**HTTP Protocol**

Overview:

* HTTP (Hypertext Transfer Protocol) is a protocol used for transmitting hypertext (e.g., HTML) over the internet.
* It operates on a client-server model, where the client (usually a web browser) makes requests to a server, which then responds with the requested resources or error messages.
* **Stateless Protocol:** Each HTTP request is independent of other; the server does not retain information from previous requests.

Request/Response Model:

* **Client Request:** The client sends an HTTP request to the server.
* **Server Response:** The server processes the request and sends back an HTTP response.

**HTTP Server**

Definition:

* An HTTP server is software that handles HTTP requests from clients and serves back responses. It processes incoming requests, executes the necessary logic (e.g., accessing a database, generating HTML), and returns the appropriate response.

Examples:

* Apache HTTP Server, Nginx, Microsoft IIS, Kestrel (used with ASP.NET Core)

Kestrel:

* Kestrel is a cross-platform web server included with ASP.NET Core.
* It is lightweight, high-performance, and suitable for running both internal and public-facing web applications.

**Request and Response Flow with Kestrel**

1. **Client Sends Request:**
2. The client (e.g., web browser) sends an HTTP request to the server.
3. **Kestrel Receives Request:**
4. Kestrel receives the request and passes it through the ASP.NET Core middleware pipeline.
5. **Request Processing:**
6. Middleware components process the request and eventually pass it to the application’s request handling logic.
7. **Generate Response:**
8. The application generates an HTTP response and sends it back through the middleware pipeline.
9. **Kestrel Sends Response:**
10. Kestrel sends the HTTP response back to the client.

How Browsers Use HTTP

* Browsers use HTTP to request resources such as HTML documents, images, CSS files, and JavaScript files from servers.
* When a user enters a URL or clicks a link, the browser sends an HTTP request to the server, which then responds with the requested resource.

Observing HTTP Requests and Responses in Chrome Dev Tools

* **Open Chrome Dev Tools:**
  + Press F12 or Ctrl + Shift + I (or Cmd+Option+I on Mac) to open Chrome Dev Tools.
* **Navigate to the Network Tab:**
  + Click on the Network tab to view HTTP requests and responses.
* **Inspect a Request:**
  + Click on any request in the list to see detailed information:
    - **Headers:** View request and response headers.
    - **Preview/Response:** View the response body.
    - **Timing:** See the timing details of the request.

HTTP Response Message Format

**Response Message Format:**

* **Start Line:** Contains the HTTP version, status code, and status message.
* **Headers:** Key-value pairs providing information about the response.
* **Body:** Optional, contains the actual data (e.g., HTML, JSON).

**Example:**

HTTP/1.1 200 OK

Content-Type: text/html

Content-Length: 137

<html>

<body>

<h1>Hello, World!</h1>

</body>

</html>

**Commonly Used Response Headers:**

* **Content-Type:** Specifies the media type of the resource.
* **Content-Length:** The size of the response body in bytes.
* **Server:** Provides information about the server handling the request.
* **Set-Cookie:** Sets cookie to be stored by the client.
* **Cache-Control:** Directives for caching mechanisms in both requests and responses.

Default Response Headers in Kestrel

* **Content-Type:** Typically defaults to text/html or application/json depending on the content being served.
* **Server:** Indicates the server software (e.g., Kestrel)
* **Date:** The date and time when the response was generated.

HTTP Status Codes

**Overview:**

* Status codes are issued by the server in response to the client’s request to indicate the result of the request.
* Categories include:
  + **1xx Informational:** Request received, continuing process.
  + **2xx Success:** The request 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 fulfil an apparently valid request.

**Common Status Codes:**

* **200 OK:** The request succeeded.
* **201 Created:** The request succeeded and a new resource was created.
* **204 No Content:** The server successfully processed the request, but is not returning any content.
* **400 Bad Request:** The server could not understand the request due to invalid syntax.
* **401 Unauthorized:** Authentication is required.
* **403 Forbidden:** The client does not have access rights to the content.
* **404 Not Found:** The server cannot find the request resource.
* **500 Internal Server Error:** Ther server encountered an unexpected condition.
* **502 Bad Gateway:** The server was acting as a gateway or proxy and received an invalid response from the upstream server.
* **503 Service Unavailable:** The server is not ready to handle the request.