1. Configuración del Proyecto

1.1 Crear la carpeta del proyecto

1. Abre tu terminal y navega a la carpeta deseada:

```
sh
Copiar código
cd C:\Users\avalt
```

2. Crea la carpeta del proyecto:

```
sh
Copiar código
mkdir p_node
cd p_node
```

1.2 Inicializar el proyecto Node.js

3. Inicializa un nuevo proyecto Node.js:

```
sh
Copiar código
npm init -y
```

2. Instalación de Dependencias

2.1 Dependencias para el backend

4. Instala las dependencias necesarias para trabajar con archivos PDF y convertirlos a XML:

```
sh
Copiar código
npm install express pdf-parse xmlbuilder
```

2.2 Dependencias para el frontend

5. Crea la carpeta para el frontend:

```
sh
Copiar código
npx create-react-app frontend
```

3. Crear el Servidor Backend

3.1 Configurar el servidor Express

6. Crea un archivo `server.js` en la raíz del proyecto con el siguiente contenido:

```
js
Copiar código
const express = require('express');
```

```
const fileUpload = require('express-fileupload');
const pdf = require('pdf-parse');
const { create } = require('xmlbuilder2');

const app = express();
const port = 3001;

app.use(fileUpload());

app.post('/upload', async (req, res) => {
    if (!req.files || !req.files.pdf) {
        return res.status(400).send('No files were uploaded.');
    }

    const pdfFile = req.files.pdf;
    const data = await pdf(pdfFile.data);

    const xml = create({ version: '1.0' })
        .ele('root')
        .ele('text')
        .txt(data.text)
        .up()
        .end({ prettyPrint: true });

    res.set('Content-Type', 'application/xml');
    res.send(xml);
});

app.listen(port, () => {
    console.log(`Server running at http://localhost:${port}`);
});
```

3.2 Instalar express-fileupload

7. Instala `express-fileupload` para manejar la subida de archivos:

```
sh
Copiar código
npm install express-fileupload
```

4. Configurar el Frontend con React

4.1 Crear el componente para la carga de archivos

8. Navega a la carpeta del frontend y crea un componente para cargar archivos:

```
sh
Copiar código
cd frontend/src
```

9. Crea un archivo `Upload.js` con el siguiente contenido:

```
jsx
Copiar código
import React, { useState } from 'react';
import axios from 'axios';

function Upload() {
```

```
    const [file, setFile] = useState(null);
    const [xmlData, setXmlData] = useState('');

    const handleFileChange = (e) => {
        setFile(e.target.files[0]);
    };

    const handleUpload = async () => {
        const formData = new FormData();
        formData.append('pdf', file);

        const response = await
axios.post('http://localhost:3001/upload', formData, {
            headers: {
                'Content-Type': 'multipart/form-data',
            },
        });

        setXmlData(response.data);
    };

    return (
        <div>
            <input type="file" onChange={handleFileChange} />
            <button onClick={handleUpload}>Upload</button>
            <pre>{xmlData}</pre>
        </div>
    );
}

export default Upload;
```

4.2 Incluir el componente en la aplicación principal

10. Abre `App.js` y reemplaza el contenido con el siguiente:

```
jsx
Copiar código
import React from 'react';
import Upload from './Upload';
import './App.css';

function App() {
    return (
        <div className="App">
            <header className="App-header">
                <h1>PDF to XML Converter</h1>
            </header>
            <Upload />
        </div>
    );
}

export default App;
```

5. Ejecutar el Proyecto

5.1 Ejecutar el backend

11. En la raíz del proyecto, ejecuta el servidor Node.js:

```
sh
Copiar código
node server.js
```

5.2 Ejecutar el frontend

12. Navega a la carpeta del frontend y ejecuta la aplicación React:

```
sh
Copiar código
cd frontend
npm start
```