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

**PE06 - Basic Node and Express - Use the .env File**

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/use-the--env-file

**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 - Use the .env File**

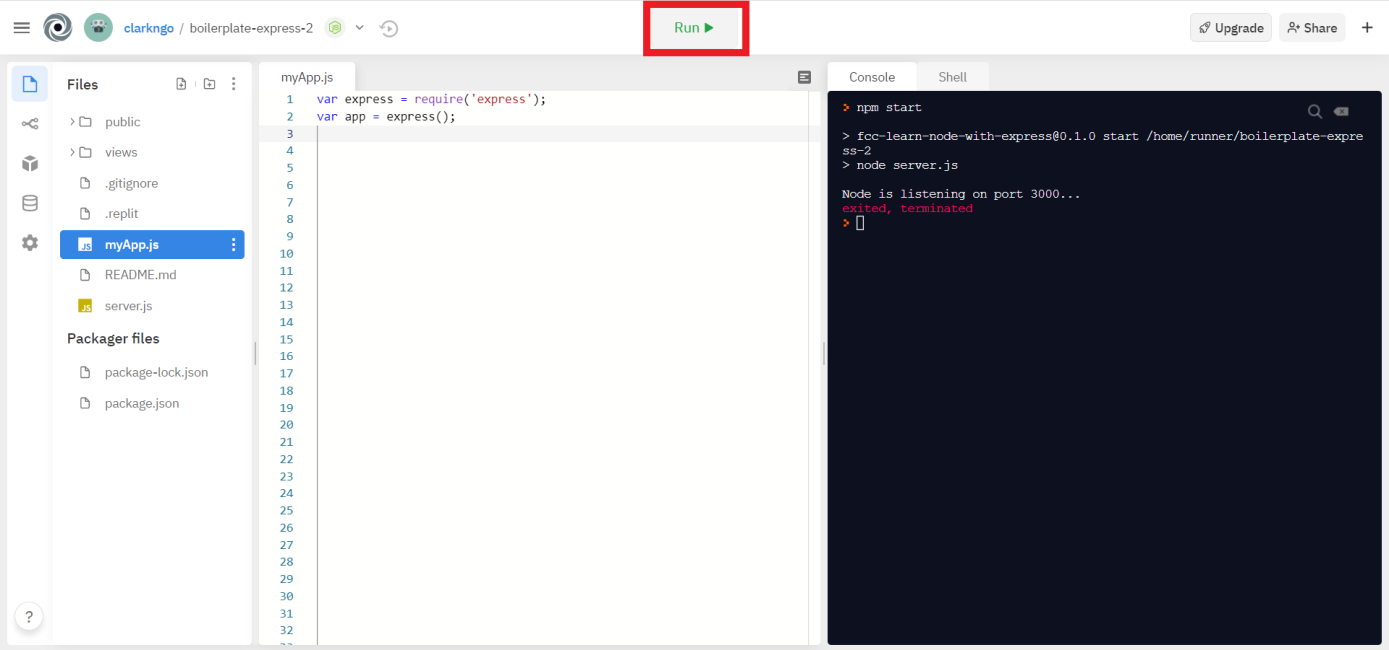
The .env file is a hidden file that is used to pass environment variables to your application. This file is secret, no one but you can access it, and it can be used to store data that you want to keep private or hidden. For example, you can store API keys from external services or your database URI. You can also use it to store configuration options. By setting configuration options, you can change the behavior of your application, without the need to rewrite some code.

The environment variables are accessible from the app as process.env.VAR\_NAME. The process.env object is a global Node object, and variables are passed as strings. By convention, the variable names are all uppercase, with words separated by an underscore. The .env is a shell file, so you don’t need to wrap names or values in quotes. It is also important to note that there cannot be space around the equals sign when you are assigning values to your variables, e.g. VAR\_NAME=value. Usually, you will put each variable definition on a separate line.

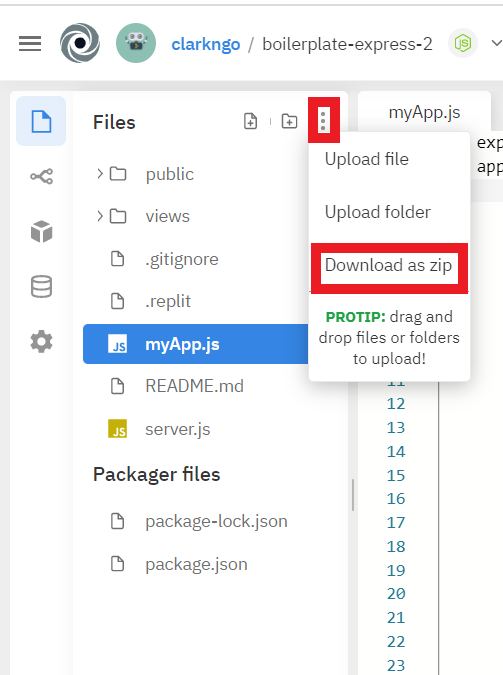
Let's add an environment variable as a configuration option.

Store the variable MESSAGE\_STYLE=uppercase in the .env file. Then tell the GET /json route handler that you created in the last challenge to transform the response object’s message to uppercase if process.env.MESSAGE\_STYLE equals uppercase. The response object should become {"message": "HELLO JSON"}.

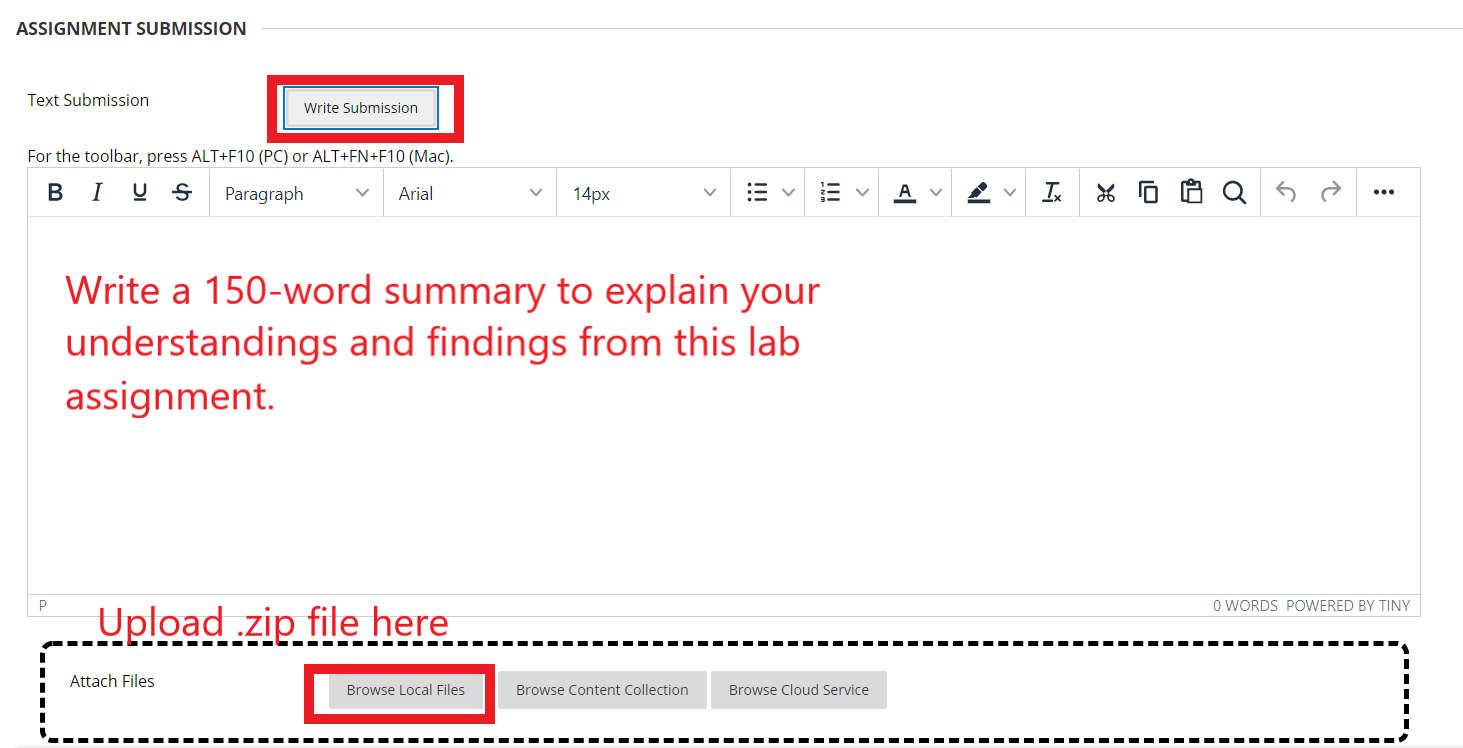
1. Start a new project on Repl.it using [this link](https://repl.it/github/freeCodeCamp/boilerplate-express).
2. Click Run to test your code



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.



Replit gives a warning when we try to create a file with the .env extension. But it allows it. Apparently using the .env file is deprecated. So we can go to the secrets tab in replit and create a key-value pair instead. Over here I created a key-value pair with the key “MESSAGE\_STYLE” and the value “uppercase”. After this, replit made me declare this variable in my controller with the following code.

   const mySecret = process.env['MESSAGE\_STYLE']

We then add logic to the controller with the following code. We create the response variable and assign it the string "Hello World". Then we check whether our MESSAGE\_STYLE environment variable is equal to “uppercase”. If so, we turn the value of the response variable to uppercase and assign it to itself.

   var response = "Hello World";

   if (process.env.MESSAGE\_STYLE === "uppercase") {

     response = response.toUpperCase();

   }

Then we say if the user asks for a get request at the /json URL, send back a json response. And we set the response to the key-value of message-response. Of course, the response variable holds the string value “HELLO WORLD!”