Full Stack IV - Express + Middleware

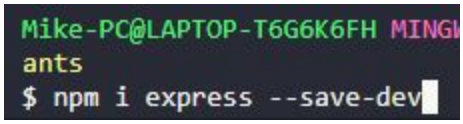
- Intro to Express
- Middleware + Postman

Developer Note:

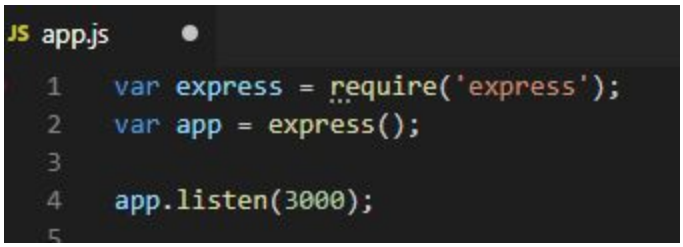
Please use your socket.io application from Full Stack lll session 5.

Exercise #1 – Download and Configure Express.js

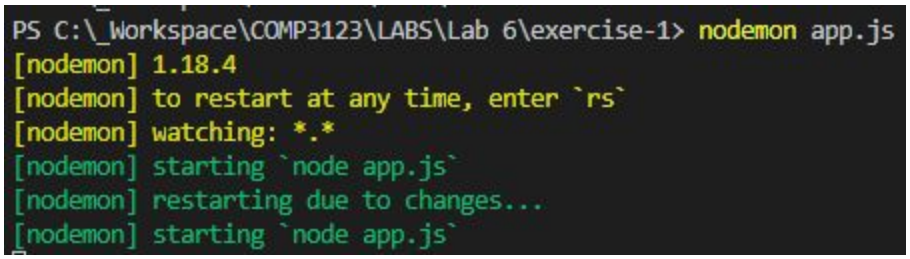
1. Navigate to the application folder and run the following command to install express js.



2. In the **server.js**, require express and configure it to listen on server port number 3000.



3. Run **nodemon app.js** at the command line to start the web server



4. Configure the following route using express to return HTML content when **'/html'** route is requested

```
app.get('/html', function (req, res) {
    res.send('<html><head></head><body><h1>Hello world!</h1></body></html>');
});
```

5. Configure the following route using express to return JSON content when the **'/json'** route is requested

```
app.get('/json', function (req, res) {
    res.json({ firstname: 'John', lastname: 'Smith' });
});
```

6. Use the browser to request the routes **/html** and **/json** to view the results.

7. Configure the following route and use express for matching the route paths.
http://expressjs.com/en/guide/routing.html

```
app.get('/toronto+team', function (req, res) {
    res.send('<html><head></head><body><h1>Go Toronto!</h1></body></html>');
});
```

8. Create a pattern to match the following routes.
There should be a wild card operator in some or one of the characters ***, +, ?, to** allow any text in between the 'toronto' and 'team' route pattern.

/toronto**raptors**team
/toronto**marlies**team
/toronto**abc**team
/torontoteam

Exercise #2 – Postman and Middleware

1. Open a command prompt create a directory for **exercise-2**
2. Open Visual Studio Code and open the folder **exercise-2**
3. Add a file named **app.js** and write the following code.

app.js

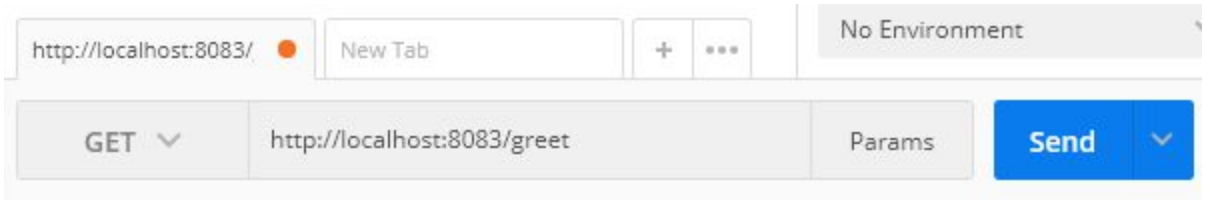
```
1  var express = require('express');
2  var app = express();
3
4  app.listen(3000);
5
```

4. Run **nodemon app.js** at the command line to start the web server

5. Install **Postman** extension for Chrome as a REST client
https://developers.sap.com/cis/tutorials/api-tools-postman-install.html

6. Write the following code in **app.js** and use Postman to trigger the routes and inspect the results.

```
app.get('/greet',  function (req, res) {
  res.send('hello world!');
})
```



7. Add the following custom middleware function to log the request time. Update the GET route to use the requestTime. Use POSTMAN to test the results.

```
var requestTime = function (req, res, next) {
    req.requestTime = Date.now()
    next()
  }

  app.use(requestTime)
```

```
app.get('/greet',  function (req, res) {
  console.log('GET recieved: ' + req.requestTime);
  res.send('hello world!');
})
```

8. Write a route for POST, PUT, DELETE routes and test with POSTMAN