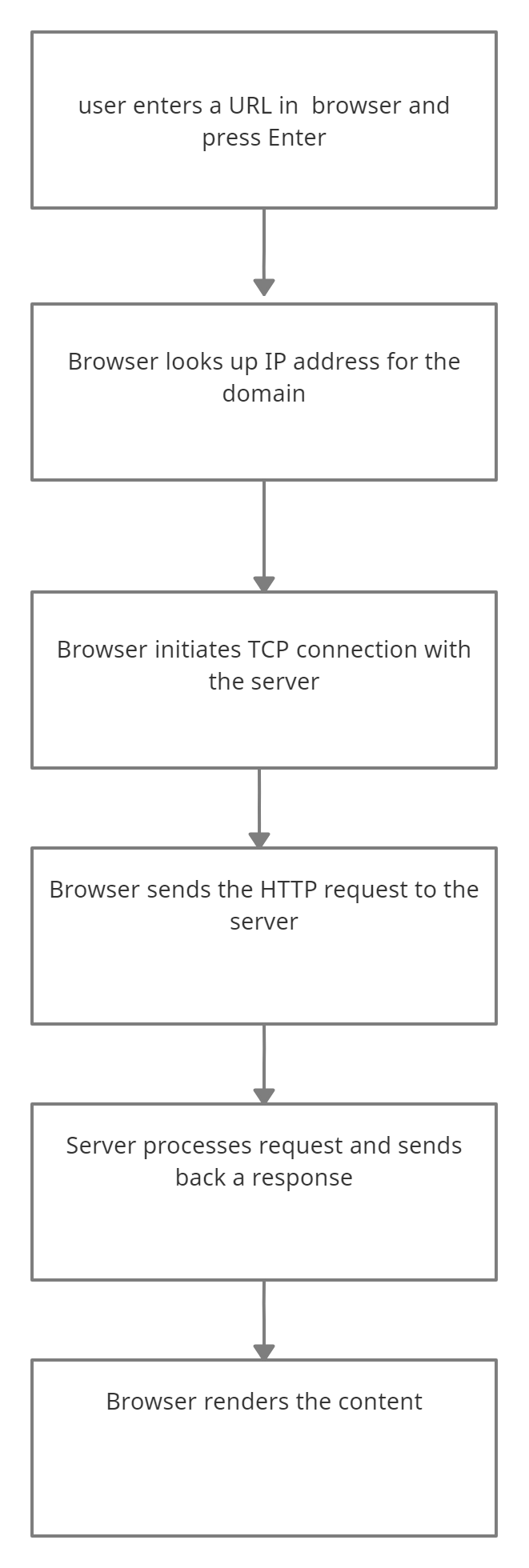
***How Browser works?***



* Websites are collections of files, often HTML, CSS, JavaScript, and images, that tell the browser how to display the site, images, and data. They need to be accessible to anyone from anywhere at any time, so hosting them on your computer at home isn’t be scalable or reliable. A powerful external computer connected to the Internet, called a server, stores these files. Browser has to figure out which server on the Internet is hosting the site. It does this by looking up the domain to find the address.
* Each device on the Internet — servers, cell phones, your smart refrigerator — all have a unique address called an IP address. An IP address contains four numbered parts: 203.0.113.0
* The domain name system, or DNS, is like the Contacts app on our phone. DNS helps our browser (and us) find servers on the Internet. We can do a DNS lookup to find the IP address of the server based on the domain name,
* https://www.google.com/images
* https:// is the scheme. HTTPS stands for Hypertext Transfer Protocol Secure. This scheme tells the browser to make a connection to the server using Transport Layer Security, or TLS. TLS is an encryption protocol to secure communications over the Internet.
* [www.google.com](http://www.google.com) is the domain name of the site. It is the memorable address and points to a specific server’s IP address
* Sometimes there is an additional path to the resource in the URL. For example, for this URL, <https://www.google.com/images> the path on the server to the requested resource images
* When we typed this URL into your browser, images is the name of the resource on the website you want to view.
* After we’ve typed the URL into your browser and pressed enter, the browser needs to figure out which server on the Internet to connect to. To do that, it needs to look up the IP address of the server hosting the website using the domain you typed in. It does this using a DNS lookup.
* Because DNS is complex and has to be blazingly fast, DNS data is cached at different layers between your browser and at various places across the Internet. The browser checks its own cache, the operating system cache, a local network cache at your router, and a DNS server cache on your corporate network or at your internet service provider (ISP). If the browser cannot find the IP address at any of those cache layers, the DNS server on your corporate network or at your ISP does a recursive DNS lookup. A recursive DNS lookup asks multiple DNS servers around the Internet, which in turn ask more DNS servers for the DNS record until it is found.
* Once the browser gets the DNS record with the IP address, it’s time for it to find the server on the Internet and establish a connection.
* Using the public Internet routing infrastructure, packets from a client browser request get routed through the router, the ISP, through an internet exchange to switch ISPs or networks, all using transmission control protocol, more commonly known as TCP, to find the server with the IP address to connect to.
* A CDN is a globally distributed network of caching servers that improve the performance of your site or app (the origin) by bringing the content closer to your users.
* Once the browser finds the server on the Internet, it establishes a TCP connection with the server and if HTTPS is being used, a TLS handshake takes place to secure the communication.
* Once the browser has established a connection with the server, the next step is to send the HTTP request to get the resource, or the page.
* Now that the browser has a connection to the server, it follows the rules of communication for the HTTP(s) protocol. It starts with the browser sending an HTTP request to the server to request the contents of the page. The HTTP request contains a request line, headers (or metadata about the request), and a body. The request line contains information that the server can use to determine what the client
* The next part of the request is the request headers. Headers pass extra information along from the client that help route the request, indicate what type of client is making the request
* The last part of the request is the body. The body is (usually) empty for a GET request like ours. For a request that manipulates resources, like POST, PUT, or PATCH, the body will contain the data from the client to create or update.
* Once the server has received the request from the client, the server processes it and sends back a response.
* The server takes the request and based on the info in the request line, headers, and body, decides how to process the request. For the request the server gets the content at this path, constructs the response and sends it back to the client. The response contains the following: a status line, response headers, the requested resource at that path
* The status line contains both the HTTP version and a numeric and text representation of the status of the request.
* The browser considers a status code in the 200s to be successful.
* Once the browser has received the response from the server, it inspects the response headers for information on how to render the resource. The Content-Type header above tells the browser it received an HTML resource in the response body.
* The first GET request returns HTML, the structure of the page. But if you inspect the HTML of the page (or any web page) with your browser’s dev tools, you’ll see it references other Javascript, CSS, image resources and requests additional data in order to render the web page as designed.
* As the browser is parsing and rendering the HTML, it is making additional requests to get Javascript, CSS, images, and data. It can do much of this in parallel, but not always.