**WDI 18 Week 3 Quiz        Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Instructions**

Each page of the quiz has a set of related questions.

Code as much as you can, even if it’s psuedo-code. We are interested in your thought process and problem solving, not just syntax or one “right” answer.

**HTTP Basics**

1. When I go to <http://google.com> this is an example of *what HTTP verb*? \_\_\_\_\_\_\_\_\_\_\_
2. When I go to <http://google.com>, what file type does the server return? \_\_\_\_\_\_\_\_\_\_\_
3. A user fills out a form online and clicks a submit button. List 3 things that happen next.
4. Give an example of an HTTP Status Code and what it means.
5. Name and describe the constituent parts of the following URL: https://www.youtube.com/watch?v=y8Kyi0WNg40&t=1s

**Concept Review**

1. What do the following acronyms stand for? Give an example of using each.

* CRUD:
* JSON:
* HTTP:
* AJAX:
* API:
* OO:
* DB:

1. What is the difference between bower and npm? Give an example of where you might use one vs. the other.

**Express**

1. Briefly describe the purpose of each of the following files or directories in *our* Express projects so far.

* index.js
* package.json
* node\_modules/
* public/js/app.js
* public/views/index.html
* .gitignore

1. Jimmy has a sweet node/express app running his site, <http://www.jimmysworld.com>.    Now Jimmy wants his site to include an API that exposes his contact information to developers.  The /api endpoint should respond with a JSON object containing his name, phone number, and email (feel free to make these up!).  Help Jimmy add a route to his server code to make this possible:

app.\_\_\_\_ ( \_\_\_\_\_\_\_\_ , function (\_\_\_, \_\_\_) {

});

**Mongo**

1. Sal the Salad Seller wants to start using Mongo!  Design a simple schema for a salad.
2. Sal can’t access the Mongo docs but *does* remember the method names listed below.  Briefly describe how Sal might use each method.

* create
* connect
* find
* remove
* Schema
* model

**BONUS – Using APIs**

Juggler Supply Co. (http://www.juggleit.co) has an API with the following endpoint documentation:

**Search:** Return all matching juggling supply products.

Path: /api/supplies/search

Parameters:

* q - search query term or phrase.
* limit - (optional) number of results to return. Default 10.
* offset - (optional) results offset. Defaults to 0.
* danger (optional) - limit results to supplies with this danger level (safe, medium, superdanger).

Example response:

{ “data”: [

{ “name”: “Simple Balls”, “danger”: “medium”},

{ “name”: “Deceptively Simple Balls”, “danger”: “superdanger”}

]}

**Random**: Return a random juggling supply product.

Path: /api/supplies/random

Example response:

{ “data” : {“name”: “Bunnies”, “danger”: “medium” } }

**Top:** Return the top most popular juggling supply pages.

Path: /api/supplies/top

Parameters:

* limit (optional) - number of results to return. Default 2.
* danger (optional) - limit results to supplies with this danger level (safe, medium, or superdanger)

Example response:

{ “data”: [

{ “name”: “Frank’s Flaming Knives”, “danger”: “superdanger”},

{ “name”: “Hilda’s Hackeysacks”, “danger”: “safe”}

]}

1. Give a jQuery code example of how a frontend developer would request the top 5 products on this week’s popular items list. **Double bonus**: Add the data to the html page using jQuery.