**Part One: Solidify Terminology**

In your own terms, define the following terms:

* What is HTTP?

HTTP is an acronym for Hypertext Transfer Protocol. This is the protocol for accessing and serving web-based files and exchanging data on a network. This is the application layer protocol for transmitting hypermedia documents, such as HTML pages, over a network.

* What is a URL?

URL is an acronym for Uniform Resource Locator. A URL is the ‘address’ of the computer where files are served from, and the name of the files and their location on the server. A URL is the address of a website. It is a text string used by web browsers, email applications, and other web applications to identify the location of resources (files, hypertext documents, etc.) on the web.

* What is DNS?

DNS is a Domain Name System that is used to provide a textual reference or name for a web server IP address. In a DNS system, when a user enters the name of a website into a browser or other web application, the local DNS list is searched first, to locate the reference in the local DNS list. If it does not find it locally, it will next look for the name / IP address reference on a DNS list on a local router, which may search for the reference on a DNS server. When the referenced website Domain Name is located, the IP address of the server is returned.

* What is a query string?

A Query string is a string of characters that provides parameters in a web search and is appended to the end of the URL to provide specific parameters as to what specific resource is being requested. The query string allows you to pass key-value pairs into the URL, in the format.  ?key1=value1&key2=value2

* What are two HTTP verbs and how are they different?

GET is used to retrieve data and/or resources from a host computer or server. It makes no changes to the server. POST is used to send data to a computer (typically to a database). A POST method will alter the data on the receiving computer.

* What is an HTTP request?

In an HTTP request, the client computer send a request to access content on a server or host computer, then waiting until it receives a response.

* What is an HTTP response?

In an HTTP response, the server or host computer returns a response to the client request to access resources, such as a web page, on the server. The server acknowledges the request and either returns the required requested resources to the client computer, or it returns an error message if it cannot locate the requested resources or if the client lacks permissions to access the server or the requested resource(s).

* What is an HTTP header? Give a couple examples of request and response headers you have seen.

Headers provide additional information about the request or the response.

Examples:

Request headers: Host, User-Agent, Accept, Cookie

Response headers: Content-Type, Last-Modified, Set-Cookie

* What are the processes that happen when you type “<http://somesite.com/some/page.html>” into a browser?

1. Your browser “resolves” the name into an IP address using DNS
2. Your browser makes a request to that IP address, including headers (info about browser, cookies data, language, type)
3. The server sends a response with a status code, typically returning an HTML document.
4. The browser creates a DOM from that HTML, and finds any other resources needed (images, CSS, JavaScript, etc)
5. The browser makes separate HTTP requests for each needed resources and receives response from the server(s) for each.

**Part Two: Practice Tools**

1. Using ***curl***, make a ***GET*** request to the *icanhazdadjoke.com* API to find all jokes involving the word “pirate”

karenrusseth@Karens-MacBook-Pro ~ % curl https://icanhazdadjoke.com/

What does a pirate pay for his corn? A buccaneer!

1. Use ***dig*** to find what the IP address is for *icanhazdadjoke.com*

A picture containing text, screenshot, font, document

Description automatically generated

1. Make a simple web page and serve it using ***python3 -m http.server***. Visit the page in a browser

A screenshot of a computer program

Description automatically generated with medium confidence

A screenshot of a cat

Description automatically generated with medium confidence

**Part Three: Explore Dev Tools**

Build a very simple HTML form that uses the GET method (it can use the same page URL for the action) when the form is submitted.

Add a field or two to the form and, after submitting it, explore in Chrome Developer tools how you can view the request and response headers.

A screenshot of a computer

Description automatically generated with medium confidence

Edit the page to change the form type to POST, refresh in the browser and re-

submit. Do you still see the field in the query string? Explore in Chrome how you can view the request and response headers, as well as the form data.

A screenshot of a computer

Description automatically generated with medium confidence

**Part Four: Explore the URL API**

At times, it’s useful for your JavaScript to look at the URL of the browser window and change how the script works depending on parts of that (particularly the query string).

[Read about the URL API](https://developer.mozilla.org/en-US/docs/Web/API/URL)

Try some of the code examples in the Chrome Console so that you can get comfortable with the basic methods and properties for instances of the URL class.