How the Web Works

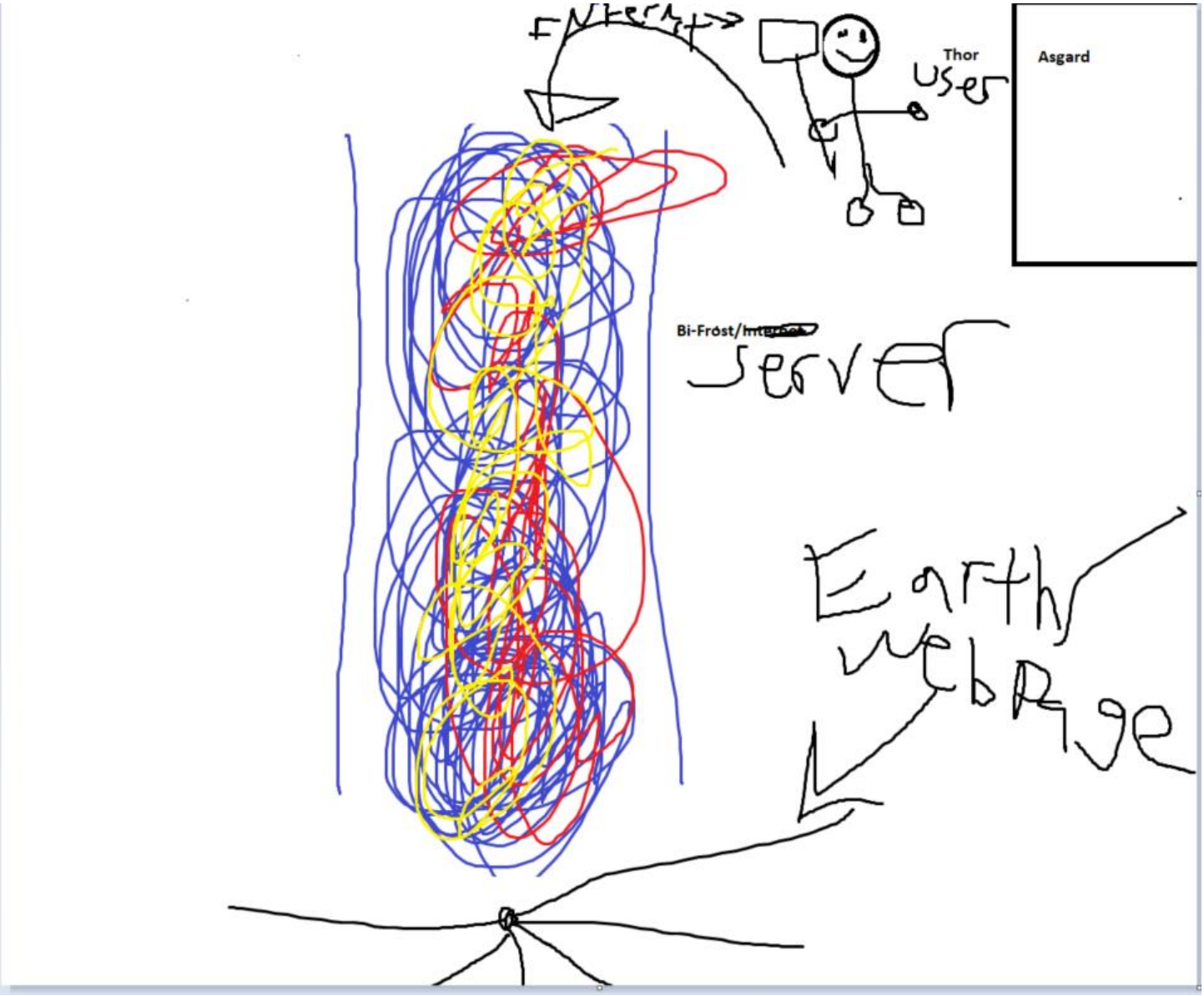
In this lab, you'll be working with a partner to explore a little more about the internet, the web, requests, responses and more. You'll be reading and writing about concepts as well as practicing some of the commands that we saw during the lecture earlier.

Topic 1: The Internet and the World Wide Web

1) What is the internet? The Internet is a worldwide of networks that uses the internet protocol suite.
2) What is the world wide web? An Information system on the internet which allows documents to be connected to other documents
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? When Two or more computers need to communicate with each other
    b) What are servers? Computers that store web pages, sites, or apps
    c) What are routers? A router is basically a central hub for a network, so that they can communicate more efficiently instead of all literally connected to each other
    d) What are packets? A "package" is data that is sent into smaller chunks across the web so it can hit more users faster.
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 earth highways are like the web and the BiFrost is the server. Asgard is our computer, his hammer is the internet. So Thor wants to access earth, he uses the hammer to summon the Bifrost to access earths highways which has multiple paths to choose from and access

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 address is an address with numbers, and domain name is a readable text way to access the website
2) What's devmountain.com's IP address? (Hint: use 'ping' in the terminal)

   172.67.9.59

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?

Because they could be under a shared server, and the IP address could be used for multiple websites, and if you did use the ip address it would bring you to the host server main page.

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)

Basically goes through a chain of searches, trying to do the least amount of searches possible. Starts with the OS, then goes to a resolver and checks the cache. It keeps going through a loop until

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	Initial meaning first
HTML processing finishes	Request reaches app server	Its gotta travel here first
App code finishes execution	Browser receives HTML, begins processing	Its there, so it needs to start something
Initial request (link clicked, URL visited)	HTML processing finishes	Processing finishes
Page rendered in browser	App code finishes execution	Goes through the code second?
Browser receives HTML, begins processing	Page rendered in browser	The Final product

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 for the 4500. For the 4500/ it will be id: globalId, date, content
2) Predict what the content-type of the response will be:

- 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? Yes I was. I figured it was this because of just the '/' on line 26.
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 I believe so, because I read the html text of the portion and I went with that

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: The dates and stuff
2) Predict what the content-type of the response will be: An array of data

- 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 was, because I saw on line 6 the stuff we needed.
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 I new if we put in the variable this will come in.

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) It is taking in the entry and making a new one
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)?

Id, date, and content

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.

curl -i -X POST -H 'Content-type: application/json' -d '{"id": "50",”date”: “August 30th”, “Content”: “Hello World”}' http://localhost:4500

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: Id: 50, date August 30th Content: Hello World
6) Predict what the content-type of the response will be: Id: 50, date August 30th Content: Hello World

- 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 whyNo because it also printed the other content
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? I just was waaaaay off

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)