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

**PE09 - Basic Node and Express - Use body-parser to Parse POST Requests**

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-body-parser-to-parse-post-requests

**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 body-parser to Parse POST Requests**

Besides GET, there is another common HTTP verb, it is POST. POST is the default method used to send client data with HTML forms. In REST convention, POST is used to send data to create new items in the database (a new user, or a new blog post). You don’t have a database in this project, but you are going to learn how to handle POST requests anyway.

In these kind of requests, the data doesn’t appear in the URL, it is hidden in the request body. The body is a part of the HTTP request, also called the payload. Even though the data is not visible in the URL, this does not mean that it is private. To see why, look at the raw content of an HTTP POST request:

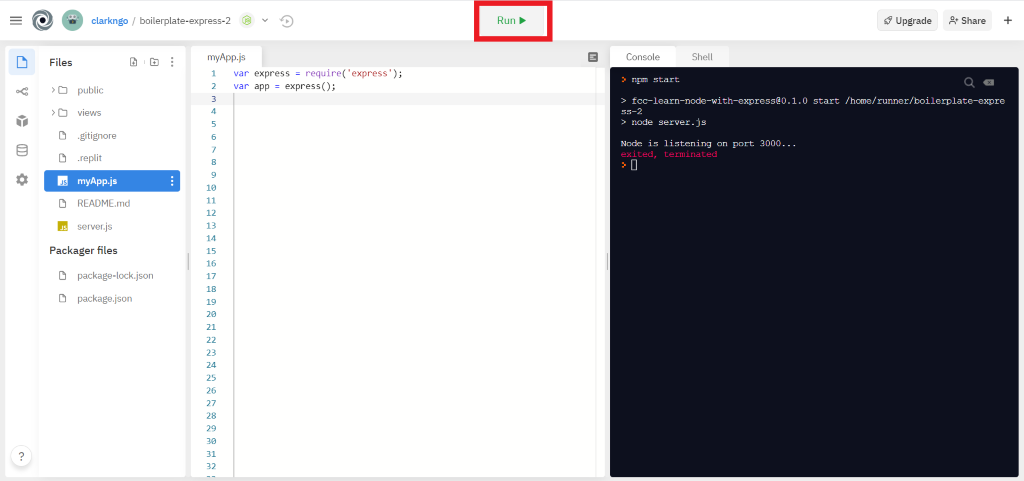
POST /path/subpath HTTP/1.0From: [john@example.com](mailto:john@example.com)User-Agent: someBrowser/1.0Content-Type: application/x-www-form-urlencodedContent-Length: 20name=John+Doe&age=25

As you can see, the body is encoded like the query string. This is the default format used by HTML forms. With Ajax, you can also use JSON to handle data having a more complex structure. There is also another type of encoding: multipart/form-data. This one is used to upload binary files. In this exercise, you will use a urlencoded body. To parse the data coming from POST requests, you have to install the body-parser package. This package allows you to use a series of middleware, which can decode data in different formats.

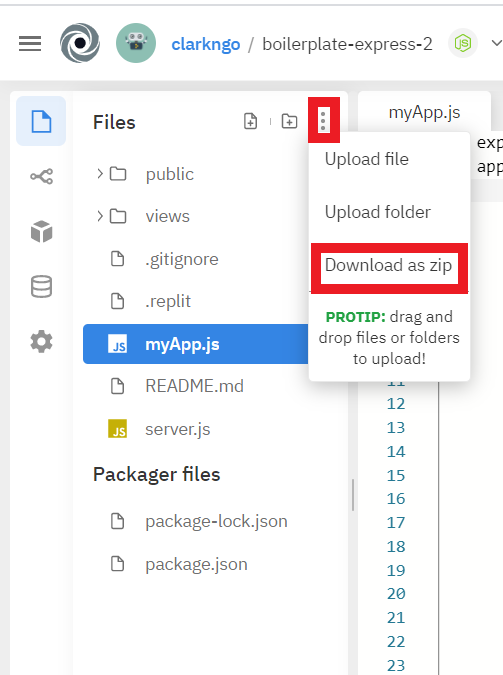
Install the body-parser module in your package.json. Then, require it at the top of the file. Store it in a variable named bodyParser. The middleware to handle urlencoded data is returned by bodyParser.urlencoded({extended: false}). Pass to app.use() the function returned by the previous method call. As usual, the middleware must be mounted before all the routes which need it.

**Note:** extended=false is a configuration option that tells the parser to use the classic encoding. When using it, values can be only strings or arrays. The extended version allows more data flexibility, but it is outmatched by JSON.

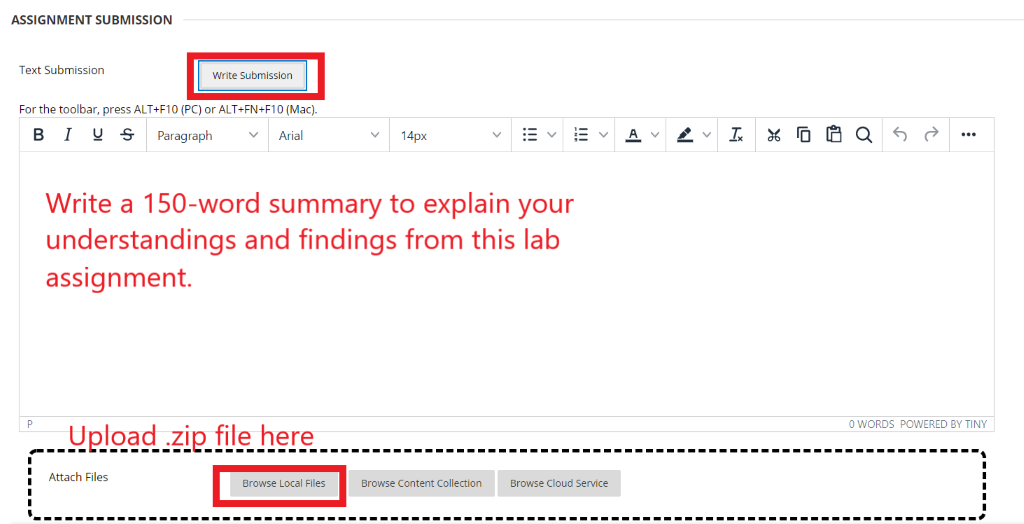
1. Start a new project on Repl.it using [this link](https://repl.it/github/freeCodeCamp/boilerplate-express).
2. Click Run to test your code



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.



This module showed us that we can invoke web services via POST requests as well as GET requests. Works the same way dynamic web pages work. I was wondering whether calling web services via post requests was possible, so glad I learned that this module.

GET request send the parameters through the URL. POST request sends the parameters in the HTTP body.

bodyParser.json middleware allows us to tell the server to parse JSON data sent in the POST request.