Web Science

Quiz 1: February 28, 2023

Enter your answers directly into this document (with the exception of #2 and #3). All answers should be In Your Own Words, using complete sentences with proper spelling and grammar.

Save this document as either a Word docx or a PDF. For all questions other than #2 and #3, you will not receive any credit for answers not placed in this document.

When finished with the quiz, put everything you wrote (this document, all code, etc.) in your personal GitHub repo in a folder named **quiz1**. You must test your code on your VM, served with Node. No Apache!

1. **Short answers** (25 points): (Answer in complete sentences, explain your answers)

    a. (5) What are the potential advantages of using TypeScript's type system as opposed to typeless JavaScript?

    While Typescript has many advantages over Javascript a few of them include strict typing, structural typing, type annotations, and, type inference. Strict typing forces that a variable stay the same type no matter what, for example, if you create a number variable, it will stay a number variable and is not allowed to change. This is very useful to make sure that the data you are getting is correct and in the format that you require. Next, structural typing means that when you compare structs only the members of the structs types are checked. This allows you to set 2 variables of struct types equal to each other which is not allowed in a nominal type scheme. Then, type annotations are key to making sure that the variable that you are declaring is the exact type or types that you want it to be. This is not allowed to be changed so you know that no matter where in the code you are, the variable will always have a value of the type you set. Lastly, type inference is a useful feature that infers the type of your variable without you having the annotate it yourself. When you initialize a variable and assign it a value, TS will infer the type and set it automatically for you. These are just some of the many advantages of using TS but there are many more that I could not cover here.

    b. (7) What is a package.json file? What is it used for? How is it created?

    A package.json file is a file that is created by the initialization of a node application. This file holds all of the data needed to ship, install, and run the application that it is linked to. Inside it has the details of the project, scripts, dependencies, dev dependencies, and much more.

Without this file, there is no way for the application to know what scripts to run, packages to install, or even what project it is.

c. (3) Of the following permutations, identify which are valid and which are invalid for shipping/installing a Node application: 1. (package.json && package-lock.json); 2. package.json only; 3. package-lock.json only
Of the following permutations, permutation 3 is invalid while permutations 1 and 2 are valid. Permutation 1 is valid because it has the package.json file which is required to install the dependencies and scripts to run the project and it also has the package-lock.json file which will just be overwritten when npm i is run. Permutation 2 is valid because all a node application needs to install all the packages and scripts is the package.json file that is provided, it does not need the lock file to be shipped or installed. Finally, permutation 3 is not valid because the package-lock.json file does not hold all of the data that is required to build out a node application. This means that the application will now know what scripts or dependencies to install making this invalid.

d. (10) Describe **in detail** the sequence(s) of transaction(s) for a frontend to request data from some external entity via Node.

The first step for the front end to access data is to make the back-end node server. In the server, there will be many endpoints that will allow the front end to fetch these endpoints and receive the data. For these to work, the server needs to listen on the same port that the front end is running on. Once the front and back ends are running and listening on the same port, the calls that the front end sends will be heard by the node server. Once that is set up, you will create an endpoint on the server that when called will fetch to the external entity. This can be an API, Database, or anything external to connect to. Then when the node endpoint gets the data from the external source using an HTTP request it will then relay that information back to the front end using another HTTP request because it is almost the same process for the external API to send the data to the node server as it is for the node server to send the data to the front end. Finally, we will parse the data that is given back to us and there we have the data we needed all through our node server.

2. **Coding question**: (60 points) Here is a free API that does not require any API keys: http://universities.hipolabs.com/ – the documentation for which can be found here: https://github.com/Hipo/university-domains-list

   Create a new input box (or extend an already created input box) that can accept the name of a university and returns the API's information about that university. If the user fails to input a valid university, return the information for RPI. You might want to extend a recently due lab…

   Creativity matters; you need to really integrate this new information into your app. Make it feel like it is a meaningful, conscious, intentioned feature of your app. How you do that is up to you. Don't make it look like some random afterthought. Go beyond the minimum (but remember that creativity doesn't have to be visual). If you need to, write a short README.md file that tells me what I should consider for creativity. (creativity: 30 points of the 60 available for this question)

   You may use any and all open source libraries you want for this coding question, so long as you cite them in a README.md file.

3. (15) Ensure the package.json file for Q2 has no errors when I run npm install.

4. **Extra credit (+5)**: Why do your URLs need /node/ in order for Node to serve web pages on your VM?

You have to put node in your URL because it tells the server that you want to host using node instead of apache which will allow your server to run in the background.