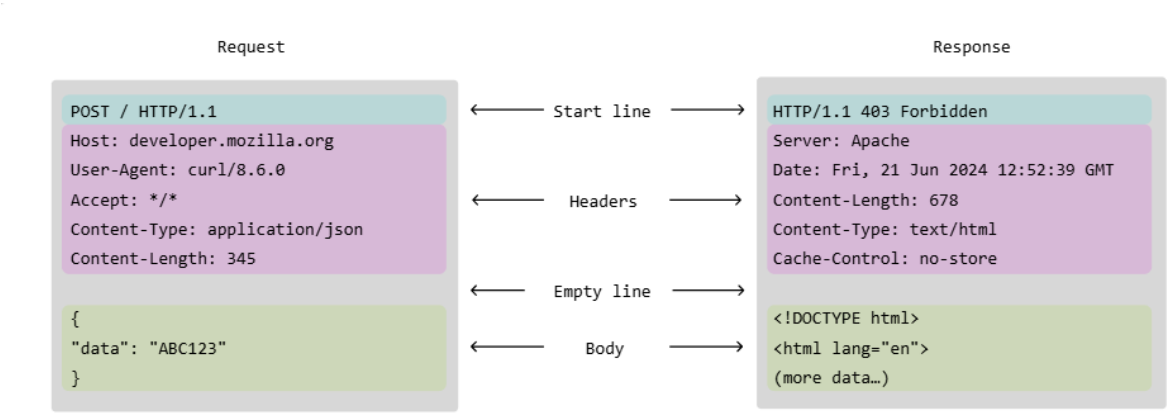
Project 1 RESTful API(17/02/2025)

Differences between HTTP and HTTPS:

Hypertext Transfer Protocol (HTTP) is a protocol or set of rules governing client-server communication. When you visit a website, your browser sends an HTTP request to the web server, which returns an HTTP response. The Web server and your browser exchange data in plain text format. In short, the HTTP protocol is the underlying technology that governs network communication. As its name suggests, the Secure Hypertext Transfer Protocol (HTTPS) is a more secure version or extension of the HTTP protocol. With HTTPS, the browser and server establish a secure, encrypted connection before transferring data. HTTPS websites must obtain an SSL/TLS certificate from an independent certification authority (CA). These websites share the certificate with the browser before exchanging data to establish trust.

The structure of an HTTP request and response:



Both requests and responses share a similar structure:

1. A start-line is a single line that describes the HTTP version along with the request method or the outcome of the request.
2. An optional set of HTTP headers containing metadata that describes the message. For example, a request for a resource might include the allowed formats of that resource, while the response might include headers to indicate the actual format returned.
3. An empty line indicating the metadata of the message is complete.
4. An optional body containing data associated with the message. This might be POST data to send to the server in a request, or some resource returned to the client in a response. Whether a message contains a body or not is determined by the start-line and HTTP headers.

The start-line in HTTP/1.x requests (POST /users HTTP/1.1 in the example above) is called a "request-line" and is made of three parts:

<method> <request-target> <protocol>

<method>:
The HTTP method (also known as an *HTTP verb*) is one of a set of defined words that describes the meaning of the request and the desired outcome. For example, GET indicates that the client would like to receive a resource in return, and POST means that the client is sending data to a server.

<request-target>:
The request target is usually an absolute or relative URL, and is characterized by the context of the request. The format of the request target depends on the HTTP method used and the request context. It is described in more detail in the request target section below.

<protocol>:
The *HTTP version*, which defines the structure of the remaining message, acting as an indicator of the expected version to use for the response. This is almost always HTTP/1.1, as HTTP/0.9 and HTTP/1.0 are obsolete. In HTTP/2 and above, the protocol version isn't included in messages since it is understood from the connection setup.

The start-line (HTTP/1.1 201 Created above) is called a "status line" in responses, and has three parts:

<protocol> <status-code> <status-text>

<protocol>:
The *HTTP version* of the remaining message

<status-code>:
A numeric status code that indicates whether the request succeeded or failed

<status-text>:
The status text is a brief, purely informational, textual description of the status code to help a human understand the HTTP message

Lists of common HTTP methods and status codes with their descriptions:

METHODS

GET : The GET method is used to retrieved information from the given server using a given URI. Requests using GET should only retrieve data and should have no other effect on the data.

GET /index.htm HTTP/1.1
User-Agent: Mozilla/4.0
Host: www.tutorialspoint.com
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive

POST : A POST request is used to send data to the server, for example, customer information, file upload, etc. using HTML forms.

POST /cgi-bin/process.cgi HTTP/1.1
User-Agent: Mozilla/4.0
Host: www.tutorialspoint.com
Content-Type: text/xml; charset=utf-8
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive

PUT : The PUT method is used to request the server to store the included entity-body at a location specified by the given URL.

PUT /index.htm HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT)
Host: www.tutorialspoint.com
Accept-Language: en-us
Connection: Keep-Alive
Content-type: text/html
Content-Length: 182

PATCH : PATCH is similar to PUT, but it's used for partial updates. Instead of replacing the entire resource, it only updates the fields you specify.

PATCH /file.txt HTTP/1.1
Host: www.example.com
Content-Type: application/example
If-Match: "e0023aa4e"
Content-Length: 100

DELETE : Removes all current representations of the target resource given by a URI.

DELETE /index.htm HTTP/1.1
User-Agent: Mozilla/4.0
Host: www.tutorialspoint.com
Accept-Language: en-us
Connection: Keep-Alive

STATUS CODES

200 OK
Standard response for successful HTTP requests. The actual response will depend on the request method used. In a GET request, the response will contain an entity corresponding to the requested resource. In a POST, the response will contain an entity describing or containing the result of the action.

403 Forbidden
The request contained valid data and was understood by the server, but the server is refusing action. This may be due to the user not having the necessary permissions for a resource or needing an account of some sort, or attempting a prohibited action. This code is also typically used if the request provided authentication by answering the WWW-Authenticate header field challenge, but the server did not accept that authentication. The request should not be repeated.

404 Not Found
The requested resource could not be found but may be available in the future. Subsequent requests by the client are permissible.

500 Internal Server Error
A generic error message, given when an unexpected condition was encountered and no more specific message is suitable.

502 Bad Gateway
The server was acting as a gateway or proxy and received an invalid response from the upstream server.