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)

-The Internet is a worldwide network of networks that uses the Internet protocol suite (also named TCP/IP from its two most important

2) What is the world wide web? (hint: here)

-is an interconnected system of public webpages accessible through the Internet. The Web is not the same as 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?

-computers connected through tiplicaly routers but possible plugins

b) What are servers?

Servers are computers that store webpages, sites, or apps. When a client device wants to access a webpage, a copy of the webpage is downloaded from the server onto the client machine to be displayed in the user's web browser.

c) What are routers?

-links others computers to each other

d) What are packets?

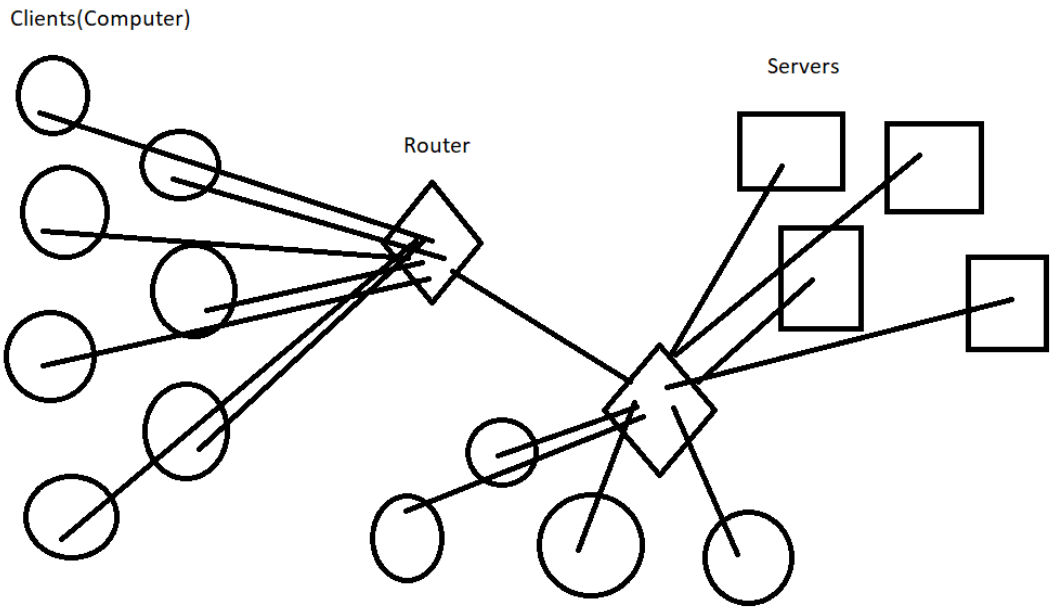
-small amounts data that are sent from server to client

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 the railroad system that connects people or things from far away and has stations(router) that send them to the right destination(server, clients)

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?

-IP is the real address that is harder to remember as people while a domain name is an easier way of remembering how to connect to the website. The domain name represents several IP addresses.

2) What's devmountain.com's IP address? (Hint: use 'ping' in the terminal)

172.66.43.107

172.66.40.149

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 helps prevent malicious traffic or bots from accessing the server.

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)

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 ____*
Initial request (link clicked, URL visited)		-I put this first because the user is requesting to the network to access the web server
Request reaches app server		Request reaches the server for processing
App code finishes execution		Server processing the request sends HTML response back
Browser receives HTML, begins processing		User receives HTML and starts to read HTML document

HTML processing finishes		User's browser is done reading html and making any additional requests
Page rendered in browser		Page is rendered in browser for user to see

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:

Jurrni, Journaling your journies

2) Predict what the content-type of the response will be:

HTML

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

Yes, it output the HTML information

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, because it says the content type is 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:

```
id: 0,

        date: 'January 1',

        content: 'Hello world'

    },

    {

        id: 1,

        date: 'January 2',

        content: 'Two days in a row!'

    },

    {

        id: 2,
```

```
        date: 'June 12',

        content: 'Whoops'

    }
```

2) Predict what the content-type of the response will be:

-json

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

Yes but it was more condensed. We knew it would return the properties of entry objects.

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, because json is used to display information related to arrays and objects.

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)

First it creates a new entry object.

Second it pushes new entry object to ‘entries’ array

Third, it adds one to the global ID.

Fourth, it will return response which will contain the code 200 and the new entries array

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)?

Date and the content. They’ll be of the String type.

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.

"{"date': "September 27", "content": "new content"}"

4) What URL will you be making this request to?

5) Predict what you'll see as the body of the response:

Status 200, and new entries array containing the previous objects and the new object that was just 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

curl -i -X POST -H 'Content-type:application/json' -d '{"date":"September 27", "content": "new content"}' localhost:4500/entry

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

Yes, because it returned the properties of the added objects

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, because objects are returned in 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)