Make sure to convert to PDF - 1-3 pages:

How the Web Functions

A web browser is an application. The application layer provides the protocols that an application needs in order to communicate with another computer. These protocols are destined for a specific port on another computer in the network. For example, the HTTP protocol will arrive on port 80, so that the server knows that the client is asking for a webpage. When you click on a link, a request is sent from the client (the browser) to a (web) server. In order for the request to be delivered to the correct server, the IP address of the web server must be identified. The link contains a domain name which is a readable version of the IP address (the IP address is a nine digit number that is an unique identifier of a computer on the internet and will ensure that the packets of information being sent from the client will arrive to the correct server). Before a HTTP request is sent, the client must find the IP address that is associated with a domain name. This is referred to the Domain Name System (DNS). First the computer will check its local DNS cache, which is a local storage of information that your computer has previously requested, if the DNS query has not yet expired from the cache, then its IP address is returned. If it did not find the IP address in in the local DNS cache or the DNS query expired, a new DNS query is sent to the DNS server. The DNS server, also referred to as a resolver, has a cache, its own local store, of common requests. If the IP address is found here, it will return it to the client. If it is not found, the DNS server will send a query to a name server. The name server will look at the top level domain, in this case it is .com, which will then send the query to a top level domain (TLD) name server. The TLD name server will forward that query to an authoritative name server which will provide the IP address for a specific domain. This IP address is returned to the DNS server and client and is also stored in the cache, the client's local cache and the DNS server's cache. The stored address is assigned a time to live until it expires. A connection between the client and web server can now be established. Since this is an HTTPS, a secure connection is first established. HTTPS is an extension of HTTP and the 'S' represents secure. HTTP sends requests in plain text, which means that anyone who can access the communication between two computers can read what data is going in between these two computers. The secure part of the HTTPS involves encrypting data, using an encryption protocol called Secure Socket Layer (SSL). A secure connection is established on a specific port on the server (port 443). A process called public key infrastructure (PKI) is used. This is where both a symmetric (the same key used for encryption is also used for decryption) and asymmetric (asymmetric cryptography - a private key and public key are constructed at the same time. When the public key is used for encryption, the private key is the only one that can decrypt it) cryptography are used. The client sends a hello message to the server. This hello message contains information on the version of SSL (TSL 1.0 is also referred to as SSL 3.1 because SSL is the predecessor of TSL(transport layer security)) (aka cipher: information such as encryption algorithm, hashing algorithm). Based on what the client is capable of doing, the server selects a key exchange method, cipher method that it is also able to do. Both the server and client come to an agreement on how data should be encrypted while data is being shared between the client and server. Next the server will send its SSL (secure socket layer) certificate to the client. This certificate verifies that the the domain name is actually owned by the owner as well as sharing its public key. The server retains its private key. The client checks the validity of

the certificate using CAs (certificate authorities). After verification, the client generates a key, called a session key, to be used in a symmetric cryptography, or session. This key is encrypted using the public key from the server. This now encrypted session key is sent to the server. The server's private key will decrypt the encrypted session key. The server can now decrypt data sent by the client. So now HTTP requests from the client are encrypted by the session key and the server will decrypt and then encrypt using the same session key back to the client. With this encryption, the HTTP information in the header is not in plain text format or is not easily readable. Integral information such as what method was requested, parameters etc are encrypted. When a web server receives an HTTP request, it will determine if the request requires a dynamic page or a static page. An HTTP request, based on Application Protocol, is sent from the browser to the web server. This request contains the URL, methods, and any other extra information (a request is made up of a header and it may have a POST body). In this case our method will be a GET request (get a file from the web server). The web server will decide if the request is looking for a static webpage or a dynamic web page before sending an HTTP response back to the browser. Static content is content that does not change or need any kind of modification. A user's preferences or input have no effect on static content. Some examples are CSS file, JS files, images, videos. This differs from dynamic content, where a user's input or preferences will result in content needing to be processed and modified before it is return back to the client. If a static page is requested, the web server can send this to the client. If a dynamic page is requested, this request is passed on to the application server which holds a program that will help complete this request. It interacts with a database to generate an html file that is sent back to the web server which then sends it back to the browser. This response can have different statuses, some examples are: that it successfully found (200 OK) the web page or a (400-not found) if the file was not found, or forbidden access(500-error)if due to a server error, (300- you were redirected). The process of making a request and receiving a response is called a HTTP transaction.

When the HTML document is returned to the client from the web server, the next step is to parse this document. The browser is an application or piece of software that consists of a number of different components. One of these parts is called a rendering engine, there are different rendering engines (like Webkit for Safari, and Blink for Chrome) based on whichever browser is used. The responsibility of this engine is to display the webpage. In order to display the web page, the page must be parsed. These rendering engines have a parser generator (such as Bison, Flex…) The result of parsing is a tree. Parsing is made up of two components, a parser and a lexer. The lexer will create tokens based on the vocabulary of the language (for example, the h1 tag is a vocab of the HTML language). The parser will check that this vocabulary follows the syntax or grammar rules of the language (for the same example, that the h1 tag has an opening and closing tag) constructs a tree from the tokens as long as the tokens follow the syntax rule else a syntax error is returned. The tree that is constructed from parsing an HTML document is called a document object model (DOM). The nodes of this tree are the tags found in the HTML document, these nodes can have children (other nodes) or lead to values or comments (which would be identified as the leaves). The CSS parser will create a CSSOM (CSS object model). Depending on which browser is used, the combined DOM and CSSOM can be called a render tree or frame tree (dependent on which rendering engine is

used). The render tree contains nodes that will be displayed or rendered in the browser, so for example, the head node from the DOM tree is not included in the render tree. JS is then used to manipulate the render tree by accessing the property of a node. This is what makes a webpage interactive. For example: document.getElementsByTagName(String) will access the DOM and returns a collection of all the elements with the specified String tag name. This collection can then be modified by using the style property to change the color or other CSS property. Then the layout is specified, using information in head of the html document, specified by the meta tag with a name and value of viewport. The width=device-width will specify the width of the page on a device. This width=device-width sets the context for the CSS formatting (how large an area needs to be and position for a content with respect to the width of the device). These areas are then painted, or pixels fill in these areas so that the content can now be viewed in the browser).
HTML/CSS using the render tree.
Main function of server side code is to tailor to a user's experience on the web. Based on a user's usage of the webpage and preferences, a web page becomes customized to the needs of a user. Some examples include: based on information collected on the user, a website can tailor its responses to the user (like on Amazon, suggestions are provided based on previous searches or purchases). Another example is sending alerts that are specific for a user (like notifying that a user has a new message in his or her Facebook messages.
Client side code is executed by the browser (instead of requesting from the server, it performs an action on its own). It plays an important role in form validation, either with HTML tags or by JavaScript. It plays an important role in making a page dynamic (through JavaScript) aside from sending requests to a server.
Anything that we request is an asset. Another case, is within the HTML document, elements containing the link or script tag, and anything that is external to the HTML document are considered assets. So depending on the number of external links, files, that a web page needs is proportional to the number of client-side assets.
server instance - one instance of an application server is created when installation occurs. Can have as many instances that is desired. Purpose is for isolation of an instance and rolling upgrades (an instance is taken from a cluster of instances, it is upgraded, and then returned to the cluster. This helps maintain operation time while upgrading occurs).
Runtime: the program(the instructions within the program) is executed. The loader of an operating system will first load a program into memory by going through the machine code of a program (for example- when writing source code in a high level language - like Java - a compiler is required in order to turn that high level language into machine code - binary values so that the program is understood by the computer). The loader also links this code to libraries that are needed by the program. Once in memory, the lines of the program are executed (control passes from the operating system to the code (called entry point). There is at least one instance of the database because during installation (a default instance) is created. Can go through the installation again to create another instance. (example: SQL Server).

https://datageek.blog/2013/02/04/db2-basics-what-is-an-instance/
https://dba.stackexchange.com/questions/34580/why-would-one-need-multiple-instances-of-same-oracle-database
https://www.mssqltips.com/sqlservertip/1048/how-and-why-would-i-use-multiple-instances-of-sql-server/
https://docs.oracle.com/cd/E11882_01/server.112/e40540/startup.htm#CNCPT955
https://docs.oracle.com/cd/B19306_01/server.102/b14220/dist_pro.htm
https://docs.oracle.com/cd/E11882_01/network.112/e41945/concepts.htm#NETAG177
https://www.theserverside.com/feature/Understanding-How-the-Application-Servers-Web-Container-Works

Pages:
https://www.upwork.com/hiring/development/a-guide-to-server-technology/
https://www.killersites.com/articles/articles_databaseDrivenSites.htm
https://docs.oracle.com/cd/B10501_01/network.920/a96580/concepts.htm

https://www.youtube.com/watch?v=33VYnE7Bzpk
https://robertheaton.com/2014/03/27/how-does-https-actually-work/
https://link-springer-com.ezproxy.rowan.edu/content/pdf/10.1007%2F978-1-4842-2499-1.pdf
https://www.youtube.com/channel/UCF-EPJcRpVWGzr-MwITe8AA/videos
https://developer.mozilla.org/en-US/docs/Learn/Server-side/First_steps/Client-Server_overview
https://softwareengineering.stackexchange.com/questions/171203/what-are-the-differences-between-server-side-and-client-side-programming
https://www.html5rocks.com/en/tutorials/internals/howbrowserswork/#Resources
https://www.quora.com/How-does-HTTP-work
https://eloquentjavascript.net/14_dom.html
https://web.stanford.edu/class/msande91si/www-spr04/readings/week1/InternetWhitepaper.htm
https://developers.google.com/web/fundamentals/performance/critical-rendering-path/render-blocking-css
https://developers.google.com/web/fundamentals/performance/critical-rendering-path/render-tree-construction

https://www.quora.com/In-reference-to-HTML-JavaScript-what-is-an-‘asset’
https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Client-side_web_APIs/Client-side_storage

https://www.maxcdn.com/one/visual-glossary/static-content/

https://www.maxcdn.com/one/visual-glossary/static-content/

***might be useful for dynamic websites and instances?
http://eds.a.ebscohost.com/ehost/ebookviewer/ebook/bmxlYmtfXzc4MTY0N19fQU41?sid=0c3ec5c3-d54f-4a57-8c65-d3fbc5461d30@sessionmgr4010&vid=0&format=EB&rid=1

Application servers use business logic to deliver dynamic content. Business logic is used between and application and a database.
https://stackoverflow.com/questions/34681936/what-is-httpd-exaclty

https://getlevelten.com/wiki/client-side-programming
https://www.linuxtopia.org/online_books/programming_books/thinking_in_java/TIJ303_019.htm
http://www.webbasedprogramming.com/CGI-Programming-Unleashed/ch24.htm#ComparisonofCGIandJavaJavaScript
Form validation: https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/Form_validation
Client side navigation:
https://www.codeproject.com/Articles/334867/Understanding-Page-Navigation-Techniques-in-ASP-NE

https://itknowledgeexchange.techtarget.com/sql-server/single-instance-vs-multiple-instances/