

Homework - Express Generator

- Express Generator + Router

Developer Note:

Create a new project structure and run install express-generator

Exercise #1 – Express Generator

1. Create a folder named Lab
2. Open a command prompt create a directory for **exercise-1**
3. Open Visual Studio Code and open the folder **exercise-1**
4. Run the following to install the express application generator

```
npm install express-generator -g
```

5. The following creates an Express app named **myapp**. The app will be created in a folder named **myapp** in the current working directory and the view engine will be set to [Pug](#)

```
express --view=pug myapp
```

The result will be the following file skeleton application structure being created in **myapp**.

```
create : myapp\  
create : myapp\public\  
create : myapp\public\javascripts\  
create : myapp\public\images\  
create : myapp\public\stylesheets\  
create : myapp\public\stylesheets\style.css  
create : myapp\routes\  
create : myapp\routes\index.js  
create : myapp\routes\users.js  
create : myapp\views\  
create : myapp\views\error.pug  
create : myapp\views\index.pug  
create : myapp\views\layout.pug  
create : myapp\app.js  
create : myapp\package.json  
create : myapp\bin\  
create : myapp\bin\www
```

6. Change directory to the **myapp** and then install the application dependencies

```
change directory:  
> cd myapp  
  
install dependencies:  
> npm install
```

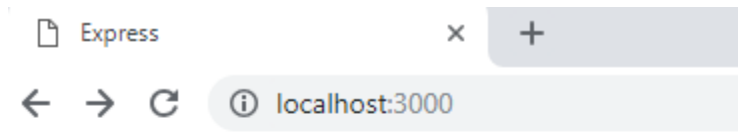
7. Navigate to the **package.json** file and review the following scripts section. A CLI **start script command** has been generated so that node will execute a file from the bin/www folder.
<https://docs.npmjs.com/cli/run-script>

```
"scripts": {  
  "start": "node ./bin/www"  
},
```

8. Run the npm start CLI command to start the web server on port 3000

```
npm start
```

9. Navigate the browser to the following URL and view the response from node.



Express

Welcome to Express

Exercise #2 – Routes + Body Parser

1. In the same **exercise-1** directory, stop the running node application and install the following body-parser middleware.

<https://www.npmjs.com/package/body-parser>

```
npm install body-parser --save
```

2. In the directory structure navigate to the router folder.

```
└─ routes
   ├── index.js
   └── users.js
```

3. In the **user.js** file add the following code to include the **body-parser** middleware and set up the router to use the body-parser. Modify the code to reflect the following:

```
var express = require('express');
var bodyParser = require('body-parser')

var router = express.Router();

router.use(bodyParser.urlencoded({ extended: true }));
```

4. Remember to stop the running application and then rebuild the application with **node app.js** and then run the application with **npm start**

5. Using POST API Testing Tool build some post parameters and the following POST URL

The screenshot shows a web browser window with a POST API Testing Tool interface. The URL bar shows 'http://localhost:3000/'. The tool is configured for a POST request to 'http://localhost:3000/users'. The 'Body' tab is selected, and the 'Content-Type' is set to 'x-www-form-urlencoded'. The request body contains two parameters: 'firstname' with value 'Test' and 'lastname' with value 'User'. The 'Send' button is visible on the right.

Key	Value	Description
<input checked="" type="checkbox"/> firstname	Test	
<input checked="" type="checkbox"/> lastname	User	
New key	Value	Description

6. Write a route in the **user.js** that handles the POST submission and uses **body-parser** middleware to pull the request parameters from the form and display them in the console log.

```
First name: Test  
Last name: User  
POST /users 200 3.995 ms - 13
```

The response sent to client ie. POSTMAN from the user route should be the following 'POST received!' message:

