Web Science

Quiz 1: March 2, 2017

100 points max

Place your name on the top of the document in the header

Enter your answers directly into this document (with the exception of #2 and #3)

All answers should be in be in Your Own Words, and use proper grammar

Make sure your answers use an alternative font and/or color

Save the document as

ITWS4500-S17-Quiz1-*yourname*-quiz1.docx

Place all documents/files including this one in a folder named

ITWS4500-S17-Quiz1-*yourname*-*yourRCSID*

When finished with the quiz, zip your folder and all related files into a file named

ITWS4500-S17-Quiz1-*yourname*-*yourRCSID*.zip

And submit it to LMS

1. **Frameworks** (25 points): (Answer in complete sentences, explain your answers)
   1. (5) What is MongoDB? How does it differ from MySQL (aka MariaDB)?

MongoDB is an open-source database. It is a Schema-less NoSQL database, JavaScript query language, with JSON-like document oriented structure and it is flexible and scalable. This differs it from MySQL because achieving scalability with MySQL often requires significant custom engineering work. MySQL is also a relational database, whereas MongoDB is Schema-less NoSQL.

* 1. (5) What is npm? How is it used? What it used for?

Npm is Node Package Manager. It manages node packages for your installation. It is a command line utility for interacting with Javascript packages you want to use. To use npm (assuming you have Node.js installed), just call “npm install <package>” in the command line.

* 1. (5) What is nvm? How does it work? Why is it used?

Nvm is Node Version Manager. It is a neat little bash script that allows you to manage multiple versions of Node.js. Calling “nvm use <version>” allows you to switch to a different version of Node.

* 1. (10) Describe the difference between Front-end and Back-end frameworks. Provide at least 2 examples for each in your answer. (Be clear in your decriptions, ie ‘why is it back/front-end?’)

Frontend frameworks are typically used to provide an interface to application for users (the view layer of an MVC). They provide predefined layouts and setups and methodology for use in creating styles for your output. Backend frameworks handle the user interaction and manipulates the data that gets displayed to the frontend/view (this is the controller layer of the mvc).

Bootstrap and Foundation are 2 front end frameworks.

Node.js and Express are 2 backend frameworks.

1. **Node.js** : (40 points) Create a webserver in node.js, using express – (NOT express-generator), which will serve a simple HTML page with an input filed and a button labeled ‘Run’ when GET request is received on <http://localhost:3000>. Upon entering a zipcode and clicking the button, the page server should get the current temperature for that zipcode and output a sentence that says whether it is Freezing (<=0C), Cold (btw 0 and 10), Warm (btw 11 and 25) or Hot (>25) – display the corresponding message in a unique color for each category. Include a button that allows the user to refresh the page and enter a new zipcode.

1. (15) Build an npm package.json file for Q2. If we run it, there should be no errors or warning when we try to install & run your code from #2 above. (You may assume your application name is *Quiz1Server*)
2. (20) Explain *in detail* what the following code does; (also add *stylized* comments to the code explaining what each line does)

*//requires the ‘net’ module, which provides an asynchronous network wrapper.*

var net = require('net')

var sockets=[]; *//creates empty array for clients*

*//create server*

var s = net.Server(function(socket) {

sockets.push(socket); //add client to array

*//event listening for when data is received*

socket.on('data', function(d) {

*//iterate through all the connected clients*

for(var i=0; i<sockets.length;i++) {

if (sockets[i]==socket) continue; *//skip the current client*

*//write the data to all sockets except the one that sent the data*

sockets[i].write(d);

}

});

*//Half-closes the socket. i.e., it sends a FIN packet.*

*//(event listening for when client disconnects)*

socket.on('end', function() {

*//finds the socket in the array*

var i=sockets.indexOf(socket);

*//then removes it*

sockets.splice(i,1);

});

});

s.listen(8088); *//start server, listening on port 8088*

The code seems like its just a simple chat server. It keeps track of all the connected clients in an array. When data is received, it writes out the data to the other clients. When a client disconnects, their socket is spliced from the array.