PROGRAMACIÓN INTEGRATIVA DE COMPONENTES WEB

NRC - ASIGNATURA:	19178
PROFESOR:	Ing. Jose A. Sancho Arias
PERÍODO ACADÉMICO:	2024_50

TAREA Nº 2.3

TÍTULO:

Manual de manejo de Formularios en frameworks de web components + React

ESTUDIANTE:

Betty Lizeth Rodriguez Salas

ENLACE VIDEO:

https://youtu.be/aC8RXSiIevQ

FECHA DE ENTREGA: 2024 / 08 / 03

Interfaz visual de formulario con listas

I. Introducción

Las actividades incluyen definir una aplicación base, implementar funcionalidades y documentar el proceso. Los entregables son una aplicación en React, la funcionalidad implementada, un manual detallado y un video demostrativo.

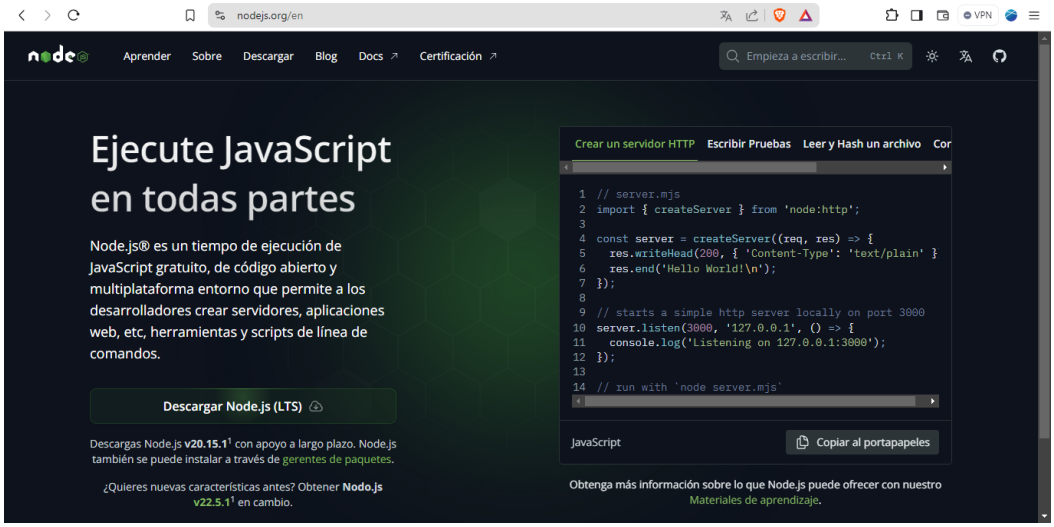
II. Objetivos

OBJETIVO: identificar el desarrollo de aplicaciones, implementando framework y librerías basadas en web componentes

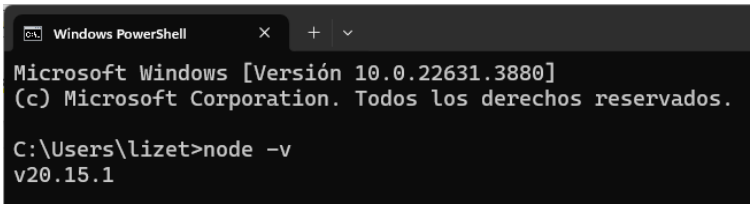
OBJETO DE ESTUDIO: Formularios con web components

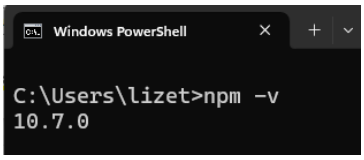
III. Instalación de plataforma de desarrollo

Primero accedemos a https://nodejs.org/en descargamos e instalamos los componentes necesarios para la plataforma base node js.



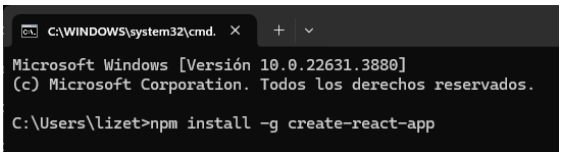
Iniciamos el cmd y ejecutamos el comando `node -version` para ver la versión de node instalado en el dispositivo.



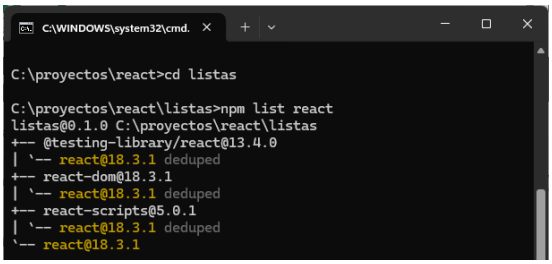


IV. Instalación framework React

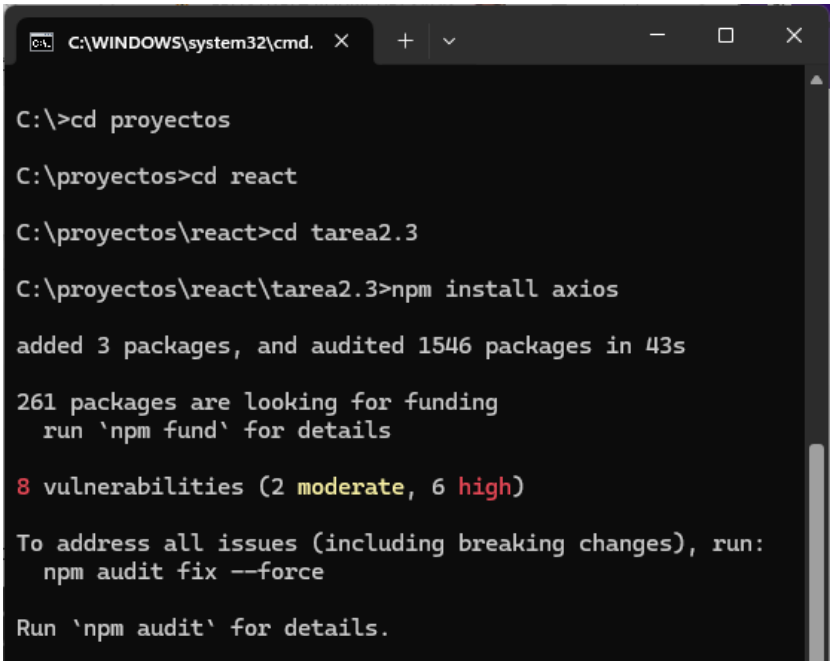
Dentro de la línea de comando vamos a instalar React a nivel global con los comandos



La versión instalada de React



Instalamos axios



Instalamos paquetes necesarios

```
C:\>cd proyectos/react/tarea2.3

C:\proyectos\react\tarea2.3>npm install reactstrap bootstrap

added 9 packages, and audited 1555 packages in 2m

263 packages are looking for funding
  run `npm fund` for details

8 vulnerabilities (2 moderate, 6 high)

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
```

V. Creación e inicialización de la tarea

Ingresamos en la carpeta de React agregando el comando `cd`

```
C:\proyectos>cd react
```

Creamos el nuevo proyecto llamado “tarea2.3”

```
C:\proyectos\react>npx create-react-app listas

Creating a new React app in C:\proyectos\react\listas.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...


added 1480 packages in 14m

261 packages are looking for funding
  run `npm fund` for details
```

Para inicializar agregamos el comando `npm start`

```
Windows PowerShell
Happy hacking!

C:\proyectos\react>cd listas

C:\proyectos\react\listas>npm start

> listas@0.1.0 start
> react-scripts start
```

Finalmente se activa el local host

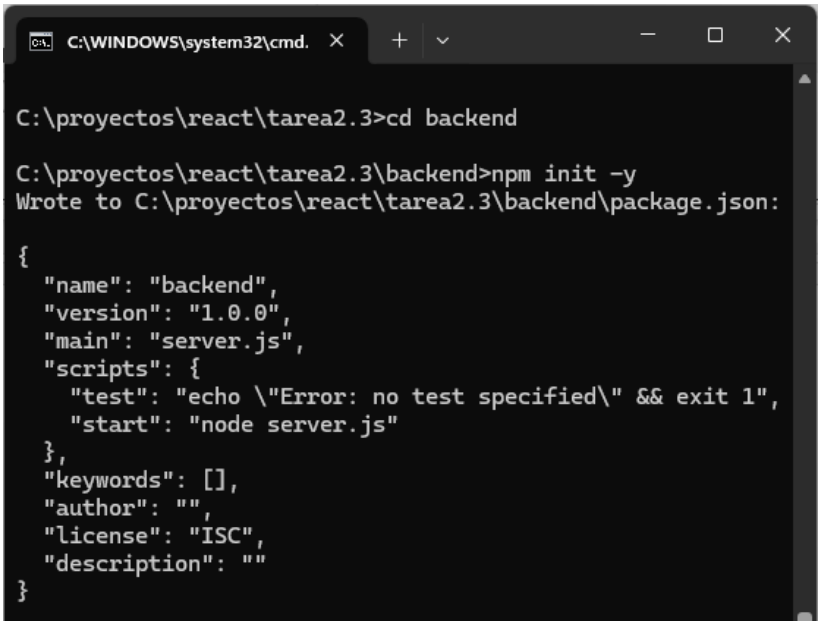
```
Windows PowerShell

You can now view listas in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://192.168.56.1:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

Visualizamos en el navegador el framework React.

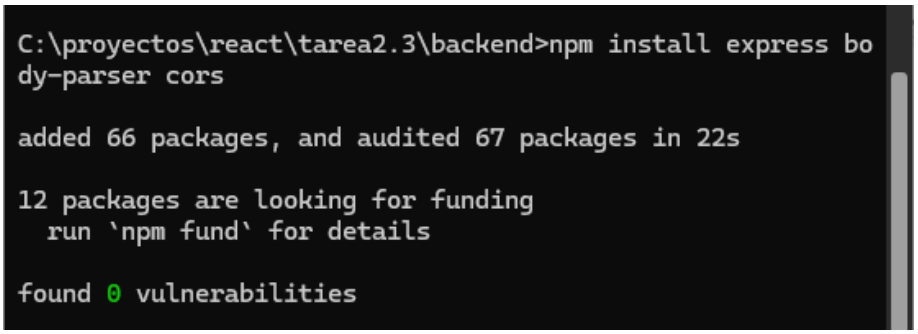
VI. Creación Backend

Para usar el servidor JSON vamos a usar Node.js con Express creamos una carpeta para backend.

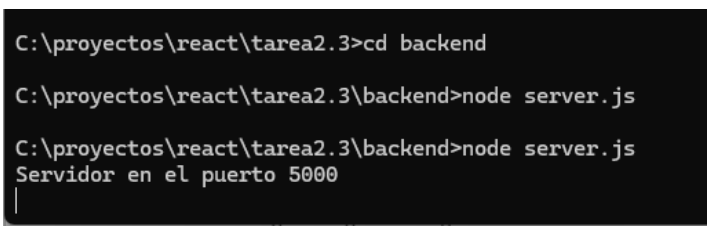
Iniciamos el proyecto node.js



Instalamos dependencias necesarias



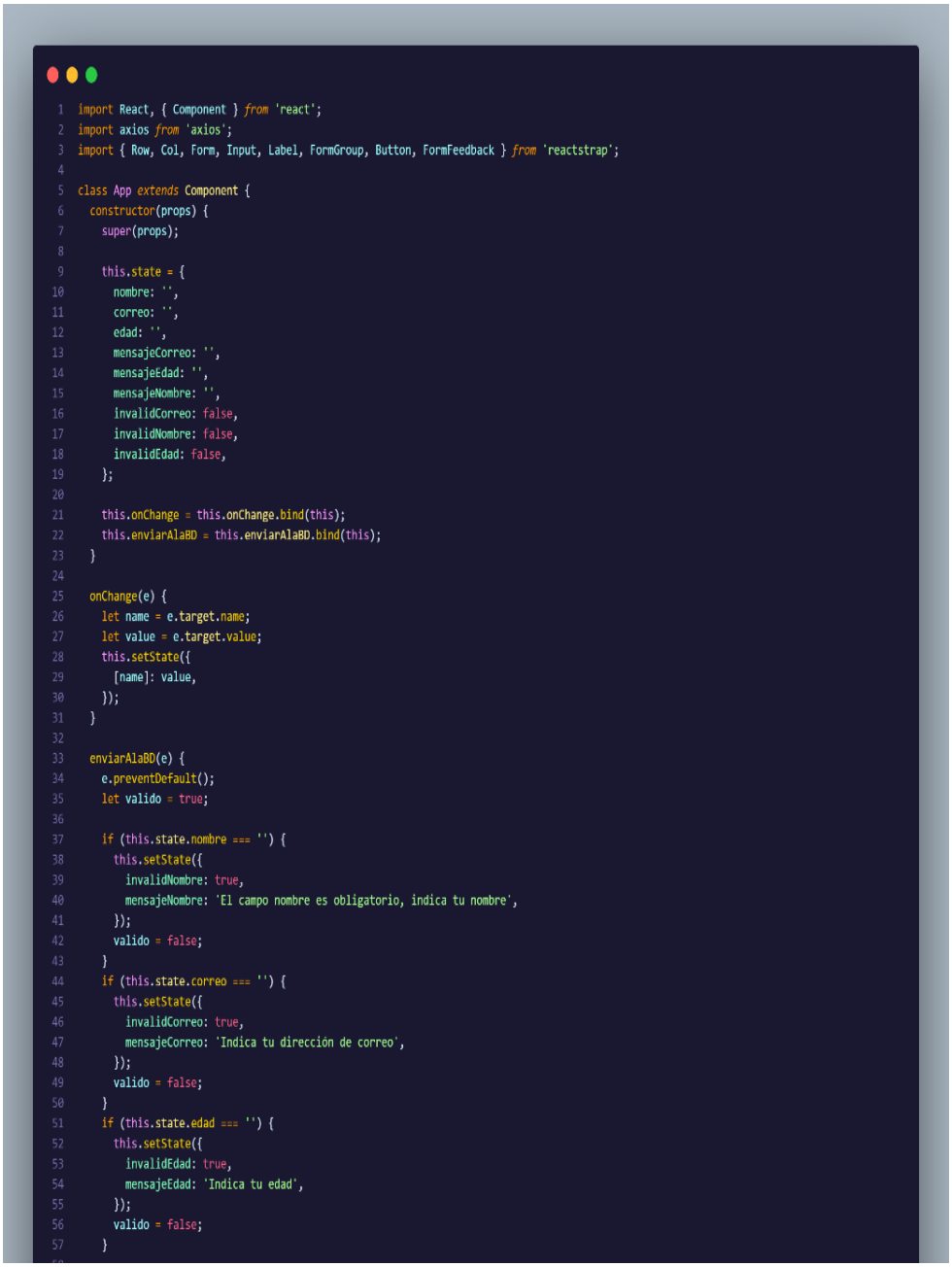
Iniciamos el servidor backend, anteriormente creamos un archivo llamado server.js



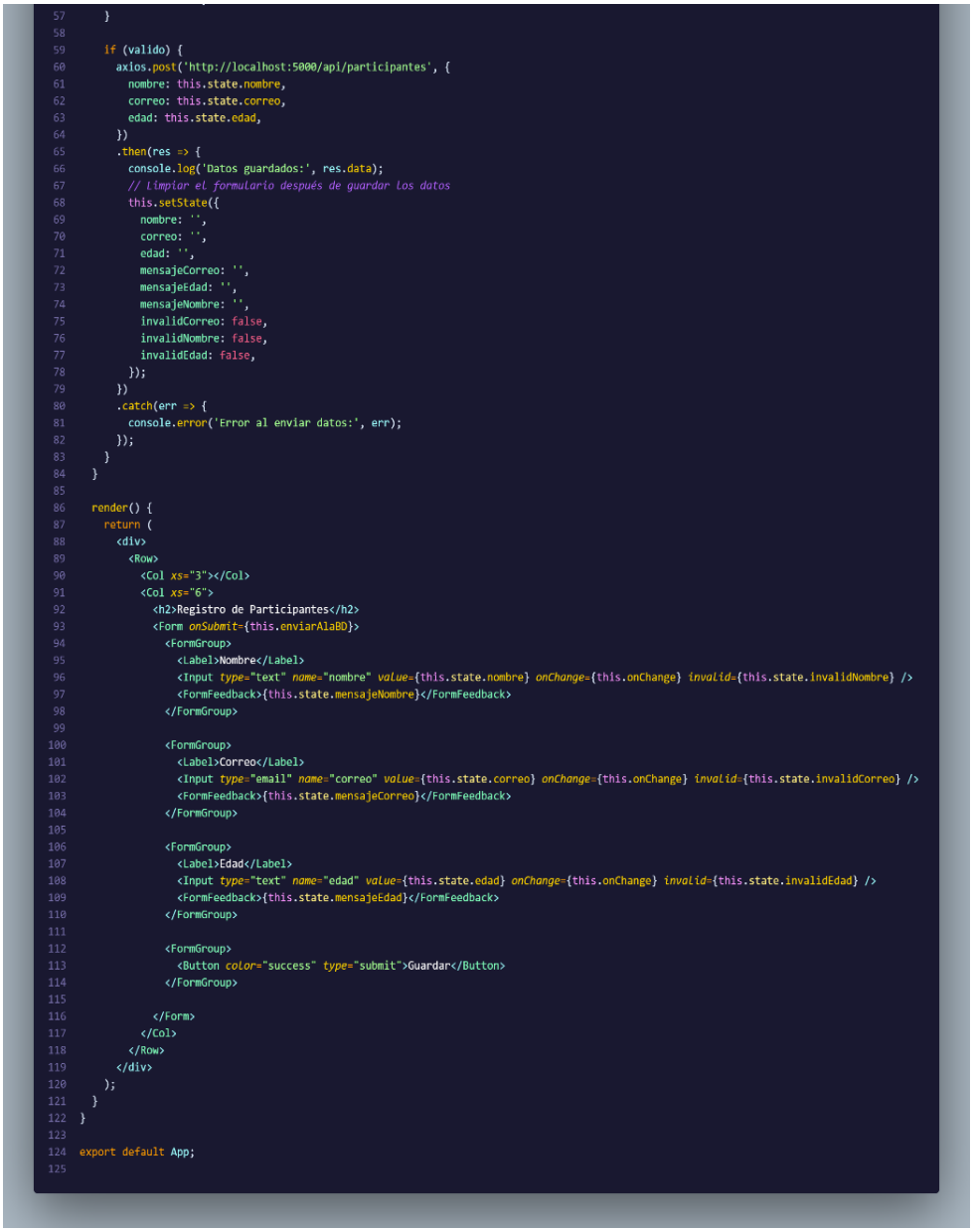
VII. Código de lógica de negocio.

Frontend

Este componente App proporciona una interfaz de usuario para registrar participantes. Incluye validación del formulario, manejo de eventos, envío de datos a un servidor backend usando Axios, y la capacidad de limpiar el formulario tras un envío exitoso. Las validaciones aseguran que todos los campos sean completados antes de enviar la solicitud al servidor. La interfaz está diseñada utilizando los componentes de reactstrap para una experiencia de usuario amigable y consistente.



```
import React, { Component } from 'react';
import axios from 'axios';
import { Row, Col, Form, Input, Label, FormGroup, Button, FormFeedback } from 'reactstrap';

class App extends Component {
  constructor(props) {
    super(props);

    this.state = {
      nombre: '',
      correo: '',
      edad: '',
      mensajeCorreo: '',
      mensajeEdad: '',
      mensajeNombre: '',
      invalidCorreo: false,
      invalidNombre: false,
      invalidEdad: false,
    };

    this.onChange = this.onChange.bind(this);
    this.enviarAlaBD = this.enviarAlaBD.bind(this);
  }

  onChange(e) {
    let name = e.target.name;
    let value = e.target.value;
    this.setState({
      [name]: value,
    });
  }

  enviarAlaBD(e) {
    e.preventDefault();
    let valido = true;

    if (this.state.nombre === '') {
      this.setState({
        invalidNombre: true,
        mensajeNombre: 'El campo nombre es obligatorio, indica tu nombre',
      });
      valido = false;
    }
    if (this.state.correo === '') {
      this.setState({
        invalidCorreo: true,
        mensajeCorreo: 'Indica tu dirección de correo',
      });
      valido = false;
    }
    if (this.state.edad === '') {
      this.setState({
        invalidEdad: true,
        mensajeEdad: 'Indica tu edad',
      });
      valido = false;
    }
```



```
    }

    if (valido) {
      axios.post('http://localhost:5000/api/participantes', {
        nombre: this.state.nombre,
        correo: this.state.correo,
        edad: this.state.edad,
      })
      .then(res => {
        console.log('Datos guardados:', res.data);
        // Limpiar el formulario después de guardar los datos
        this.setState({
          nombre: '',
          correo: '',
          edad: '',
          mensajeCorreo: '',
          mensajeEdad: '',
          mensajeNombre: '',
          invalidCorreo: false,
          invalidNombre: false,
          invalidEdad: false,
        });
      })
      .catch(err => {
        console.error('Error al enviar datos:', err);
      });
    }
  }

  render() {
    return (
      <div>
        <Row>
          <Col xs="3"></Col>
          <Col xs="6">
            <h2>Registro de Participantes</h2>
            <Form onSubmit={this.enviarAlaBD}>
              <FormGroup>
                <Label>Nombre</Label>
                <Input type="text" name="nombre" value={this.state.nombre} onChange={this.onChange} invalid={this.state.invalidNombre} />
                <FormFeedback>{this.state.mensajeNombre}</FormFeedback>
              </FormGroup>

              <FormGroup>
                <Label>Correo</Label>
                <Input type="email" name="correo" value={this.state.correo} onChange={this.onChange} invalid={this.state.invalidCorreo} />
                <FormFeedback>{this.state.mensajeCorreo}</FormFeedback>
              </FormGroup>

              <FormGroup>
                <Label>Edad</Label>
                <Input type="text" name="edad" value={this.state.edad} onChange={this.onChange} invalid={this.state.invalidEdad} />
                <FormFeedback>{this.state.mensajeEdad}</FormFeedback>
              </FormGroup>

              <FormGroup>
                <Button color="success" type="submit">Guardar</Button>
              </FormGroup>

            </Form>
          </Col>
        </Row>
      </div>
    );
  }
}

export default App;

```

Este archivo index.js configura el punto de entrada principal para la aplicación React. Importa las librerías y módulos necesarios, crea un contenedor (root) en el DOM para la aplicación, y renderiza el componente App en dicho contenedor. Además, incluye una configuración opcional para medir y registrar el rendimiento de la aplicación. La inclusión del CSS de Bootstrap permite utilizar estilos y componentes predefinidos para una interfaz de usuario profesional y consistente.

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';
import 'bootstrap/dist/css/bootstrap.min.css';


const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);

// If you want to start measuring performance in your app, pass a function
// to log results (for example: reportWebVitals(console.log))
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
reportWebVitals();

```

El archivo package.json define la configuración y las dependencias para el proyecto React denominado "tarea2.3". Incluye configuraciones esenciales para la gestión de dependencias, scripts para tareas comunes (como iniciar el servidor de desarrollo y construir la aplicación), y un proxy para redirigir solicitudes API al backend durante el desarrollo. Este archivo es fundamental para la administración y funcionamiento del proyecto.

```
{
    "name": "tarea2.3",
    "version": "1.0.0",
    "proxy": "http://localhost:5000",
    "dependencies": {
      "axios": "^0.21.1",
      "bootstrap": "^5.1.0",
      "reactstrap": "^8.9.0",
      "react": "^17.0.2",
      "react-dom": "^17.0.2",
      "react-scripts": "4.0.3"
    },
    "scripts": {
      "start": "react-scripts start",
      "build": "react-scripts build",
      "test": "react-scripts test",
      "eject": "react-scripts eject"
    }
  }

```

Backend

Este archivo configura un servidor backend básico usando Node.js y Express. El servidor maneja solicitudes para obtener y agregar participantes a un archivo JSON (data.json). Incluye una ruta GET para recuperar la lista de participantes y una ruta POST para agregar nuevos participantes. El servidor

se ejecuta en el puerto 5000 y utiliza middleware para manejar cuerpos de solicitudes JSON. La gestión de archivos se realiza mediante los módulos fs y path para leer y escribir datos en el archivo JSON.

```
const express = require('express');
const fs = require('fs');
const path = require('path');
const app = express();
const port = 5000; // Puerto en el que el servidor backend estará escuchando

app.use(express.json()); // Para manejar JSON en el cuerpo de las solicitudes

// Ruta para obtener los participantes
app.get('/api/participantes', (req, res) => {
  fs.readFile(path.join(__dirname, 'data.json'), 'utf8', (err, data) => {
    if (err) {
      return res.status(500).send('Error al leer los datos.');
    }
    res.json(JSON.parse(data));
  });
});

// Ruta para agregar un nuevo participante
app.post('/api/participantes', (req, res) => {
  const { nombre, correo, edad } = req.body;

  fs.readFile(path.join(__dirname, 'data.json'), 'utf8', (err, data) => {
    if (err) {
      return res.status(500).send('Error al leer los datos.');
    }

    const participantes = JSON.parse(data);
    participantes.push({ nombre, correo, edad });

    fs.writeFile(path.join(__dirname, 'data.json'), JSON.stringify(participantes, null, 2), (err) => {
      if (err) {
        return res.status(500).send('Error al guardar los datos.');
      }
      res.status(201).json({ nombre, correo, edad });
    });
  });
});

app.listen(port, () => {
  console.log(`Servidor backend corriendo en http://localhost:${port}`);
});
```

El archivo reportWebVitals.js define y exporta la función reportWebVitals, que se utiliza para medir y registrar métricas clave de rendimiento de una aplicación React. Utiliza el módulo web-vitals para capturar métricas como el Cumulative Layout Shift (CLS), el First Input Delay (FID), el First Contentful Paint (FCP), el Largest Contentful Paint (LCP) y el Time to First Byte (TTFB). La función reportWebVitals se invoca con una función de manejo proporcionada, que puede registrar o enviar estas métricas a un servicio de análisis para evaluar el rendimiento de la aplicación.

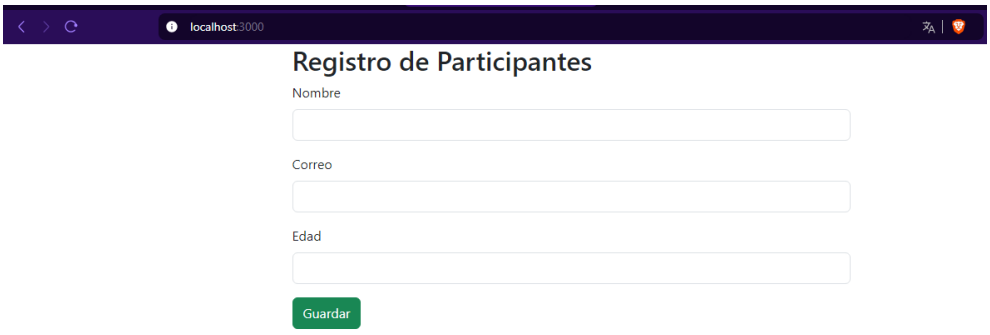
```
const reportWebVitals = onPerfEntry => {
  if (onPerfEntry && onPerfEntry instanceof Function) {
    import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) => {
      getCLS(onPerfEntry);
      getFID(onPerfEntry);
      getFCP(onPerfEntry);
      getLCP(onPerfEntry);
      getTTFB(onPerfEntry);
    });
  }
};

export default reportWebVitals;
```

El fragmento JSON proporciona una representación estructurada de la información de un participante, incluyendo su nombre, correo electrónico y edad. Está diseñado en un formato que puede ser fácilmente interpretado y manipulado por aplicaciones web y servidores backend.

```
[
    {
    "nombre": "John Diaz",
    "correo": "john@gmail.com",
    "edad": 30
    }
]
```

VIII. Navegación

Se evidencia la funcionalidad del formulario



De igual manera el registro de los usuarios ingresados a la data.json