Before you begin:

Please note that using resources like ChatGPT as a mentor is highly valuable, but please don't try to submit answers gleaned from this remarkable bit of technology as your own :) Answers to these questions will consist of the answers themselves AS WELL AS the links/references to the sources you consumed that were helpful to you in providing your answer.

Be sure to use code samples in answering open ended questions. Submit your answers however you wish (Word/Google doc, zip file, github, etc).

**When would I use an SVG over a PNG?**I tend to use the following questions to determine whether it makes sense to use an SVG over a PNG:  
- Will an image or artwork need to be available in different sizes across the same page?  
- Are animations needed?  
  
One of the main advantages of SVGs is that they can scale to various sizes using the same file. In comparison, we would need various PNGs at the correct size to provide a similar experience. Using a single file for SVGs means less data to load for the client and you can do it without sacrificing quality which is impossible for raster images. Another really good reason to use SVGs is that, depending on how you add them on the site, you can animate them or use CSS and JavaScript to manipulate properties directly in the browser.

**What is the DOM?**DOM stands for document object model and it is essentially the representation of a website after the HTML and JS have been parsed. It is essentially a “tree of nodes where each node represents an HTML element” (FreeCodeCamp.org). It’s also what we can use to manipulate what a user sees in the browser. It’s important to note that the DOM often looks like regular HTML, but a key difference is that the DOM also included things that are added dynamically through JavaScript.  
  
Sources  
https://www.freecodecamp.org/news/what-is-the-dom-document-object-model-meaning-in-javascript/

**What's bubbling?**Bubbling is what happens when an event is triggered in JavaScript. For example, if I add a click event listener on an HTML element, the child will trigger the event, and then the parent will also have that event triggered. That chain of triggers up through the tree of nodes is what is referred to as bubbling in JavaScript. It’s important to note that events can be set up to disable bubbling.

**What's a Promise?**

A promise is used in asynchronous JavaScript code. Essentially it works like an IOU that allows the program continue executing without blocking while we wait for a response. Once we get a response, whether valid or not, the promise is fulfilled. This is regularly used for API calls.

**How would I make a "sticky" nav bar?  
HTML:**

<nav class=”main-nav”>

  <ul>

    <li><a href="#">"Nav Link"</a></li>

    <li><a href="#">"Nav Link"</a></li>

    <li><a href="#">"Nav Link"</a></li>

  </ul>

</nav>

**CSS:**

.main-nav {

position: sticky;

top: 0;

}

The position sticky in the CSS makes sure that the nav will stay at the top even once we scroll down. Alternatively, we could use position: fixed for the header since it will always be at the top.

BONUS: How would I make that nav bar disappear after scrolling past a certain element?

**What would be a couple ways I could make a "gallery" that would organize tiles on a page? What are the pros and cons of each approach?**

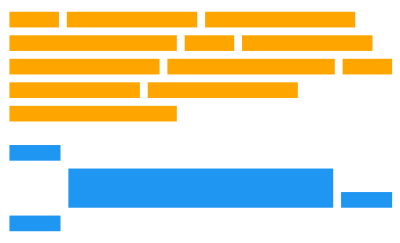
The two main ways to set this up would be by using CSS grids or Flexbox. Grids offer specific placements and sizing using columns and rows to control how elements are displayed. In contrast, flexbox only allows you to control if the items in the container will be aligned as a column or a row (see Figure 1).  


Figure Flexbox on top, Grid on bottom

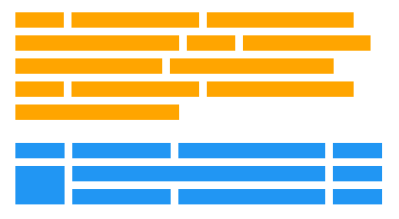
Another thing difference is in wrapping. Flexbox can use wrapping so content can fit using both axes, but we have no way of controlling both. For grid, we can control where content will be placed and enable wrapping, but any elements in the grid will still take up the full space of a grid column and row which can create visual gaps. This is different from flexbox that will only take up the necessary space for an element (see Figure 2).  


Figure : Flexbox on top, Grid on bottom

Sources:  
https://css-tricks.com/quick-whats-the-difference-between-flexbox-and-grid/

**Tell me the difference between the typical uses/implementations of ports 22, 80, and 443?**  
General use cases:  
- Port 22: SSH  
- Port 80: HTTP  
- Port 443: HTTPS

SSH essentially gives you remote access to a computer through a command line interface (CLI) provided that SSH is enabled, you know the computers IP, and you have user credentials. Port 80 and 443 are similar because they both handle HTTP requests, but port 443 requires the use of the HTTPS protocol which means that a valid SSL certificate must be present and that the connection will be encrypted. Most browsers use a lock next to the URL to show that HTTPS is being used with a valid SSL certificate.

**What kind of request would I use to submit credit card information? Why (as compared to a different type of request)?**Any credit card information should be submitted with a POST request through HTTPS. This makes sure that the data will be encrypted when sent to the server. This makes more sense than a GET request because you can pass parameters as part of the request payload instead of through query parameters which is one more layer of obfuscation. Though it is important to note that handling credit card information has more requirements and sites that handle this data should be PCI compliant.

**Write me some JavaScript that will generate a random, unique alphanumeric string of variable length.**

**How would I look at a previous version of a document using git?**The first step is to find the revision that had the previous version we want to see. We can get the history of a file with git log <file name>. Once we know the revision, we can see the previous version of a file with git show <rev>:<file name>.  
  
Sources:  
https://linuxhint.com/how-to-view-old-version-of-a-file-with-git/

**Go to any Amazon search result page (provide the link) and write me 2 query selectors (you can use CSS selectors or XPATH) in your browser:**I was unsure about this task since the search results page does not have a product description, so much as a product title..  
  
[Search Results](https://www.amazon.com/s?k=ram+32gb&ref=nb_sb_ss_ts-doa-p_2_4):

* **One that selects all the product images**let allImages = document.querySelectorAll(“img.s-image”)
* **and one that returns the items’text descriptions**let allIDescriptions = document.querySelectorAll(`.s-main-slot .sg-row h2 > .a-link-normal > span`)

[Item Page](https://www.amazon.com/VENGEANCE-C40-40-40-77-Optimized-Compatible-Performance/dp/B0B62WMDZP/ref=sr_1_1_sspa?keywords=ram+32gb&sr=8-1-spons&sp_csd=d2lkZ2V0TmFtZT1zcF9hdGY&psc=1):

* **One that selects all the product images**let allImages = document.querySelectorAll("#altImages .a-button-inner .a-button-text img")  
  Please note that these are the product images on the left sidebar. This selector will not work for the larger images since they are being displayed dynamically with JS.
* **and one that returns the items’text descriptions**let allIDescriptions = document.querySelector("#feature-bullets")

Use these selectors to wrap the values up in a JSON object and save it to local browser storage:

product\_summary = [

{

"image": <url>,

"text": <text>

},

{...}

]

Create a small web page with two buttons:

* One is green, the other red.
* Use CSS to position one directly in the center of the screen
* Use JavaScript to position the other 10px away from the right/bottom sides of the window after the document has loaded.
* When either is clicked, their colors should be exchanged (red-->green, green-->red)
* When either is clicked, inform the user which button was clicked in a form that appears automatically and can be dismissed by clicking an icon (https://icons.getbootstrap.com/ is a great resource for icons to include on your page)

BONUS:

* When either is clicked:
  + send an AJAX POST request to https://lambda.aws.com/982iduf0q38909ew8di0asaisd90qw0eid with the POST body:

{

'token': 'ASDHe89asdjkqwQ423',

'btn\_name': <name of button>,

'random\_code': <random 7 char string - should be generated on every request>

}

* The response should be an MD5 hash of a string with the following format:

f"{token} {btn\_name} {random\_code}"

* Create an MD5 hash of the POST body elements you sent by using a single space to separate the token, btn\_name, and random\_code.
* Compare your hash with the POST response value. Log your comparison to the browser console.
* If the request fails, still log the POST body and associated MD5 hash to the console.

BONUS:

Let’s say I only know part of a file’s name and I wanted to find it using a linux terminal. How would I do that?

If I only know a part of a file then I would opt for using the ls command with the -lR flags to list and go through all subdirectories. Then I would pipe that into a grep command to match the name. For example, to find a file with the text “hello” in the name I would use the following:  
ls -lR | grep “hello”.