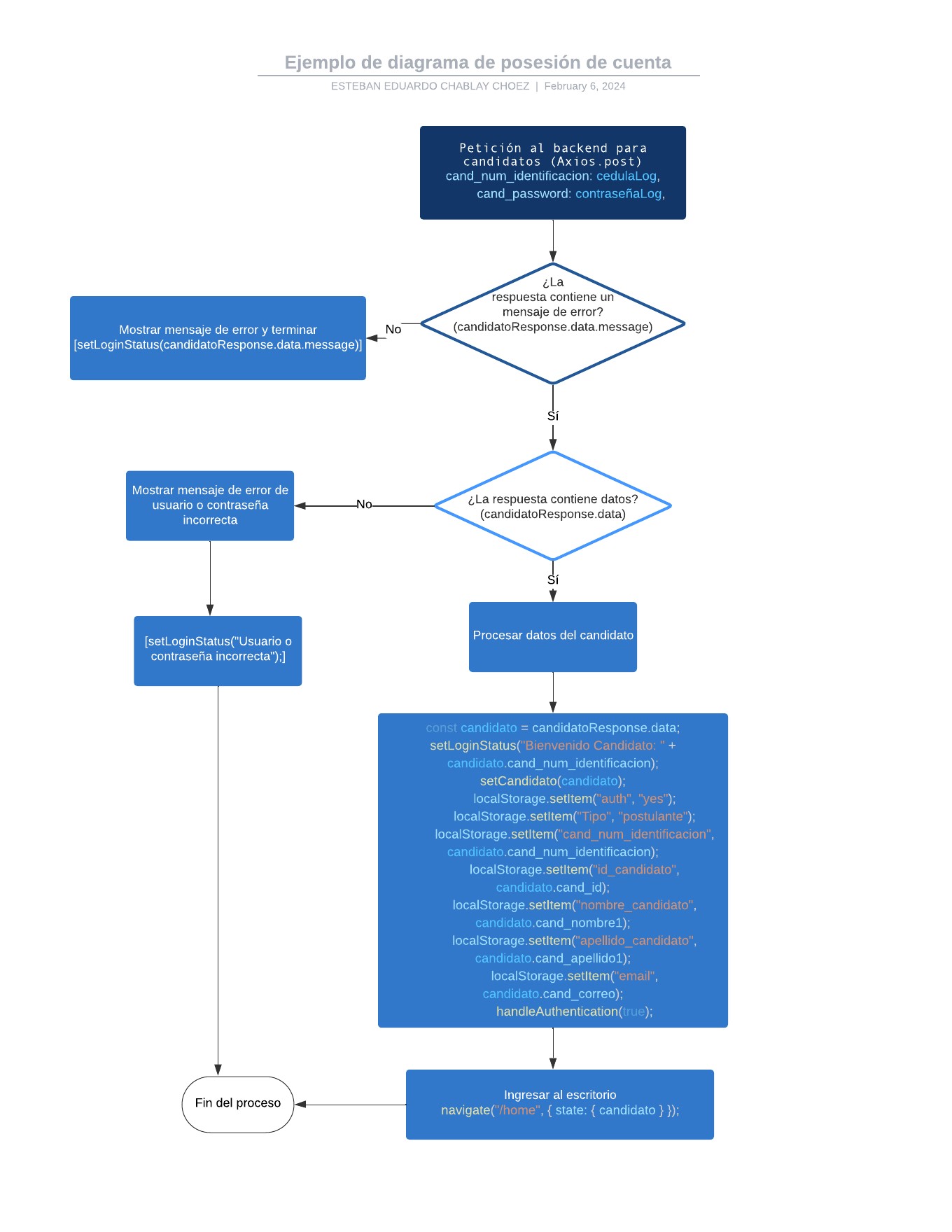
|  |
| --- |
|  |
| Prueba de Caja Blanca |
| ***“Sistema de Postulación - ESPE”*** |
| Versión 1.1 |
|  |
| **Integrantes:**  Esteban Chablay Adrian Mosquera Fernando Paredes  Santiago Risueño  **Fecha 06/02/2024** |

**BUSCAR PROBLEMAS**

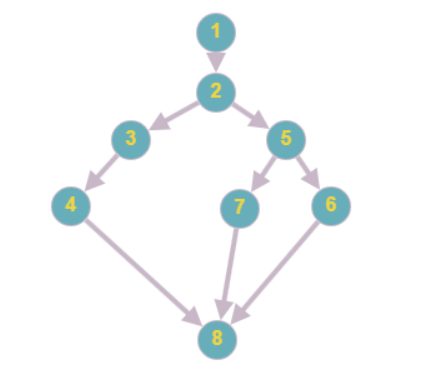
**CÓDIGO FUENTE**

|  |
| --- |
| const handleEnviarClick = () => {      // Check if the user has uploaded all the required documents here      // For simplicity, let's assume that all documents are required      const areAllDocumentsUploaded = sheetsCount.every((sheets) => sheets > 0);      if (areAllDocumentsUploaded) {        // Show the confirmation modal if all documents are uploaded        setShowConfirmModal(true);      } else {        // Display an error message if any document is missing        alert("Por favor, suba todos los documentos requeridos antes de enviar.");      }    };    const [sheetsCount, setSheetsCount] = useState(Array(8).fill(0));    const [linkPDF, setLinkPDF] = useState(Array(8).fill(0));    const handleFileChange = async (index, event) => {      const file = event.target.files[0];      if (file) {        const formData = new FormData();        formData.append("file", file);        formData.append("username", "Usuario");        formData.append("tipoDocumento", documentLabels[index]);        try {          const response = await axios.post("http://localhost:5000/api/upload", formData, {            headers: {              "Content-Type": "multipart/form-data",            },          });          const { numPages } = response.data;          const updatedSheetsCount = [...sheetsCount];          const updatedLinkPDF = [...linkPDF];          updatedSheetsCount[index] = numPages;          updatedLinkPDF[index] = response.data.url;          setSheetsCount(updatedSheetsCount);          setLinkPDF(updatedLinkPDF);          console.log("PDF subido a la Base de Datos");        } catch (error) {          console.error("Error uploading file:", error);        }      }    };    return (      <Container>        <h1>Subir Documentos</h1>        <Form>          {documentLabels.map((label, index) => (            <div key={index}>              <label htmlFor={`file${index + 1}`}>{label}:</label>              <input                type="file"                id={`file${index + 1}`}                onChange={(e) => handleFileChange(index, e)}              />              {/\*<a href={linkPDF[index]} target="\_blank" rel="noreferrer"><button onClick={(e) => e.preventDefault()}>📄 Previsualizar</button></a>\*/}              {sheetsCount[index] > 0 && <p>Cantidad de Páginas: {sheetsCount[index]}</p>}            </div>          ))}          {sheetsCount.reduce((a, b) => a + b, 0) > 0 && <h3>Conteo Acumulado de Páginas: {sheetsCount.reduce((a, b) => a + b, 0)}</h3>}          <button type="button" onClick={handleEnviarClick}>            Enviar          </button>        </Form> |

**Diagrama de Flujo**



**GRAFO**

****

**NODOS:**

1. Inicio
2. Cargar documentos
3. Mostrar mensaje de la cantidad de páginas del documentos
4. Fin

**RUTAS**

**R1:** 1 -> 2 -> 3 -> 4

**COMPLEJIDAD CICLOMÁTICA**

La complejidad ciclomática se calcula como E - N + 2P, donde:

* E es el número de aristas
* N es el número de nodos
* P es el número de componentes conectados

Para este grafo:

* E = 3 (número de aristas)
* N = 4 (número de nodos)
* P = 1 (un solo componente conectado)

Por lo tanto, la complejidad ciclomática es:

C = 3 - 4 + 2\*1 = 1