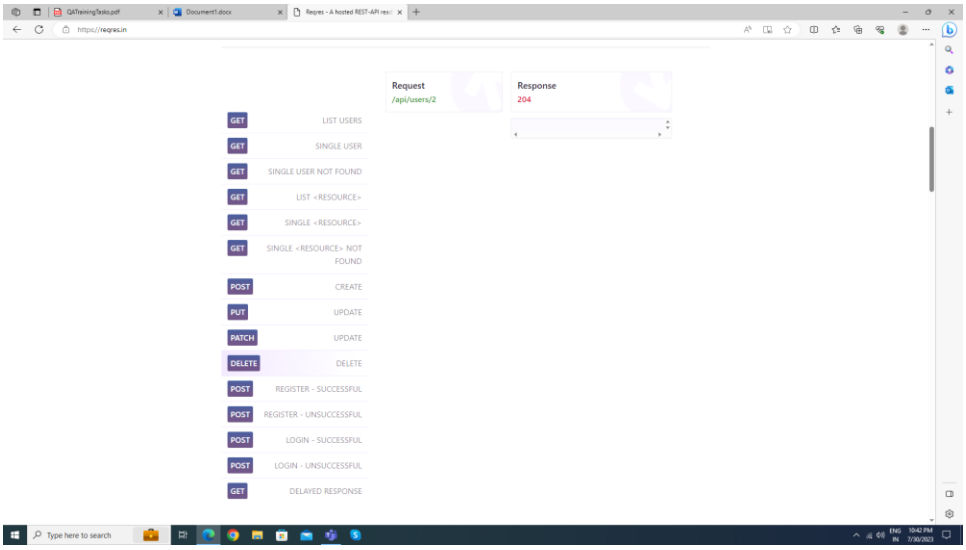
c. PUT::


Request
/api/users/2
Response
200
GET LIST USERS
GET SINGLE USER
GET SINGLE USER NOT FOUND
GET LIST <RESOURCE>
GET SINGLE <RESOURCE>
GET SINGLE <RESOURCE> NOT FOUND
POST CREATE
PUT UPDATE
PATCH UPDATE
DELETE DELETE
POST REGISTER - SUCCESSFUL
POST REGISTER - UNSUCCESSFUL
POST LOGIN - SUCCESSFUL
POST LOGIN - UNSUCCESSFUL
GET DELAYED RESPONSE

d. PATCH:


Request
/api/users/2
Response
200
GET LIST USERS
GET SINGLE USER
GET SINGLE USER NOT FOUND
GET LIST <RESOURCE>
GET SINGLE <RESOURCE>
GET SINGLE <RESOURCE> NOT FOUND
POST CREATE
PUT UPDATE
PATCH UPDATE
DELETE DELETE
POST REGISTER - SUCCESSFUL
POST REGISTER - UNSUCCESSFUL
POST LOGIN - SUCCESSFUL
POST LOGIN - UNSUCCESSFUL
GET DELAYED RESPONSE

e. DELETE:


Request
/api/users/2
Response
204
GET LIST USERS
GET SINGLE USER
GET SINGLE USER NOT FOUND
GET LIST <RESOURCE>
GET SINGLE <RESOURCE>
GET SINGLE <RESOURCE> NOT FOUND
POST CREATE
PUT UPDATE
PATCH UPDATE
DELETE DELETE
POST REGISTER - SUCCESSFUL
POST REGISTER - UNSUCCESSFUL
POST LOGIN - SUCCESSFUL
POST LOGIN - UNSUCCESSFUL
GET DELAYED RESPONSE