**Node & Express: Assignment**

This checkpoint should take you between 60 and 90 minutes. If you spend longer than that on this checkpoint, reach out for help!

**Instructions**

Your goal for this checkpoint is to get the tests to pass. To do so, you will be creating a simple server with three distinct routes and error handling. Your server should follow the structure you've learned in the course. Complete the following tasks to pass the tests and this assignment.

**Note:** Because the app.js file is currently empty, your tests will fail immediately. These failures will revert back to normal test failures once you have the basic app structure up and running.

**Existing files**

| **file path** | **description** |
| --- | --- |
| app.js | An empty file. Place your application code here. |
| server.js | An empty file. Place your server code here. |
| middleware/validateZip.js | Includes a function definition. You will need to complete this middleware function. |
| utils/getZoos.js | A function that retrieves zoos based off of a zip. You will need to use this function but you should not need to modify it. |
| test/assignment.test.js | The tests your code will run against. You do not need to edit this file. |

**Tasks**

1. In the app.js file, require Express, create an Express application, and export that application.
2. In the server.js file, require the exported Express application from the app.js file and start your server.
3. (**Note:** Skip this step for now.) Use the morgan middleware. Make sure that it is included above any routes.
4. In the app.js file, you will create three routes. See below for more information on the routes.
5. In the middleware/validateZip.js file, you will fill in the function definition to create middleware that is checking for a valid zip code. Read the instructions below as you will *not* be building a robust zip code validator.
6. In the app.js file, you will create error handlers in the case of a missing route and a general error handler.

**NOTE:** If you are having trouble getting your tests to pass but think you've gotten it right, make sure to check for punctuation and spelling. The tests are looking for *exact* string matches.

**Routes**

To complete this project, you will create the following routes:

**/check/:zip**

This route will respond with a short message saying whether or not the provided zip code is recorded in the getZoos.js file. Use the getZoos() function and pass in the zip code provided via the route parameter to receive the zoos in that zip code.

* If the zip is a valid zip code (see middleware below) and matches an existing zip code respond with a short message.
* If the zip is a valid zip code (see middleware below) and *does not* match an existing zip code, respond with a different short message.

**Example Request URL**

http://localhost:5000/check/07502

**Matching Zip Response**

07502 exists in our records.

**No Matching Zip Response**

12345 does not exist in our records.

**/zoos/:zip**

This route will respond with zoos for the specified zip code. Use the getZoos() function and pass in the zip code provided via the route parameter to receive the zoos for that zip.

* If the zip is a valid zip code (see middleware below) and matches an existing zip code, respond with a short message and the zoos *joined by a semicolon.*
* If the zip is a valid zip code (see middleware below) and *does not* match an existing zip code, respond with a short message.

**Example Request URL**

http://localhost:5000/zoos/07502

**Correct Response with Single Zoo**

02121 zoos: COMMONWEALTH ZOOLOGICAL CORPORATION, Boston, Massachusetts

**Correct Response with Multiple Zoos**

33890 zoos: HARDEE COUNTY BOARD OF COUNTY COMMISSION, Zolfo Springs, Florida; PEACE RIVER REFUGE & RANCH INC, Zolfo Springs, Florida

**Correct Response with No Zoos**

07502 has no zoos.

**/zoos/all**

This route will respond with all zoos, but will only do so if the admin query parameter is provided and the value is set to true. Use the getZoos() function and pass in *no arguments* to receive all zoos.

* If these constraints *are* met, the route will respond with a brief message followed by all of the zoos *joined by a semicolon.*
* If these constraints *are not* met, the route will respond with an error message.

**Correct Request URL**

http://localhost:5000/zoos/all?admin=true

**Correct Query Parameter Response**

All zoos: WILDLIFE CONSERVATION SOCIETY, Bronx, New York; HARDEE COUNTY BOARD OF COUNTY COMMISSION, Zolfo Springs, Florida; PEACE RIVER REFUGE & RANCH INC, Zolfo Springs, Florida; CLOSE UP CREATURES L L C, Naples, Florida; KOWIACHOBEE ANIMAL PRESERVE INC, Naples, Florida; PRIVATE OWNER, Naples, Florida; COMMONWEALTH ZOOLOGICAL CORPORATION, Boston, Massachusetts

**Incorrect Query Parameter Response**

You do not have access to that route.

**Middleware**

Update the validateZip() function so that it validates an incoming parameter called zip according to the following constraints. You do not need to do any more validation than this to get the tests to pass.

1. The zip code must be 5 characters long.
2. The zip code must be all numeric. That is, no non-digit characters allowed.

Based on whether or not the zip code is valid, your server should respond in the following way:

* If the zip code *is* valid, move along to the next middleware function.
* If the zip code *is not* valid, move along to the *error handler function.*

**Invalid Zip Code Response**

Zip (ABCDE) is invalid!

**Error Handling**

If the user goes to a route that is not defined, respond with the following message:

That route could not be found!