|  |  |
| --- | --- |
| Sprint x, Assignment x Please also update the doc name with correct numbers. | |
| Assignment type: JS Interactive | |
| Assignment name  Can remain the same as the assignment topic name, or…  can be phrased in the active/imperative voice(depending on the task), e.g. “Create your own X” | “Requests, Responses & Routing with Express” |
| Description  The essential tasks a student must perform in order to complete the assignment | It’s time to create a server, and begin handling front end requests! Make your app listen, get, and respond to a web browser in preperation for attempting your first full-stack app! |
| Link to full code file (Answer)  Push a full code file to our [GitHub repo](https://github.com/bitdegree-foundation/academy-assignments-code-files) for this sprint’s assignment and paste a link to that file here. Please write the file in a concise and clear format, according to the “[example-js-code-file](https://github.com/bitdegree-foundation/academy-assignments-code-files/blob/master/example-js-code-file.js)” found in our repo. |  |

|  |  |  |
| --- | --- | --- |
| # | Step  Write each small step of the task. These assignments should be doable by students on their own (without any BitDegree instructor help). It can be any number of steps, but keep them pretty clear and separate (don’t combine 2 steps into 1 step). | 1x hint  Write brief text or partial code that would help the student figure out how to move forward or accomplish the given step. |
| 1 | Create a parent folder named "5.6 express-project". Navigate to that folder and run the following three commands: 1) npm init -y, 2) npm install express, 3) and npm install nodemon. |  |
| 2 | In your package.json, add "type": "module" and "dev": "nodemon 5.6.index.js" |  |
| 3 | (In your 5.6.index.js file) import express and store its function in a constant named "app". Then, declare a constant named "PORT" and set it to "3000". | Hint: use express(); |
| 4 | Next, make your app listen to your 'port' variable, and log a message to terminal that tells you what port your app is listening on. Open a webpage to that port and see if it shows anything. (remember to use "npm run dev" - instead of node 5.6.index.js - from now on to have your server update automatically) | Hint: use the .listen method  Hint: use an arrow function  Hint: if you see "cannot GET", you're app is working! |
| 5 | Give your app something to do for a GET request, like log the request and then send a response saying, "Your 5.6 app is running". | Hint: use the .get method  Hint: make this get only deal with completely-empty endpoints ("/")  Hint: use the .send method |
| 6 | Give your app a second GET route for "/search", have it log and store the query in a constant, and send back the query as a part of string response | Hint: use .send method  Hint: use { q } |
| 7 | Create a folder within your 5.6 express-project named "router". Create a new file in this folder named "routes.js". In your routes.js, import express; also, export a constant named "router" that is storing your express's Router function. Cut and paste your app.get logic from your index.js file to your routes.js file | Hint: use .Router() |
| 8 | In your routes.js file, update your app.get statements to router.get ones. In your index.js file, 1) import your router constant from your routes.js file, and 2) make your app use it. Reload your webpage to confirm that you have succesfully set up your router's routes. | Hint: use { router }  Hint: use .use on your app, and pass-in 'router' |
|  |  |  |
|  |  |  |
| ... |  |  |