Documentation Packet Week Ending [Sep 17th]

|  |  |
| --- | --- |
| Team: \_\_\_\_\_\_\_ | Student: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

# Goals:

* Import ExpressJS middleware into your Nodejs HTTP server project
* Create “endpoints” users can navigate to in ExpressJS
* Serve HTML files with CSS stylesheets and Javascript files from ExpressJS
* Create a complete website using ExpressJS and the goals above
* Learn how to update, replace missing, and print DocPacs
* Create a schedule and track it over time

# Included Documentation

* ExpressJS Middleware (pg. 2)
* ExpressJS Team Website (pg. 2)

# Required Documentation

*  “Express Middleware (pg. 2)”
* “ExpressJS Team Website (pg. 2)”



*  Updated Version of this DocPac

# Changes:

* To reduce confusion, “Companies” and “Departments” are just called “Teams”

# Events:

* Sep. 13th: Quick Weekly Review
* Sep. 14th: [Live Tutorial] Updating DocPacs Lesson
* Sep. 15th: Last Week’s DocPac Full Review
* Sep. 15th: [Live Tutorial] Building a Schedule
* Sep. 15th: [Live Tutorial] Building an ExpressJS Application
* Sep. 17th: All Updated DocPacs due

# ExpressJS Middleware

As you know, extra code modules can be loaded into your projects to give you more tools to work with. Creating an HTTP server with only the ‘http’ module can be painstaking, which is why many people use ExpressJS. ExpressJS is a module that can “intercept” connections to your HTTP server, and perform a lot of actions automatically for you. There are many great online tutorials and videos on how to use ExpressJS for a basic HTTP server that can send static files. Take some time to research and practice these.

* Create a Nodejs HTTP server using ExpressJS to serve static files
* Print the code and submit in this DocPac

# ExpressJS Team Website

You and your team are responsible for creating a website for your team using NodeJS and ExpressJS. The requirements for this website are as follows:

* ExpressJS must have 5-6 endpoints, as follows:
  + A home page, telling the user about your team and linking to all other pages.
  + An ‘About’ page, listing each of the students in your team and linking to their about page
  + A page for each team member describing them, their specialties, and career goals.
* Each web page must use the same HTML layout, CSS stylesheet, and at least one feature from a “\*.js” file written in Javascript
* ExpressJS must serve these files according to their endpoints
* One team member must start a Github repo
  + All other members must have their code added to the project by having a Pull Request accepted by the project repo owner.
* Website must be running and presented to Mr. Smith before the DocPacs are submitted to receive credit.

# Reflection

**What was the most impactful thing you learned this week?** **How can/will you use it in the future?**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

**What did you do *wrong* this week that you could do better in the future, and how?**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

**What is an obstacle to succeeding in class that you can foresee, and what plan do you have to overcome it?**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

# Grading

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | Complete | Followed Instructions | |
| Reflection | | ❑ | ❑ | |
|  | | Present | Complete | No Errors |
| C:\Users\csmith\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\6867D38F.tmp ExpressJS Middleware (pg. 2) | | ❑ | ❑ | ❑ |
|  | | Present | Complete | No Errors |
| Eye Icon - Free Download at Icons8 ExpressJS Team Website (pg. 2) | | ❑ | ❑ | ❑ |
|  | | THIS IS THE WRONG VERSION | | |
| C:\Users\csmith\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\6867D38F.tmp Updated Version of this DocPac | | DO NOT SUBMIT THIS VERSION | | |
|  |  | | | | |
|  | | | | |

***FIRST ASSIGNMENT CODE***

//requirements to run the server

const express = require('express');

const http = require('http');

const url = require('url');

const path = require("path");

var app = express();

// these are the endpoints, whatever you look up, will follow the guide. all others follow the star path

app.get('/hello', function(req, res){

res.send("greetings humans i am computer");

});

//this endpoint looks up index.html, a file inside the folder, and places it on the webpage

app.get('/files', function(req, res){

res.sendFile(path.join(\_\_dirname+'/index.html'));

});

//if there is no endpoint, this is the default path

app.get('\*', function(req, res){

res.send('welcome to the page of nothingness');

});

//this allows the use of req.params, placing the values used on the website

app.get('/things/:name/:id', function(req, res) {

res.send('the number you typed is ' + req.params.id + ' and the word you typed was ' + req.params.name);

}

});

//the code defining the port number

app.listen(9001)