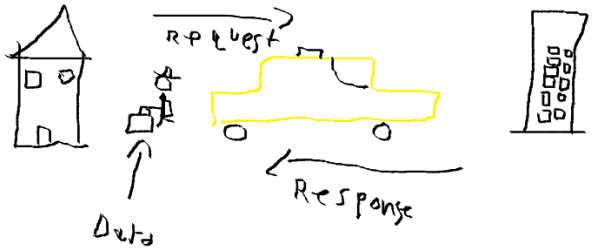
How the Web Works

In this lab, you'll be working with a partner to explore a little more about the internet, the web, requests, responses and more. You'll be reading and writing about concepts as well as practicing some of the commands that we saw during the lecture earlier.

Topic 1: The Internet and the World Wide Web

1) What is the internet? (hint: here)
   A network made up of a lot of networks of computers sending information with each other.
2) What is the world wide web? (hint: here)
   The web is an application built on the structure of the internet which allows users to interact with websites.
3) Partner One: read this page on how the internet works, Partner Two: read this page on how the world wide web works. When you're done reading, come back together and and answer the following questions
   a) What are networks?
      A series of computers that connect and share data.
   b) What are servers?
      A computer that is dedicated to sending its info to other computers.
   c) What are routers?
      A computer that connects to other computers to the ISP.
   d) What are packets?
      A smaller chunk of data that is sent instead of large chunks in order to prevent data loss.

4) Come up with a metaphor for the internet and the web, you can do a single one if you think of one that puts them together or two separate ones (feel free to use one you've heard today or read about if you can't think of a new one, but spend at least 10 minutes trying to think of something different before you resort to that)
   The internet is like a large city, and the web is like a taxi system that moves everything around the citry.
5) Draw out a diagram of the infrastructure of the internet and how a request and response travel using your metaphor (like the map and letters we saw during the lecture). Insert the drawing into this document (can be a picture of a physical drawing, a Google Drawing, a Figma drawing, etc)



Topic 2: IP Addresses and Domains

1) What is the difference between an IP address and a domain name?
   A domain name is a much easier to read name assigned to an IP address to make it easier for users.
2) What's devmountain.com's IP address? (Hint: use 'ping' in the terminal)
   `[104.22.13.35]`
3) Try to access devmountain.com by its IP address. It shouldn't work because we have our sites protected by a service called CloudFlare. Why might it be important to not let users access your site directly at the IP address?
   It makes it easier to hack as well as find information about the site.
4) How do our browsers know the IP address of a website when we type in its domain name? (If you need a refresher, go read this comic linked in the handout from this lecture)
   They get the assigned domain name for the IP address from the DNS.

Topic 3: How a web page loads into a browser

The steps of how a web page is requested and sent are in the table below. However, **they are out of order**. Unscramble them and explain your thinking/reasoning in the second two columns of the table.

Steps Scrambled	Steps in Correct Order	Why did you put this step in this position?
Example: Here is an example step	*Here is an example step*	*- I put this step first because ____* *- I put this step before/after ____ because ____*
Request reaches app server	Initial request	I put this step first since you would have to submit a request to start the process
HTML processing finishes	Request reaches app server	I put this here since the request would have to reach the server for it to react to it
App code finishes execution	Browser receives HTML	I put this here since the browser would have to receive the html before it can display it
Initial request (link clicked, URL visited)	HTML processing finishes	This goes after the browser receives the html since it couldn't process what it doesn't have.
Page rendered in browser	Page rendered in browser	I put this here since the page can only render after the code has been processed.
Browser receives HTML, begins processing	App code finishes execution	I put this here since the would only end after the page is displayed.

Topic 4: Requests and Responses

Setup

- Download the folder for this exercise from Frodo.
- Make sure you unzip it.
- Open it in VS Code
- Run `npm i` in the terminal (make sure you're in the web-works folder you just downloaded).

    - You'll know it was successful if you see a node_modules folder in the web-works folder.
- Run `node server.js` in the terminal (also in the web-works folder) and you should see a log to the terminal saying 'serving up port 4500'
- You'll be using this file to figure out what will happen when you make requests to this server, so read it over to see what's going on. We'll be getting into the two GET functions and the POST function.

Part A: GET /

- You'll start by looking at the function that runs when we make a get request to /, which looks like this: http://localhost:4500 or http://localhost:4500/
- You'll use the curl command to make a request and read the response in your terminal

1) Predict what you'll see as the body of the response:
   I would guess that it would be the entries array that contains the objects for each date.
2) Predict what the content-type of the response will be:
   I assume the content type will be a string.

- Open a terminal window and run `curl -i http:localhost:4500`

3) Were you correct about the body? If yes, how/why did you make your prediction? If not, what was it and why?
   I was not. I think the body was the title and the sub header of the page
4) Were you correct about the content-type of the response? If yes, how/why did you make your prediction? If not, what was it and why?
   Sort of. I predicted a string, and the content type is text/html.

Part B: GET /entries

- Now look at the next function, the one that runs on get requests to /entries.
- You'll use the curl command again. This time, you'll need to figure out how to modify it to get the response that you need.

1) Predict what you'll see as the body of the response:
   This time it should be the entries array.
2) Predict what the content-type of the response will be:
   JSON

- In your terminal, run a curl command to get request this server for /entries

3) Were you correct about the body? If yes, how/why did you make your prediction? If not, what was it and why?
   Yes. I looked at the /entries function and saw it was using the entries array.
4) Were you correct about the content-type of the response? If yes, how/why did you make your prediction? If not, what was it and why?
   Yes. The array is full of objects, so I guessed it would output as JSON.

Part C: POST /entry

- Last, read over the function that runs a post request.

1) At a base level, what is this function doing? (There are four parts to this)
   Creating a newEntry object, pushing it to the entries array, adding 1 to the globalID, and sendign the entries to the page
2) To get this function to work, we need to send a body object with our request. Looking at the function in server.js, what properties do you know you'll need to include on that body object? And what data types will they be (hint: look at the objects in the entries array)?
   We'll need an ID, which is a number, a date, which might be a string, and the content which will be a string.

3) Plan the object that you'll send with your request. Remember that it needs to be written as a JSON object inside strings. JSON objects properties/keys and values need to be in **double quotes** and separated by commas.

   {
   id: 3,
   date: "August 2"
   content: "It's been a minute"
   }

4) What URL will you be making this request to?
   Localhost:4500/entry
5) Predict what you'll see as the body of the response:
   All the same objects as the last page, but with the new one added.
6) Predict what the content-type of the response will be:
   JSON

- In your terminal, enter the curl command to make this request. It should look something like the example below, with the information you decided on in steps 3 and 4 instead of the ALL CAPS WORDS.
    - curl -i -X POST -H 'Content-type: application/json' -d JSONOBJECT URL

7) Were you correct about the body? If yes, how/why did you make your prediction? If not, what was it and why?
   Yes. The function adds the new object to the entries array, so it would show up with that.
8) Were you correct about the content-type of the response? If yes, how/why did you make your prediction? If not, what was it and why?
   Yes. It's adding a JSON object, so the type would be JSON.

Submission

1. Save this document as a PDF
2. Go to Github and create a new repository. (Click the little + in the upper right hand corner.)
3. Name your repository "web-works" (or something like that).
4. Click "uploading an existing file" under the "Quick setup heading".
5. Choose your web works PDF document to upload.
6. Add "commit message" under the heading "Commit changes". A good commit message would be something like "Adding web works problems."
7. Click commit changes.

Further Study: More curl

Visit this link and do the exercises using the website provided. Keep track of the commands you used in this document. (Don't forget to resubmit to GitHub when you complete this section)