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) ***The internet is a worldwide network of networks that uses the internet protocol suite from its 2 most important protocols. It's the backbone of the web and a way to connect computers all together and ensure that they find a way to stay connected.***
2) What is the world wide web? (hint: here)
    a) ***It's an interconnected system of public webpages accessible built on top of the internet. The web is different from the internet, the web is one of many applications built on top of the internet.***
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?
        i) ***A way to get more than one computer to communicate with one or others. You have to be able to link them together by connecting somehow with a cable or wifi, so on. A network isn't limited because you can connect as many computers as you want.***
    b) What are servers?
        i) ***Servers are a way computers are connected to the web and how they interact through responses and requests. They are computers that store webpages, sites, apps. When a device wants to access a webpage, a copy of the webpage is downloaded from the server onto the client machine to be displayed.***
    c) What are routers?
        i) ***Router are packet switches. A router is connected between networks to route packets between them. Each router know about it's sub-networks and which IP addresses they use.***
    d) What are packets?
        i) ***Packets describe the format in which the data is sent from server to client. When data is sent across the web, it sent as thousands if small chunks so that many different users can download one at a time.***
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)
    a) ***Roads are a good metaphor for the internet and the web can be vehicles that use the road. There are lots of transportation for the road and that could be the web.***
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)
    a) ***https://www.figma.com/file/eFDLcsC4X4MaVMPPGfAJK2/How-the-Web-Works?node-id=0%3A1***

Topic 2: IP Addresses and Domains

1) What is the difference between an IP address and a domain name?

   a) ***IP addresses are unique address to find the computer on the network. They are four numbers(0-255) connected by dots. A domain name is the information that you enter into a web browser in order to reach a specific website. The domain name functions as a link to the IP address.***
2) What's devmountain.com's IP address? (Hint: use 'ping' in the terminal)
   a) ***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?
   a) ***The domain is one of many under a single IP address because of shared servers. It also could be that the IP address is not appropriately assigned to the domain. Shared server accounts include Business, Reseller or WordPress specific hosting plans. In this case, the domain cannot be addressed through the IP address because they do not have a unique IP address associated with the account.***
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)
   a) ***The browser and OS search their cache first to see, but then it moves to the ISP(internet service provider), only if it doesn't have the IP address, it then goes to the root server which knows where to locate the .com TLD server(top level domain)…all of this can take milliseconds***

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 (link clicked, URL visited)	I put this step first because a page load begins when a user actually selects a link
HTML processing finishes	Request reaches app server	I put this step after the selection because the request has to reach the app server so it can then process the request
App code finishes execution	App code finishes execution	This is next because the app has to finish execution before sending a response back to the browser
Initial request (link clicked, URL visited)	Browser receives HTML, begins processing	This comes after the app code finishes execution so that the browser can receive the HTML response to process the DOM(Document Object Model)
Page rendered in browser	HTML processing finishes	This comes after the DOM processing so that the page can next be rendered(This is when the DOM finishes loading and is DOM ready)

Browser receives HTML, begins processing	Page rendered in browser	The page finishes rendering is last and the window load event fires.

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:
    a) ***I think I will see a big heading 1 text of Jurrni and then a smaller heading of Journaling your journies on the main page (only HTML on page)***
2) Predict what the content-type of the response will be:
    a) ***Same as above, just the HTML text of 'Jurrni' and 'Journaling your journies'***

- 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?
    a) ***I was correct about the body! I made the prediction by looking in the server.js file in VS code and looking to see what the function returns from the 'app.get' area. I read the file in server.js and looked at the app.get parts to see what the functions are doing.***
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?
    a) ***Same as above! I looked at the server.js file in VS code and looked at what the HTML text is to see what the content is in the app.get areas in the file.***

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.
    - ***In the terminal, I ran curl -i http://localhost:4500/entries***

1) Predict what you'll see as the body of the response:
    a) ***I had no idea what I was going to see since I'm not quite sure what this is all about yet and am learning since this is new to me.***
2) Predict what the content-type of the response will be:

- In your terminal, run a curl command to get request this server for /entries
    - ***Same as above, since I am new to this type of learning. Since I already ran the command, I did predict that I would get something more of the date and time and things like that and I did get some dates. The content is just Hello world basically and more text.***

3) Were you correct about the body? If yes, how/why did you make your prediction? If not, what was it and why?

   a) ***Not really since I wasn't sure what to expect, but I do know that I did not think it would be an array of object.***

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?
   a) ***I wasn't really correct but I did start to understand the content type because of checking to see if I was correct about the the body part.***

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)
   a) ***The function is creating a newEntry to the 'entries' array, so it's basically creating another object for the array/***
   b) ***The second part of the function is pushing the new object to the 'entries' array and adding it to the end of the array.***
   c) ***The globalId++ is just incrementing one number to the id part of the object.***
   d) ***The response is the send the desired HTTP response and the request is an object containing information about the HTTP request with the parameters.***
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)?
   a) ***The properties are the id, date, and content and data types would be number, string, 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.
   a) ***{"id": globalId, "date": "September 16", "content": "my birthday!"}***
4) What URL will you be making this request to?
   - ***http://localhost:4500/***
5) Predict what you'll see as the body of the response:
   a) ***I predict that I will see my new entry to the end of the object array just added to the end.***
6) Predict what the content-type of the response will be:
   a) ***I predict that I will see my exact content added to the end of the array.***

- 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?
   a) ***NO, I wasn't correct on the body because I couldn't get the command to work correctly in the terminal. Here is the command I used: curl -i -X POST -H 'Content-type: application/json' -d '{"id": globalId, "date": "September 16", "content": "my birthday!"}' http://localhost:4500/***
   b) ***Here is the command that did work and add to the array: curl -i -X POST -H 'Content-type: application/json' -d '{"date": "September 16", "content": "my birthday!"}' http://localhost:4500/entry***
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?
   a) ***No, I wan't correct on the content-type either because I am trying to just get the command to work in terminal.***
   b) ***Now that it is working with the right command it does look exactly how I thought it would.***

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)