1. When we hit https://www.techtonicgroup.com/ what happens?

When you enter https://www.techtonicgroup.com/ into the address bar the browser needs to resolve this to the numeric IP address.

The browser will have cached domains that you have already visited, and the operating system will have cached queries. If the browser or the OS don't have a cached copy of the IP address, then a request is sent to the DNS server.

Once you have the IP address that uniquely identifies a machine on the Internet, your browser issues a "HTTP/GET" request. The request includes metadata about the browser, user preferences, and any stored cookies for that domain. This HTTP request is sent off to the host server as packets. The packets have sequence numbers that allow them to be reassembled in order even if they take different paths.

Once the request arrives at the webserver, it generates a response which includes the HTML. The response is sent back to the client. The browser receives the response and begins to parse it for display. The HTML body will include links to CSS, JS and images. All of these will trigger additional calls back to servers. Once all the information has been obtained from the host, the final page is assembled and rendered to the user.

2. From start to finish how does that data reach you to be rendered in the browser?

Once the client receives the HTML from the host, the browser then starts parsing the page and builds the DOM tree.

When it finds a reference to an external entity such as a CSS file, an image file, a JS file, or anything else external to the page, it prepares to make another GET request for that resource. However, the HTTP standard specifies that the browser should not make more than two concurrent requests to the same domain. So it puts each request to a particular domain in a queue, and as each entity is returned, it starts the next one in the queue for that domain.

Stylesheets are parsed, and the rendering information in each gets attached to the matching node in the DOM tree. Javascript is parsed and executed, and DOM nodes are moved and style information is updated accordingly. When the parsing is ended and the document is ready and loaded, the events onload is fired. The page is now loaded and ready for the client's interaction.

3. What code is rendered in the browser?

HTML and CSS.

4. What is the server-side code's main function?

Server-side code's main function is to control what information is sent to the user. Server-side code is also used to interact with permanent storage like databases or files, process user input, and navigate to other pages.

5. What is the client-side code's main function?

Client-side code has mostly to do with the user interface. It also makes pages interactive, interacts with temporary and local storage, sends requests to the server, and receives data from server.

6. How many instances of the client-side assets (HTML, CSS, JS, Images, etc.) are created?

The browser will download the files referenced in the HTML page (CSS, JS, Images). Those files are parsed by your browser into their respective models and discarded. The browser then caches those requests after it has loaded them the first time. One GET request gets you one file.

7. How many instances of the server-side code are available at any given time?

The code on the server-side is present on the host server in one location. Packets of information are sent to the client according to the client's request. Sometimes there are multiple servers to handle more requests, or in case of a failure of one server. In which case the server-side code would be present in however many servers they have.

8. What is runtime?

Runtime is software that is executed while your program is running.

9. How many instances of the databases connected to the server application are created?

You usually have 1 DB and each application server would have multiple connections (or instances of a connection) to the DB, often referred to as a connection pool.