## **HTTP (Hypertext Transfer Protocol)**

**It** is the foundation of data communication on the World Wide Web. It's a protocol used for transmitting hypermedia documents, such as HTML.

Let's break down the key components you've mentioned:

## 1. HTTP Methods:

- GET: Requests data from a specified resource.

- POST: Submits data to be processed to a specified resource.

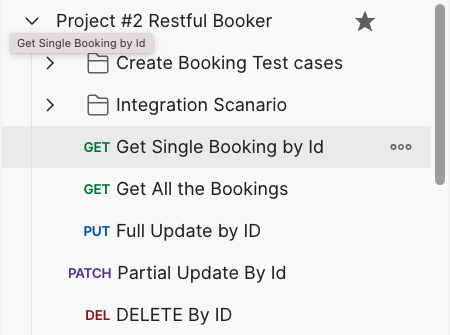
- PUT: Updates a specified resource.

- DELETE: Deletes the specified resource.

- HEAD: Similar to GET, but it retrieves only the headers (no body).

- OPTIONS: Describes the communication options for the target resource.

- PATCH: Partially modifies a resource.

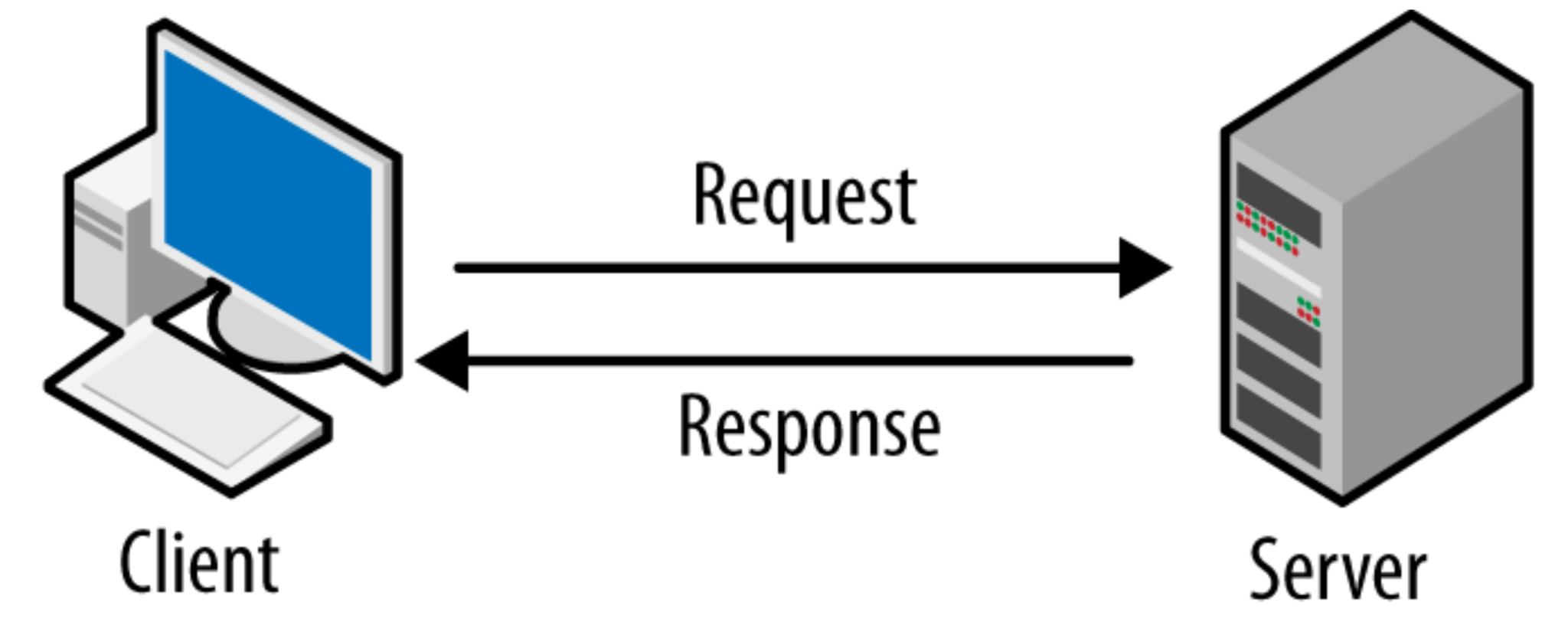
**Restful Booker CRUD Example - **

<https://api.postman.com/collections/611814-e28692c4-a8b5-4de5-bbf2-85471163e0db?access_key=PMAT-01HFDXYJ9V97RT0EY6HRJ0JRB7>

## 2. Client-Server Model:

- In this model, a client (such as a web browser) sends an HTTP request to the server; then the server returns the response.

- The client initiates the request and waits for the server to respond. The server processes the request and returns a response.



## 3. Authentication:

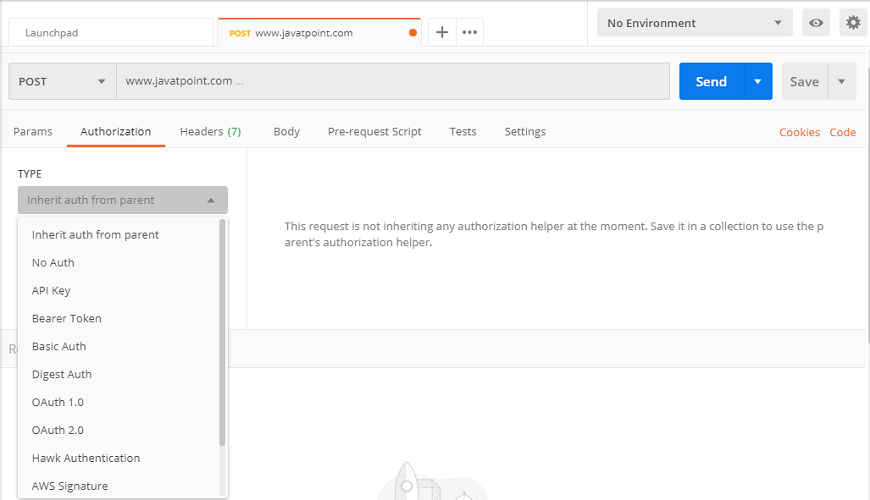
- HTTP provides several authentication mechanisms like Basic, Digest, and Bearer (Token) to control access to resources.

- Authentication information is typically sent in the HTTP header.

[HTTP Methods. Authentication & Cookie basics](https://youtu.be/9CHkyni6_sQ)

Read more -

<https://scrolltest.com/2017/06/15/http-methods-authentication-cookie-basics/>



## 4. Cookie:

- Cookies are small pieces of data stored on the client side and sent to the server with HTTP requests.

- They are used to remember stateful information (like items in a shopping cart) or to record browsing activity.

- Key and value

**EditThisCookie** https://chrome.google.com/webstore/detail/editthiscookie/fngmhnnpilhplaeedifhccceomclgfbg

## 5. URL (Uniform Resource Locator):

- A URL is a reference to a web resource. It specifies the location of a resource on a computer network and a mechanism for retrieving it.

- A typical URL includes the protocol (HTTP or HTTPS), domain name, optionally the port, the path of the resource, and query strings.

A URL, or Uniform Resource Locator, is a reference or address used to access resources on the internet. It specifies the location of a resource on a computer network and a mechanism for retrieving it. A URL is the most common type of Uniform Resource Identifier (URI), although many people use the terms URL and URI interchangeably.

Here's a breakdown of the typical components of a URL:

1. **Scheme**: This part of the URL indicates the protocol used to access the resource. Common protocols include HTTP (Hypertext Transfer Protocol), HTTPS (HTTP Secure), FTP (File Transfer Protocol), and mailto (for email addresses).

2. **Host:** Also known as the domain name, this part specifies which server on the Internet is holding the resource. For example, in `www.example.com`, "example.com" is the domain name.

3. **Port (optional):** This is a numerical value used to specify a specific service endpoint at the host. It is separated from the host by a colon. For instance, `http://example.com:80` where `80` is the port for HTTP. If not specified, the default port for the protocol is used (e.g., 80 for HTTP, 443 for HTTPS).

4. **Path:** This part specifies the exact location of the resource within the host. For example, in `http://www.example.com/index.html`, "/index.html" is the path to a specific file.

5. **Query (optional)**: A query string starts with a question mark (`?`) and is used to provide additional information to the server. It usually consists of key-value pairs separated by ampersands (`&`). For example, in `http://www.example.com/search?q=keyword`, the query is `q=keyword`.

6. **Fragment (optional):** A fragment identifier, introduced by a hash mark (`#`), is used to point to a specific part of a document. For example, in `http://www.example.com/index.html#section2`, "#section2" refers to a specific section of the "index.html" page.

A typical URL looks like this: `http://www.example.com:80/index.html?q=search#section2`, where:

- `http` is the scheme.

- `www.example.com` is the host.

- `80` is the port.

- `/index.html` is the path.

- `q=search` is the query.

- `#section2` is the fragment.

## 6. Headers:

- HTTP headers let the client and server pass additional information with an HTTP request or response.

- Headers can include metadata such as content type, content length, server type, set cookies, and much more.

## 7. Status Codes:

- Status codes are issued by a server in response to a client's request made to the server.

- They include:

- `1xx` (Informational): Request received, continuing process.

- `2xx` (Successful): The action was successfully received, understood, and accepted.

- `3xx` (Redirection): Further action needs to be taken in order to complete the request.

- `4xx` (Client Error): The request contains bad syntax or cannot be fulfilled.

- `5xx` (Server Error): The server failed to fulfill an apparently valid request.

Each of these components plays a vital role in the communication between a web browser (or any HTTP client) and a web server, ensuring the seamless operation of the web as we know it.

## 8. Request and Response Structure:

Request: An HTTP request contains a method (like GET, POST, etc.), a URL, HTTP version, headers, and sometimes a body (for POST, PUT, PATCH).

Response: An HTTP response includes a status code, HTTP version, headers, and a body (which contains the requested data or the result of the operation).

## 9. Content Types:

The Content-Type header in HTTP is used to indicate the media type of the resource.

In API testing, common content types are application/json for JSON and application/xml for XML.

## 10. Authentication and Authorization:

Beyond basic authentication, APIs often use tokens (like JWT - JSON Web Tokens) or OAuth for secure access.

Testing should include scenarios where authentication is required, testing with valid/invalid tokens, and ensuring proper access control.

## 11. Query Parameters and Path Variables:

Query Parameters: Used to filter or customize the response. For example, /api/users?age=25 might only return users who are 25 years old.

Path Variables: Part of the URL's path, typically used to identify a specific resource. For example, /api/users/123 might refer to the user with ID 123.