

Informe de Laboratorio 04

Tema: NodeJS + Express

Nota

| Estudiante | Escuela | Asignatura |
|--|--|--|
| Rafael Diego Nina Calizaya rninacal@unsa.edu.pe | Escuela Profesional de Ingeniería de Sistemas | Programación Web 2 Semestre: III Código: 1702122 |

| Laboratorio | Tema | Duración |
|-------------|------------------|----------|
| 04 | NodeJS + Express | 06 |

| Semestre académico | Fecha de inicio | Fecha de entrega |
|--------------------|-----------------|------------------|
| 2024-A | 20 Mayo 2024 | 24 Mayo 2024 |

1. URL del Repositorio:

https://github.com/DrN25/pw2_24a/tree/main/Lab04

2. Tarea:

Cree una aplicación NodeJS con express, para administrar una agenda personal.

Peticiones y Funciones destacadas.

Se crearon varias peticiones y funciones para el funcionamiento de la página usando NodeJS y Express, entre ellas las más destacadas son:

- **app.post('/submit' ...)**: Esta petición se encarga de crear los eventos. Para ello recibe 3 atributos (fecha, hora, contenido), con los cuales crea una carpeta (si es que no existe) y un documento .txt con su contenido. Una vez creado el evento se almacena en la carpeta ".agenda" de manera funcional.

```
16 app.post('/submit', (req, res) => {
17     const {fecha, hora, contenido} = req.body;
18     const folderPath = path.join(__dirname, 'agenda', fecha);
19     const filePath = path.join(folderPath, `${hora}.txt`);
20     if(!fs.existsSync(folderPath)) {
21         fs.mkdirSync(folderPath, { recursive: true });
22     }
23     fs.writeFileSync(filePath, contenido);
24     res.send('Archivo guardado correctamente.');
```

- **app.get('/buscar' ...)**: Esta petición es la primera parte de la sección Editar. Esta solo recibe 2 atributos (fecha, hora) y con estos verifica que el documento elegido exista, si es así se coloca el contenido del documento en el textarea continuo al botón para que el usuario pueda editar el documento ya presente, caso contrario se notifica que no existe un documento con los datos enviados.

```
27 app.get('/buscar', (req, res) => {
28     const {fecha, hora} = req.query;
29     const filePath = path.join(__dirname, 'agenda', fecha, `${hora}.txt`);
30     if(fs.existsSync(filePath)) {
31         const contenido = fs.readFileSync(filePath, 'utf-8');
32         res.json({contenido});
33     } else {
34         res.status(404).send('Archivo no encontrado');
35     }
36 });
```

- **app.post('/editar' ...)**: Esta petición es la segunda parte de la sección Editar. Tras la verificación de la petición anterior y con el texto ya modificado, al presionar el botón de enviar se mandaran los 3 atributos (fecha, hora, contenido) como en la petición /submit, y tras ubicar al archivo se reescribirá el mismo pero con el nuevo contenido entregado por el usuario.

```
38 app.post('/editar', (req, res) => {
39     const {fecha, hora, contenido} = req.body;
40     const folderPath = path.join(__dirname, 'agenda', fecha);
41     const filePath = path.join(folderPath, `${hora}.txt`);
42     if(fs.existsSync(filePath)) {
43         fs.writeFileSync(filePath, contenido);
44         res.send('Archivo editado exitosamente');
45     } else {
46         res.status(404).send('Archivo no encontrado');
47     }
48 });
```

- **app.post('/eliminar' ...)**: Esta petición se encarga de eliminar los eventos. Para ello se reciben 2 atributos (fecha, hora), con los cuales se ubicará al documento para luego proceder a eliminarlo de su carpeta designada.

```
50 app.post('/eliminar', (req, res) => {
51     const { fecha, hora } = req.body;
52     const folderPath = path.join(__dirname, 'agenda', fecha);
53     const filePath = path.join(folderPath, `${hora}.txt`);
54     if (fs.existsSync(filePath)) {
55         fs.unlinkSync(filePath);
56         res.send('Archivo eliminado exitosamente');
57     } else {
58         res.status(404).send('Archivo no encontrado');
59     }
60 });
```

- **app.get('/arbol' ...)**: Esta petición se encarga de realizar el proceso para retornar el árbol que contiene a los directorios y documentos. Este llama a la función crearArbol() mandando como atributo la dirección del directorio, la cual retorna un árbol de su contenido con el formato pedido en la práctica.

```
81   app.get('/arbol', (req, res) => {
82       const arbol = crearArbol(path.join(__dirname, 'agenda'));
83       res.send(`${arbol.replace(/\n/g, '<br>')}`);
84   });
```

- **crearArbol():** Esta función lee el contenido del directorio actual, creando una constante la cual contendrá el árbol a devolver. Se itera sobre cada elemento, donde dependiendo si es carpeta o documento se colocará un prefijo -guiones u otros, y se utilizan — para indicar la jerarquía de carpetas y documentos.

```
62  const crearArbol = (dirActual) => {
63      let resultado = `${path.basename(dirActual)} [DIR]\n`;
64      const items = fs.readdirSync(dirActual);
65      items.forEach((item) => {
66          const rutaItem = path.join(dirActual, item);
67          const esDirectorio = fs.statSync(rutaItem).isDirectory();
68          const prefijo = esDirectorio ? '[DIR]' : '[FILE]';
69          const guion = esDirectorio ? '---' : '--';
70          resultado += `|${guion} ${item} ${prefijo}\n`;
71          if (esDirectorio) {
72              const subItems = fs.readdirSync(rutaItem);
73              subItems.forEach((subItem) => {
74                  resultado += `|    |-- ${subItem} [FILE]\n`;
75              });
76          }
77      });
78      return resultado;
79  };
```

Funcionamiento de la Página Web.

- **Página Web:**

Crear Nuevo Evento

Fecha24/05/2024

Hora9-10

ContenidoContenido 9-10
Version1

Crear

Editar Evento

Fecha25/05/2024

Hora12-30

ContenidoContenido 12-30
Version2

Buscar

Editar

Eliminar Evento Evento

Fecha25/05/2024

Hora12-40

Eliminar

Ver Eventos

```

agenda [DIR]
|-- 2024-05-24 [DIR]
|   |-- 9-10.txt [FILE]
|-- 2024-05-25 [DIR]
|   |-- 12-30.txt [FILE]
|   |-- 12-40.txt [FILE]

```

■ Probando Creación de Eventos:

Crear Nuevo Evento

Fecha25/05/2024

Hora10-00

ContenidoContenido 10-00
Version1

Enviar

```

agenda [DIR]
|-- 2024-05-24 [DIR]
|   |-- 9-10.txt [FILE]
|-- 2024-05-25 [DIR]
|   |-- 10-00.txt [FILE]
|   |-- 12-30.txt [FILE]
|   |-- 12-40.txt [FILE]

```

■ Probando Edición de Eventos:

Editar Evento

Fecha

25/05/2024

Hora

10-00

Buscar

Contenido

Nuevo contenido
Version2

Editar

10-00.txt: Bloc de notas

Archivo Edición Formato Ver Ayuda

Nuevo contenido

Version2

■ Probando Eliminación de Eventos:

Eliminar Evento Evento

Fecha

25/05/2024

Hora

10-00

Eliminar

```

agenda [DIR]
|--- 2024-05-24 [DIR]
|   |-- 9-10.txt [FILE]
|--- 2024-05-25 [DIR]
|   |-- 12-30.txt [FILE]
|   |-- 12-40.txt [FILE]
    
```

Pregunta.

Mencione la diferencia entre conexiones asíncronas usando el objeto XMLHttpRequest, JQuery.ajax y Fetch. Justifique su respuesta con un ejemplo muy básico. Eje: Hola Mundo, IMC, etc.

- **XMLHttpRequest:** Es la API original de JavaScript, requiere manejar de forma manual múltiples eventos.

```
const xml = new XMLHttpRequest();
xml.open('GET', 'https://url');
xml.onload = function() {
  if (xml.status === 200) {
    console.log(xml.responseText); // Se imprime "Hola Mundo"
  } else {
    console.error('Error:', xml.statusText);
  }
};
xmlh.send();
```

- **jQuery.ajax:** Es una biblioteca de JavaScript, la cual usa la función \$.ajax al configurar objetos.

```
$.ajax({
  url: 'https://url',
  success: function(data) {
    console.log(data); // Se imprime "Hola Mundo"
  },
  error: function(error) {
    console.error('Error:', error);
  }
});
```

- **Fetch:** Es la API más moderna. Su funcionamiento viene acompañado de "promesas", manejando respuestas como .then y .catch.

```
fetch('https://url')
  .then(response => response.text())
  .then(text => console.log(text)) // Se imprime "Hola Mundo"
  .catch(error => console.error('Error:', error));
```

3. Commits realizados:

Cambiando la implementacion de /arbol, en la cual ahora se mostrara el arbol en la misma carpeta, ademas de que se adecua su estructura a una similar a la pedida en la practica, y sigue mostrando ...

DrN25 committed 1 hour ago

Aplicando estilos a la pagina web

DrN25 committed 1 hour ago

Implementando la funcion de eliminar documento con la ruta post /eliminar. Se eliminara la pagina en caso de que al introducir la fecha y hora de un documento este exista, soltando la alerta de eli...

DrN25 committed 3 hours ago

Trabajando la funcion de editar archivo de la pagina, para lo cual se crearon una ruta get /buscar y otra post /editar. En la primera mediante la fecha y hora ingresados se buscara si ese archivo e...

DrN25 committed 3 hours ago

En index.js se creo la funcion crearArbol(), la cual se encargara de revisar los directorios creados en la carpeta actual, y luego revisara los documentos en cada una de estos, y en este proceso ir...

DrN25 committed 6 hours ago

Commits on May 24, 2024

Creando el archivo index.html e index.js. En el js se reciben parametros mediante un formulario hacia /submit, y en este llamado se crea una carpeta con la fecha indicada, y si existe este paso se ...

DrN25 committed yesterday

4. Rúbrica:

| Nivel | | | | |
|--------|----------------------|-----------------|--------------------|---------------------|
| Puntos | Insatisfactorio 25 % | En Proceso 50 % | Satisfactorio 75 % | Sobresaliente 100 % |
| 2.0 | 0.5 | 1.0 | 1.5 | 2.0 |
| 4.0 | 1.0 | 2.0 | 3.0 | 4.0 |

| Contenido y demostración | | Puntos | Checklist | Estudiante | Profesor |
|--------------------------|---|--------|-----------|------------|----------|
| 1. GitHub | Repositorio se pudo clonar y se evidencia la estructura adecuada para revisar los entregables. (Se descontará puntos por error o omisión) | 4 | X | 4 | |
| 2. Commits | Hay porciones de código fuente asociado a los commits planificados con explicaciones detalladas. (El profesor puede preguntar para refrendar calificación). | 4 | X | 3 | |
| 3. Ejecución | Se incluyen comandos para ejecuciones y pruebas del código fuente explicadas gradualmente que permitirían replicar el proyecto. (Se descontará puntos por cada omisión) | 4 | X | 3 | |
| 4. Pregunta | Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación). | 2 | X | 1.5 | |
| 7. Ortografía | El documento no muestra errores ortográficos. (Se descontará puntos por error encontrado) | 2 | X | 2 | |
| 8. Madurez | El Informe muestra de manera general una evolución de la madurez del código fuente con explicaciones puntuales pero precisas, agregando diagramas generados a partir del código fuente y refleja un acabado impecable. (El profesor puede preguntar para refrendar calificación). | 4 | x | 3 | |
| TOTAL | | 20 | | 16.5 | |