**CS628 Full-Stack Development II – Backend**

**PE01 - Basic Node and Express - Meet the Node console**

Developed by Clark Ngo on December 29, 2020

School of Technology & Computing (STC)

City University of Seattle (CityU)

**Before You Start**

* Some steps are not explained in the tutorial**.** If you are not sure what to do:
  1. Consult the resources listed below.
  2. If you cannot solve the problem after a few tries, ask a TA for help.

**Learning Outcomes**

Students will be able to:

* Learn Basics of Node and Express

**Resources**

* https://www.freecodecamp.org/learn/apis-and-microservices/basic-node-and-express/

**How to Submit**

* **Upload .zip file**
* **Write a 150-word summary to explain your understandings and findings from this lab assignment.**

**Basic Node and Express - Meet the Node console**

Node.js is a JavaScript runtime that allows developers to write backend (server-side) programs in JavaScript. Node.js comes with a handful of built-in modules - small, independent programs - that help facilitate this purpose. Some of the core modules include:

* HTTP: a module that acts as a server
* File System: a module that reads and modifies files
* Path: a module for working with directory and file paths
* Assertion Testing: a module that checks code against prescribed constraints

Express, while not included with Node.js, is another module often used with it. Express runs between the server created by Node.js and the frontend pages of a web application. Express also handles an application's routing. Routing directs users to the correct page based on their interaction with the application. While there are alternatives to using Express, its simplicity makes it a good place to begin when learning the interaction between a backend powered by Node.js and the frontend.

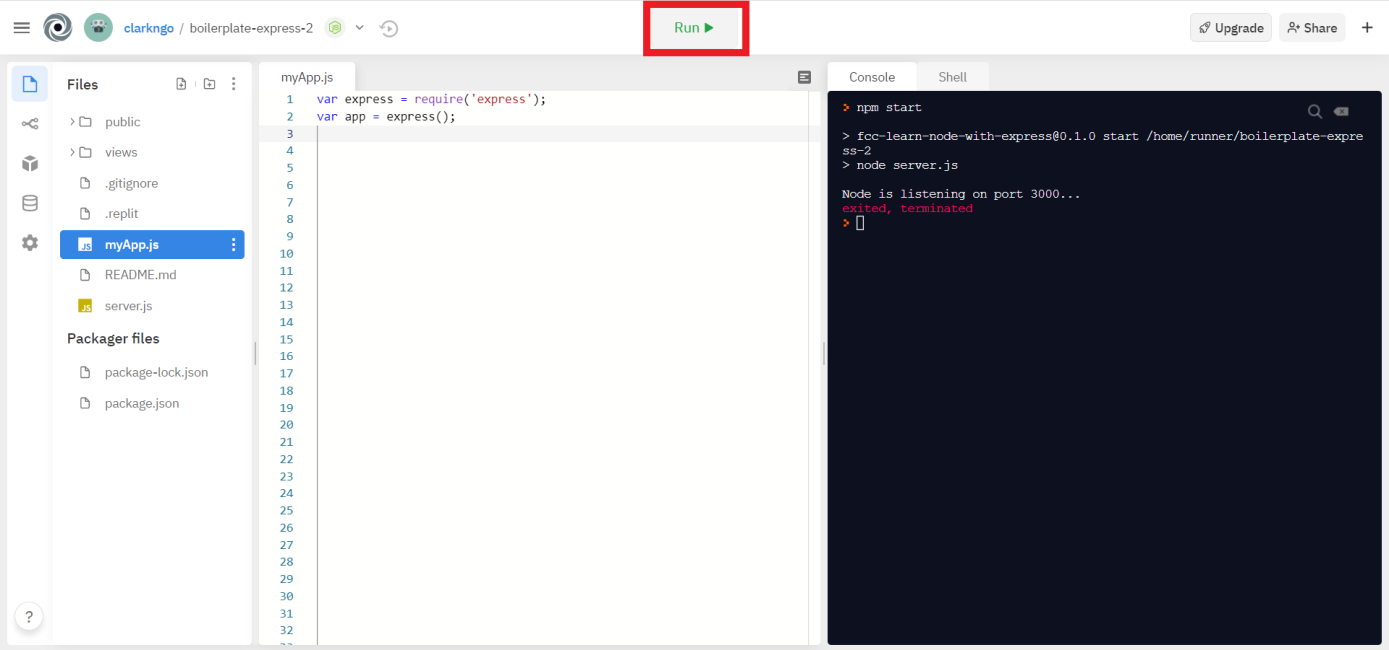
During the development process, it is important to be able to check what’s going on in your code.

Node is just a JavaScript environment. Like client-side JavaScript, you can use the console to display useful debug information. On your local machine, you would see console output in a terminal. On Repl.it, a terminal is open in the right pane by default.

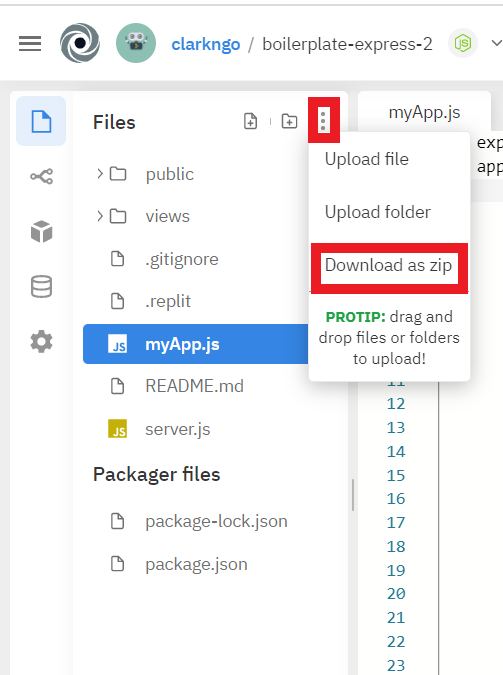
Modify the myApp.js file to log "Hello World" to the console.

We recommend keeping the terminal open while working on these challenges. By reading the output in the terminal, you can see any errors that may occur.

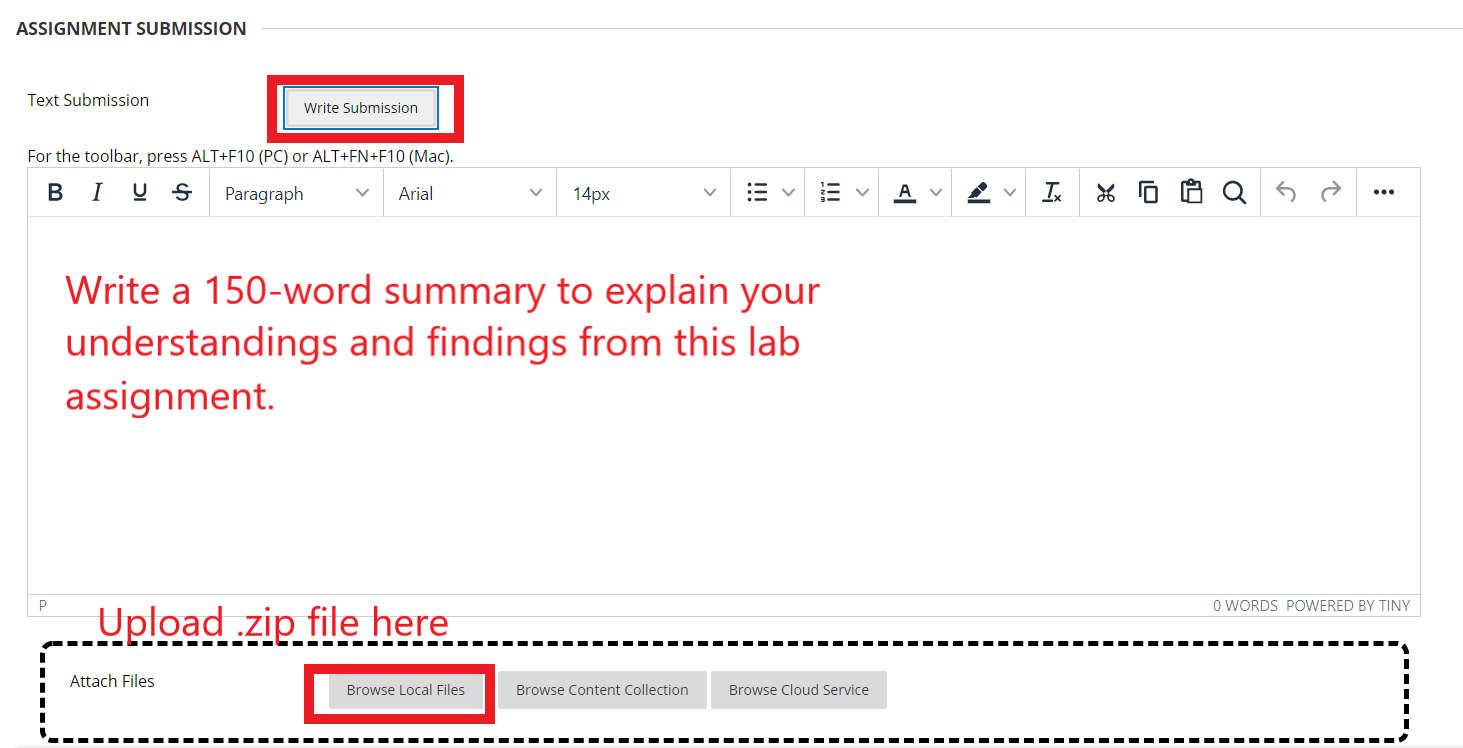
1. Start a new project on Repl.it using [this link](https://repl.it/github/freeCodeCamp/boilerplate-express).
2. Click Run to test your code



1. Download the project as .zip.



1. In BlackBoard, upload the .zip file and write a 150-word summary to explain your understandings and findings from this lab assignment.



I have experience with servlet and JSP technologies of java so I am familiar with dynamic web programming. As far as I understand javascript was a front-end only language at first. But looks like some people developed tools such as node to run javascript also in the back end. If we take a look at the folder and files of this web application, we can see that there is a view folder that includes an HTML file. And it includes an HTML form. We can also see the CSS file in the public folder. And then in the main directory, we can see the js files. These js files are required to make the website dynamic. Dynamic meaning taking data from the user in the form of get or post requests that include key-value pairs, processing the information in server-side, and then returning a response to the user.