***Documentation Packet [ 06 27 09 24 ] Sep 27th 2024***

|  |  |
| --- | --- |
| Student Name: |  |
| Goals:  1. Collect HTTP POST Request parameters 2. Save to a file using NodeJS | Events:  1. DocPac Due *Next Monday* 2. Two-hour delay, Thursday |
| Included Documentation  1. HTML Forms 2. Home Game Server [Bonus] 3. Formbar Developer Listing (digital) | Required Documentation:  1. HTML Forms 2. Reflections |
| Changes/Notes:  * Additional Formbar position opened | |

# Home Game Server [Bonus]

1. Create a new NodeJS project
2. Place a HTML game you made in the ‘public’ folder, and enable reading from said public folder in your NodeJS project.
   1. You can use “Button Masher”, “Touch Controls”, “Canvas Mover”, your own game, or try to use your RPGMaker game from 9th grade.
3. Create a GET endpoint that uses **sendFile()** to send the main page of your game to the user
4. Get your parent’s permission to log into the home internet router
5. **Open the port** used by your NodeJS project
6. Find your **external IP address**
7. **Find or make a friend**
8. Get your friend to go to your external IP address at your NodeJS port and play your game
9. **[Bonus Bonus]** Buy a domain name online, and create an **A Record** to send users to your external IP address
10. **[Bonus Bonus Bonus]** At the end of the game, use client side javascript to send a POST request back to the NodeJS server, which uses the user’s IP address to update their high score in a file (see HTML Forms for most of this data).

# HTML Forms

1. Create an idea for a program that involves a user submitting information to be save and displayed later. For security reasons, assume you will be the only user.
2. Create a new NodeJS HTTP Server project using ExpressJS and EJS
3. Create an empty file called “data.json” with the following code inside:
   1. { “data” : [] }
4. Create a root GET endpoint ( ‘/’ ) that **renders** a ‘index.ejs’
   1. This page will contain two links. One to ‘/add’ a new entry, and another to ‘/view’ all entries
5. Create a GET ‘/add’ endpoint that renders ‘add.ejs’
   1. This page contains an **HTML Form**. It will have an **id**, the **method** of **POST**, and ‘/add’ as its **action**. This form must contain at least two of the following, but should contain as many as your program idea needs

|  |  |
| --- | --- |
| **input** type **checkbox**  **input** type **color**  **input** type **hidden**  **input** type **password**  **input** type **radio**  **input** type **range** | **input** type **text**  **textarea**  **input** type **checkbox**  **select** with **options**  **datalist** |

* 1. The form will also contain an **input** type **submit** button with a **value**

1. Create a POST ‘/add’ endpoint.
   1. Inside of a **try**:
      1. Validate the POST parameters that came in from the Form. If there is any missing or incorrect data, **throw** an **Error()**.
      2. If needed, convert the parameters to a **Javascript** **object** of your design to store the form data (it is already an object when it comes in)
      3. **Read** the ‘data.json’ file and save it’s content to a **Javascript object** using **JSON.parse()**. Save this to a variable.
      4. **Push** the form data object into the ‘data’ array of the variable you created in the step above.
      5. Convertthe variable you created above back to a string by using **JSOB.stringify()**. **Write** the string to the ‘data.json’ file, overwriting its contents.
      6. **Redirect** the user back to ‘/’
   2. Inside of a **catch**:
      1. **Render** a ‘error.ejs’ template, with the catch error as the argument.
         1. In the ‘error.ejs’ template, display the error in plain text
         2. Provide a link to go back to ‘/add’
2. Create a GET ‘/view’ endpoint:
   1. **Read** the ‘data.json’ file, conver to an object, and save to a variable as described above
   2. **Render** a ‘view.ejs’ template, with the file data as the argument
      1. **Using an for loop in EJS**, display each entry in the file’s ‘data’ array
      2. Provide a link back to ‘/’ or ‘/add’
3. **BONUS**: Instead of listing all entries, list a name or identifier as a link to the entry. When clicked, the link redirects to ‘/view?entry=<identifier>’. If a get request to ‘/view’ has an ‘entry’ query parameter, only render that entry on the ‘view.ejs’.

# Reflection

**How have you done with not having a teacher in the class this week? What tools did you use to get help or communicate?**

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

**What was one mistake you made in school or otherwise that you can recognize? What can you do in the future to prevent it from happening again?**

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

**How difficult was the assignment this week? What made it difficult/easy for you?**

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

**What are more ideas you can create now that you have completed the assignment for this week?**

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

# A picture containing text, monitor, screen, clipart Description automatically generatedGrading

|  |  |  |
| --- | --- | --- |
| 10 | You went above and beyond expectations. You applied knowledge that was not taught in this class in addition to what was taught. Additional rewards are given | * ***If the assignment does not have its own rubric, it will default to the rubric on the left.*** * All assignments start at 10/10 possible points * 1 point is deducted per infraction   + Lateness   + Mistakes   + Unprofessionalism   + Not following instructions * Outstanding submissions, or submissions on assignments not marked in “Required Documentation” can reward pogs |
| 10 | You performed as well as can be expected for this class. You show a complete understanding and made no mistakes. You have mastered the subject. |
| 8 | Assignment is complete. You show a good understanding of the subject, but there are mistakes or minor incorrect details. You are ready to move to new subjects. |
| 7 | You show and understanding of the subject, but there are serious errors, or there are pieces you can practically use without understanding them. Remediation needed. |
| 6 | Assignment is incomplete but/or you showed that you understand at least the fundamentals of the subject. Assignment is low effort. Serious need of remediation. |
| 5 | You show minimum effort, assignment is incomplete, or have serious mistakes. You did not demonstrate that you understand the content or purpose of the submission. |
| 0 | The work was not submitted, damaged, seriously incorrect, or unprofessional. The submission is rejected. |
|  |  |  |

# HTML Forms

|  |  |  |  |
| --- | --- | --- | --- |
| Pull Request correct | Works as described | Did not commit  ‘node\_modules’ folder |  |

# Reflection

|  |  |  |  |
| --- | --- | --- | --- |
| Selected an answer for each question that is unique to you and this week | Answered every question in each prompt | Answers were not repeats of previous weeks | Answers were not copies of assigned work this week |