



UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
INGENIERÍA DE SOFTWARE

NRC: 15035

Materia: Análisis y Diseño de software

Tema: Control del inicio de sesión en el sistema tiendita

De La Cadena Moncayo Leonardo Javier

Morales Noroña Susana Camila

Morales Pisco Johao Alejandro

Tello Martinez Cristian Andres

Ing. Jenny Alexandra Ruiz Robalino

Sangolquí, 18 de Febrero del 2024

LOGIN

CÓDIGO FUENTE

```
const [username, setUsername] = useState("");
const [password, setPassword] = useState("");
const [isLoggedIn, setIsLoggedIn] = useState(false);

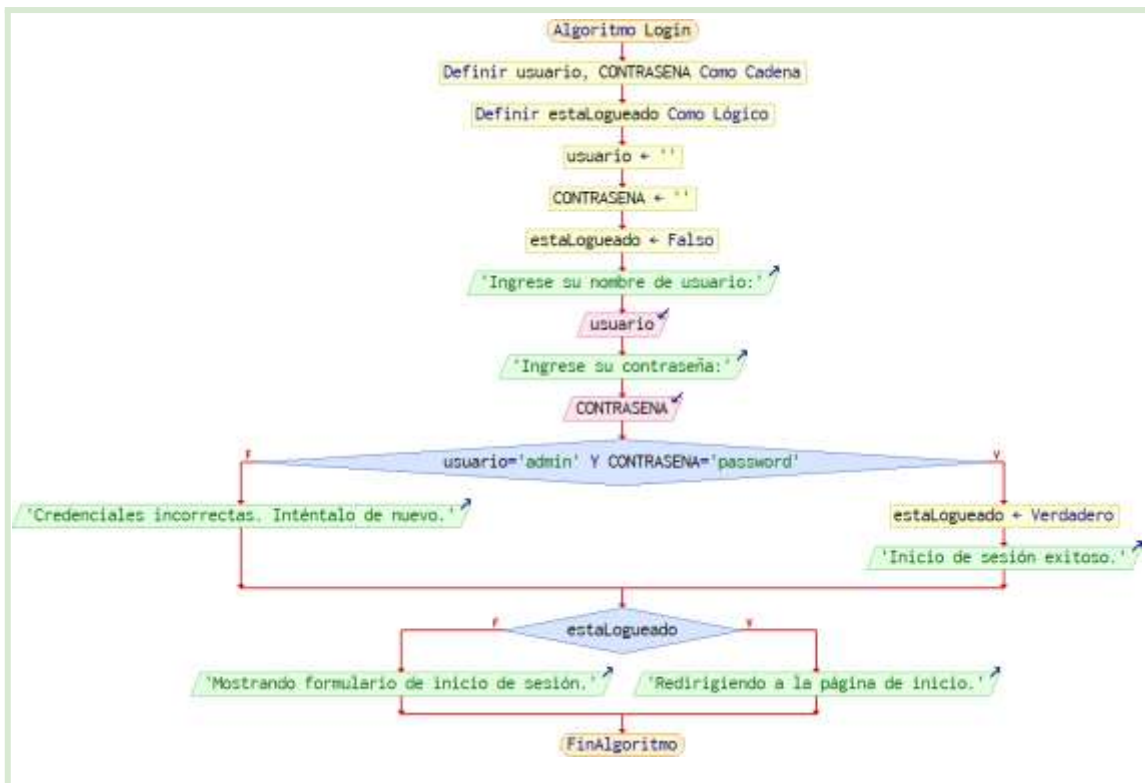
const handleInputChange = (event) => {
  const { name, value } = event.target;
  if (name === "username") {
    setUsername(value);
  } else if (name === "password") {
    setPassword(value);
  }
};

const handleLogin = (event) => {
  event.preventDefault();

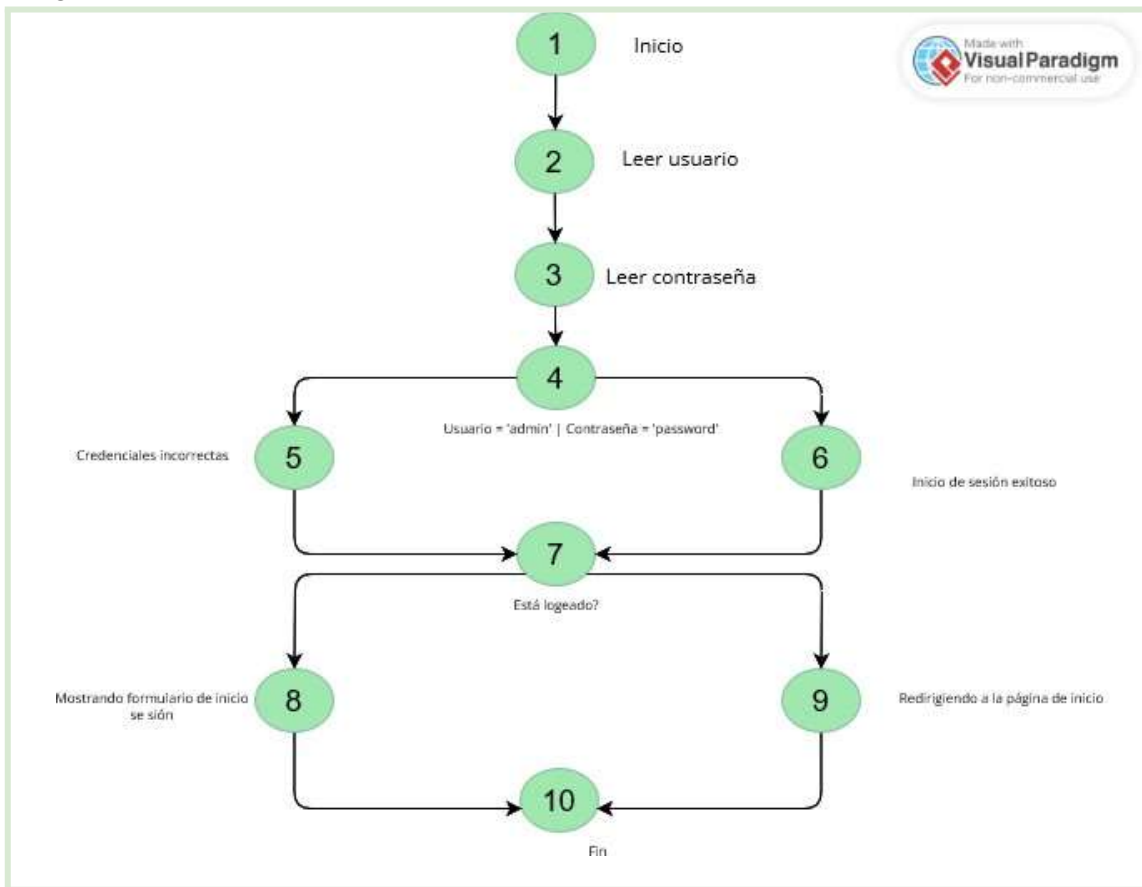
  if (username === "admin" && password === "password") {
    setIsLoggedIn(true);
    localStorage.setItem("username", username);
  } else {
    alert("Credenciales incorrectas. Inténtalo de nuevo.");
  }
};

if (isLoggedIn) {
  // Si el usuario ha iniciado sesión, utiliza Navigate para redirigir a /registro-cliente
  return <Navigate to="/home" />;
} else {
  // Si el usuario aún no ha iniciado sesión, muestra el formulario de inicio de sesión
}
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 4,5,7,8,10

R2: 1, 2, 3, 4,6,7,9,10

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G) = \text{número de nodos predichados(decisiones)} + 1$
 $V(G) = 2 + 1 = 3$
- $V(G) = A - N + 2$
 $V(G) = 11 - 10 + 2 = 3$

DONDE:

P: Número de nodos predichado

A: Número de aristas

N: Número de nodos

AGREGAR NUEVO CLIENTE

CÓDIGO FUENTE

```
//INICIO DEL COMPONENTE
const RegistroCliente = () => {
  // Estado para controlar si se muestra el modal o no
  const [showModal, setShowModal] = useState(false);
  const [showEditModal, setEditShowModal] = useState(false);
  //ESTADO PARA CARGAR DEL FORMULARIO EL INDICE DE BUSQUEDA POR CEDULA
  const [searchTerm, setSearchTerm] = useState("");
```

```
//AGREGAMOS ESTADOS PARA CARGAR LOS CLIENTES
const [clientes, setClientes] = useState([]);
//ESTADOS PARA LOS DATOS
const { id } = useParams();
const [Cedula, setCedula] = useState("");

const [PrimerApellido, setPrimerApellido] = useState("");
const [SegundoApellido, setSegundoApellido] = useState("");
const [PrimerNombre, setPrimerNombre] = useState("");
const [SegundoNombre, setSegundoNombre] = useState("");
const [Email, setEmail] = useState("");
const [Contacto, setContacto] = useState("");
const [Direccion, setDireccion] = useState("");
const [FecNacimiento, setFecNacimiento] = useState("");
const [Estado, setEstado] = useState("");
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G) = \text{número de nodos predicados(decisiones)} + 1$
 $V(G) = 0 + 1 = 1$
- $V(G) = A - N + 2$
 $V(G) = 13 - 14 + 2 = 1$

EDITAR UN CLIENTE

CÓDIGO FUENTE

```
const update = async (e) => {
  e.preventDefault();
  if (!verificarCedula(Cedula)) {
    alert("Cédula no válida");
    return;
  }

  const dob = new Date(FecNacimiento);
  const today = new Date();

  if (dob > today) {
    alert('La fecha de nacimiento no es valida.');
```

return;

}

let age = today.getFullYear() - dob.getFullYear();

if (today.getMonth() < dob.getMonth() || (today.getMonth() === dob.getMonth() && today.getDate() < dob.getDate())) {

age--;

}

if (age < 18) {

alert('El cliente debe tener al menos 18 años.');

return;

}

try {

const idAux = localStorage.getItem("id");

await axios.put(URI + idAux, {

CLI_CEDULA: Cedula,

CLI_PRIMERAPELLIDO: PrimerApellido,

CLI_SEGUNDOAPELLIDO: SegundoApellido,

CLI_PRIMERNOMBRE: PrimerNombre,

CLI_SEGUNDONOMBRE: SegundoNombre,

CLI_EMAIL: Email,

CLI_CONTACTO: Contacto,

CLI_DIRECCION: Direccion,

CLI_FECNACIMIENTO: FecNacimiento,

CLI_ESTADO: Estado,

});

setEditShowModal(false);

getClientes();

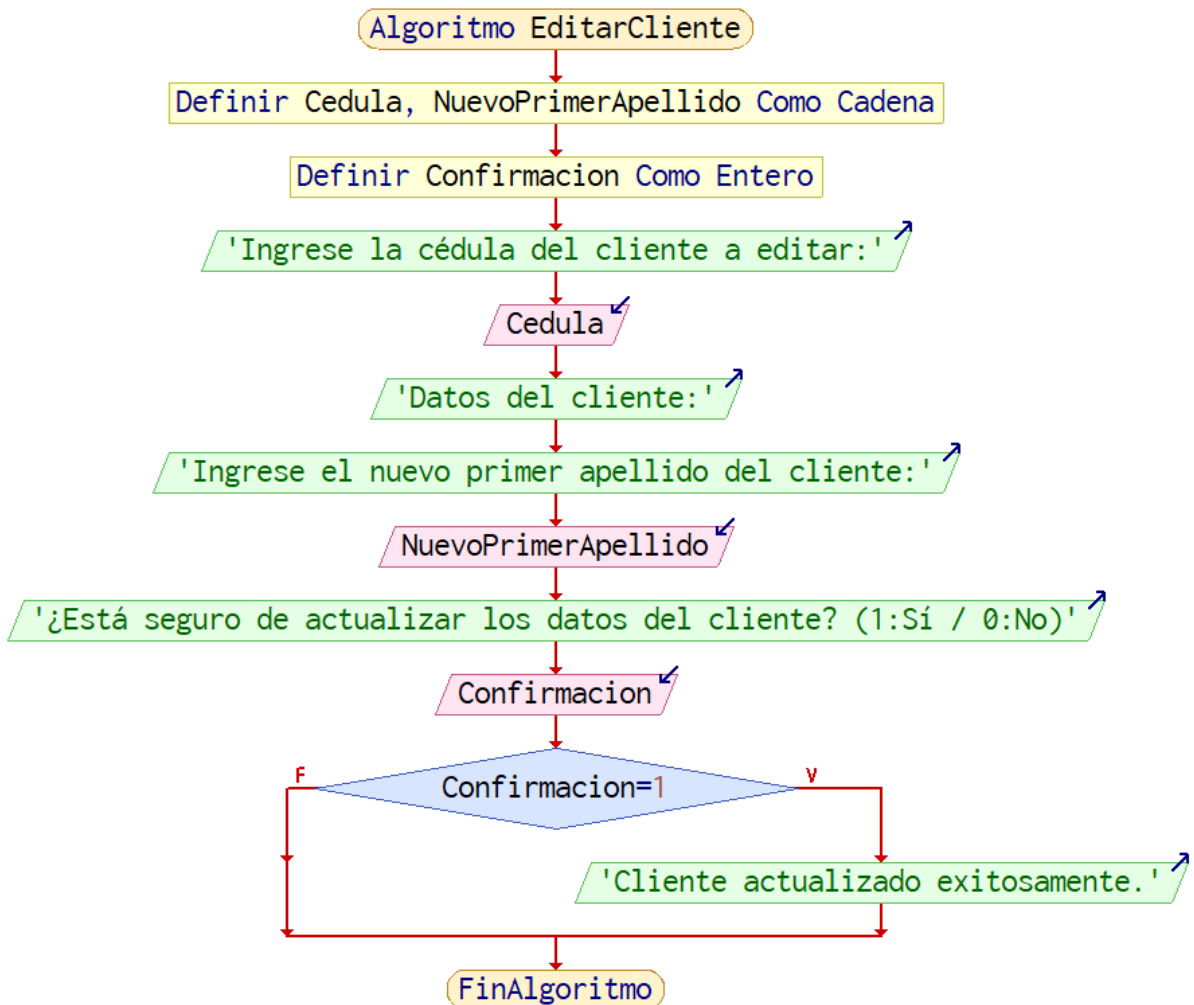
} catch (error) {

```

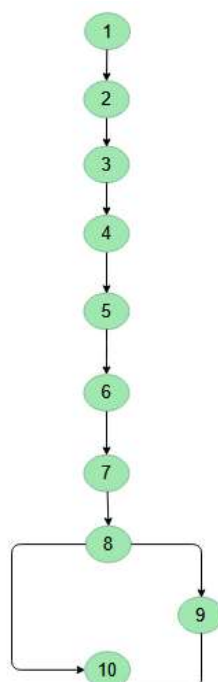
    console.error("Error al Actualizar Datos del Cliente:", error);
  }
};

```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 4, 5, 6, 7, 8.

R2: 1, 2, 3, 4, 5, 6, 7, 9.

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G) = \text{número de nodos predichados(decisiones)} + 1$
 $V(G) = 1 + 1 = 1$
- $V(G) = A - N + 2$
 $V(G) = 10 - 10 + 2 = 1$

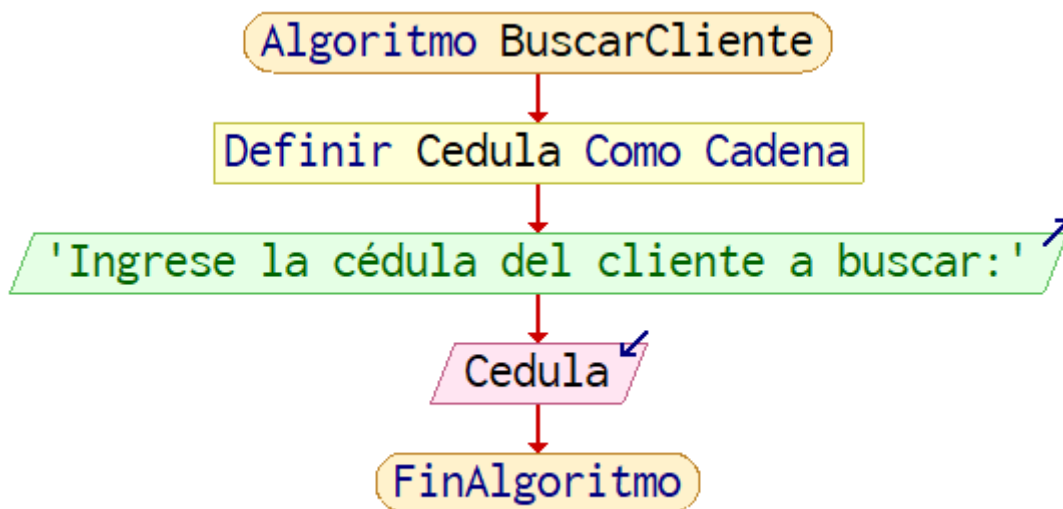
BUSCAR CLIENTE

CÓDIGO FUENTE

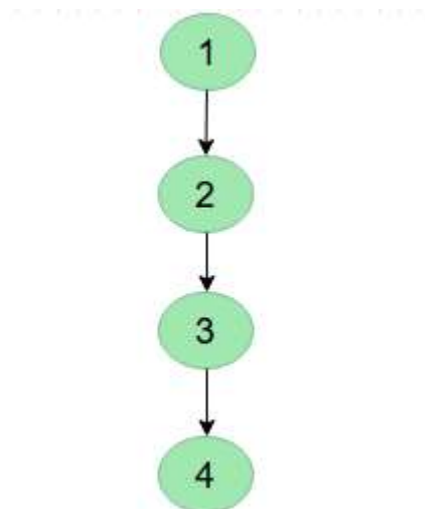
```
const getClienteById = async () => {
  try {
    var idAux = localStorage.getItem("id");
    const res = await axios.get(URI + idAux);

    if (res.data.length > 0) {
      setCedula(res.data[0].CLI_CEDULA);
      setPrimerApellido(res.data[0].CLI_PRIMERAPELLIDO);
      setSegundoApellido(res.data[0].CLI_SEGUNDOAPELLIDO);
      setPrimerNombre(res.data[0].CLI_PRIMERNOMBRE);
      setSegundoNombre(res.data[0].CLI_SEGUNDONOMBRE);
      setEmail(res.data[0].CLI_EMAIL);
      setContacto(res.data[0].CLI_CONTACTO);
      setDireccion(res.data[0].CLI_DIRECCION);
      const formattedDate = new Date(res.data[0].CLI_FECNACIMIENTO)
        .toISOString()
        .split("T")[0];
      setFecNacimiento(formattedDate);
      console.log(res.data[0].CLI_FECNACIMIENTO);
      setEstado(res.data[0].CLI_ESTADO);
    } else {
      // Si no se encuentra ningún cliente con el id proporcionado
      console.log("No se encontró ningún cliente con el ID: " + idAux);
    }
  } catch (error) {
    // Manejo de errores
    console.error("Error al obtener los datos del cliente:", error);
  }
};
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: R1: 1, 2, 3, 4

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G) = \text{número de nodos predicaos(decisiones)} + 1$
 $V(G) = 0 + 1 = 1$
- $V(G) = A - N + 2$
 $V(G) = 3 - 4 + 2 = 1$

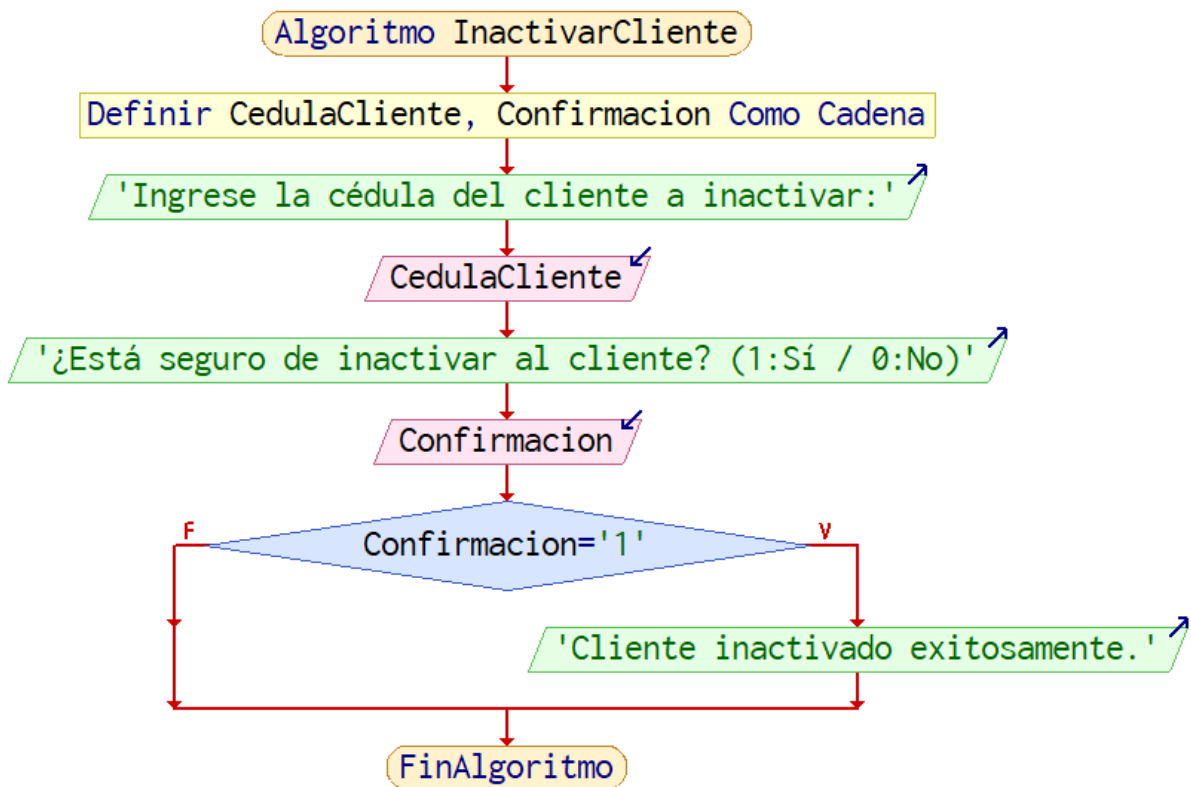
INACTIVAR CLIENTE

CÓDIGO FUENTE

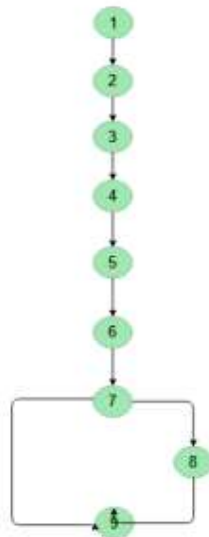
```

const deleteClientes = async (id) => {
  await axios.put(URI + id, {
    CLI_ESTADO: "Inactivo",
  });
  getClientes();
};
  
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 4, 5, 6, 7, 8.

R2: 1, 2, 3, 4, 5, 6, 7, 9.

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G) = \text{número de nodos predicados(decisiones)} + 1$
 $V(G) = 1 + 1 = 2$
- $V(G) = A - N + 2$
 $V(G) = 9 - 9 + 2 = 2$

AGREGAR NUEVO PRODUCTO

CÓDIGO FUENTE

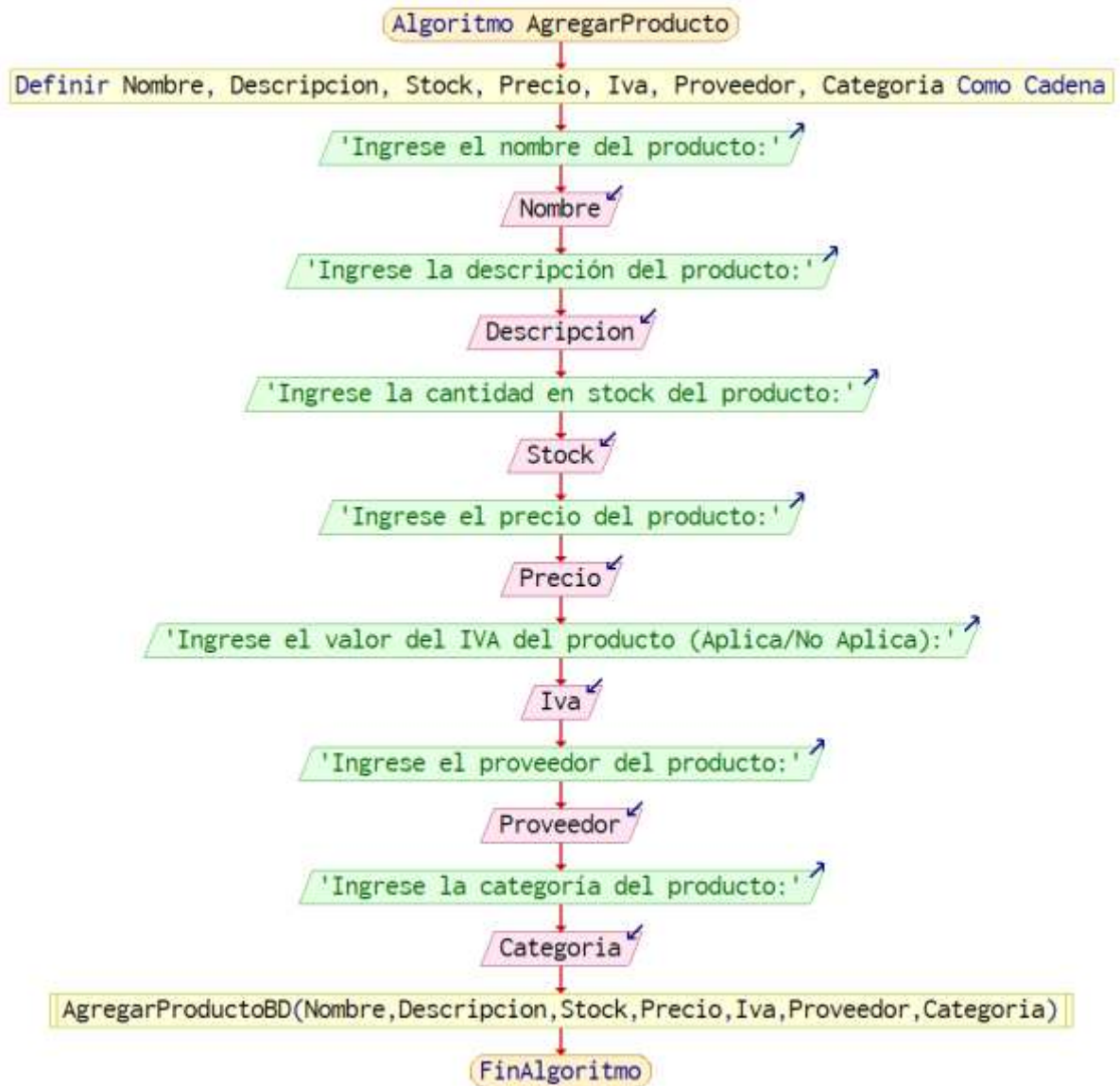
```
const nuevoProducto = async (e) => {
  e.preventDefault();
  await axios.post(URI2, {
    PRO_NOMBRE: Nombre,
    PROV_ID: Provid,
    CAT_ID: Catid,
    PRO_DESCRIPCION: Descripcion,
    PRO_STOCK: 0,
    PRO_PRECIO: Precio,
    PRO_IVA: Iva,
    PRO_ESTADO: "Activo",
  });
  setShowModal(false);
  getProductos();

  }; const [ProvNombre, setProvNombre] = useState("");
//DATOS DE LA CATEGORIA
const [CatNombre, setCatNombre] = useState("");
const [Catid, setCatid] = useState("");
//DATOS PARA INGRESAR EL INVENTARIO
const [Cantidad, setCantidad] = useState("");

const navigate = useNavigate();
const { id } = useParams();

useEffect(() => {
  getProductos();
}, []);
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

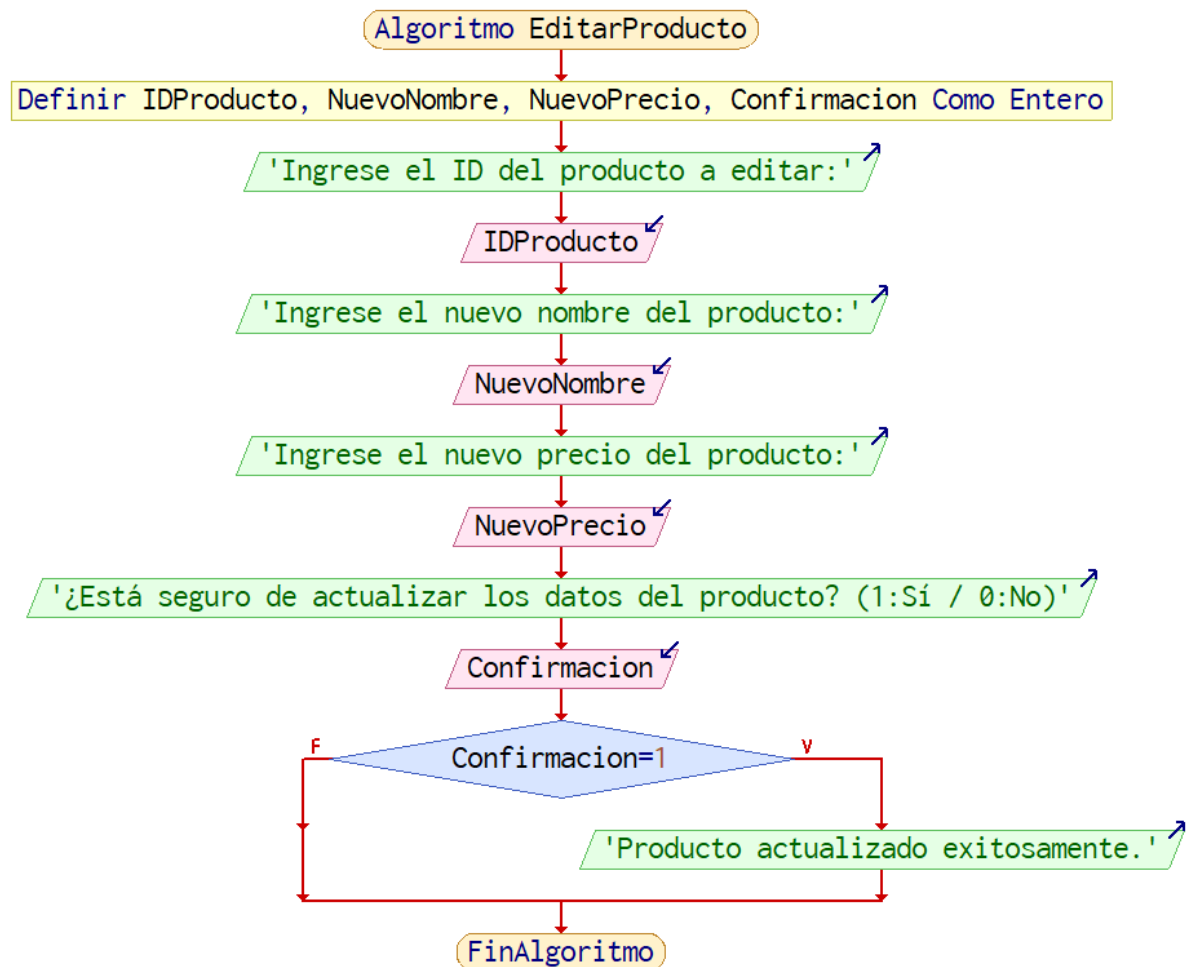
- $V(G) = \text{número de nodos predicados(decisiones)} + 1$
 $V(G) = 0 + 1 = 1$
- $V(G) = A - N + 2$
 $V(G) = 10 - 11 + 2 = 1$

EDITAR PRODUCTO

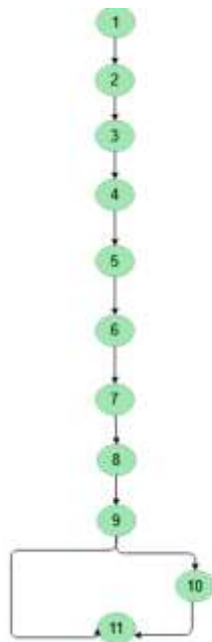
CÓDIGO FUENTE

```
const update = async (e) => {  
  e.preventDefault();  
  var idAux = localStorage.getItem("id");  
  await axios.put(URI2 + idAux, {  
    PRO_NOMBRE: Nombre,  
    PROV_ID: Provid,  
    CAT_ID: Catid,  
    PRO_DESCRIPCION: Descripcion,  
    PRO_STOCK: Stock,  
    PRO_PRECIO: Precio,  
    PRO_IVA: Iva,  
    PRO_ESTADO: Estado,  
  });  
  setEditShowModal(false);  
  getProductos();  
};  
  
useEffect(() => {  
  getProductoById();  
}, []);
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11.

R2: 1, 2, 3, 4, 5, 6, 7, 8, 9.

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G) = \text{número de nodos predichados(decisiones)} + 1$
 $V(G) = 1 + 1 = 2$
- $V(G) = A - N + 2$
 $V(G) = 11 - 11 + 2 = 2$

BUSCAR PRODUCTO

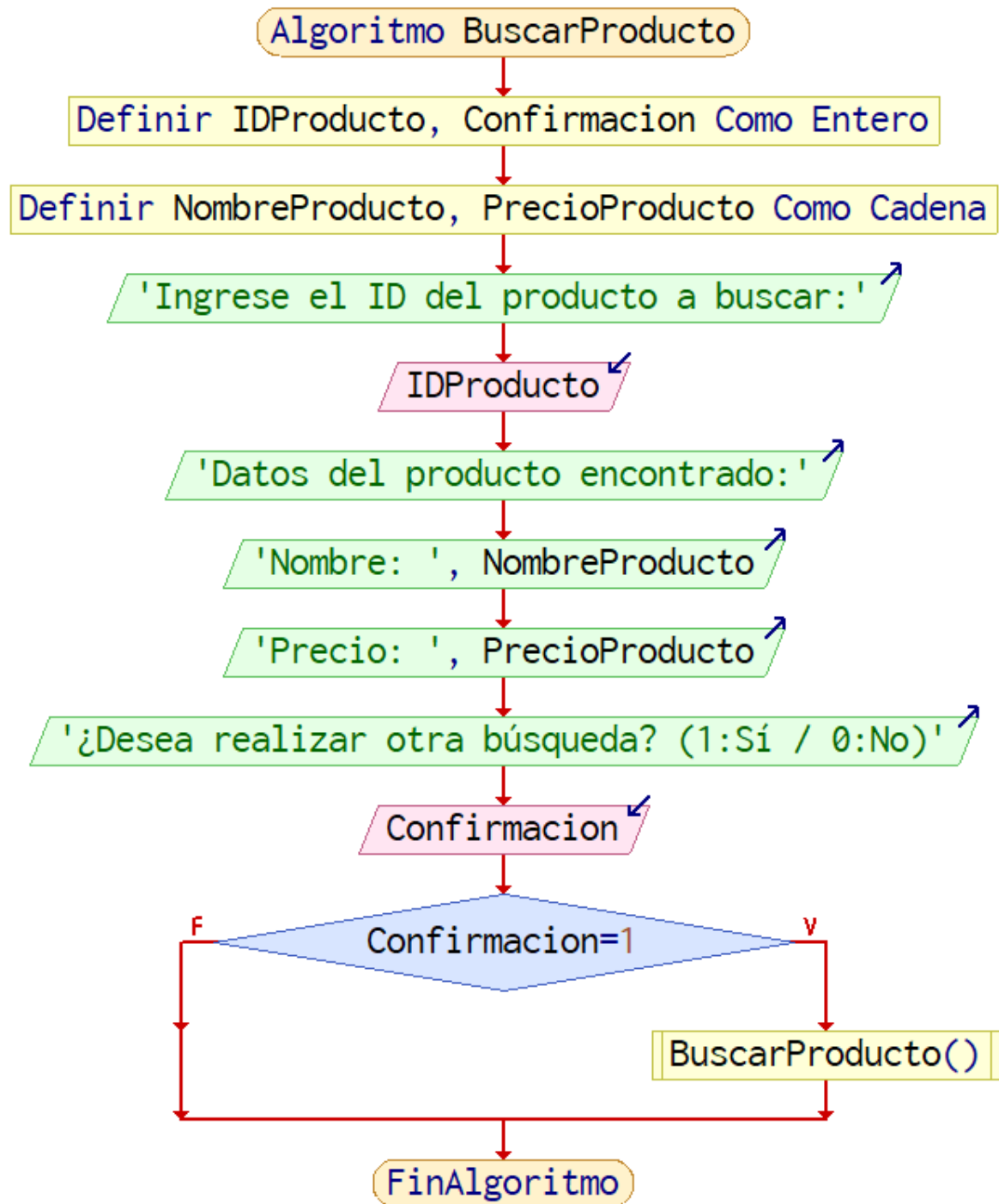
CÓDIGO FUENTE

```
const getProductoById = async () => {
  try {
    var idAux = localStorage.getItem("id");
    const res = await axios.get(URI2 + idAux);

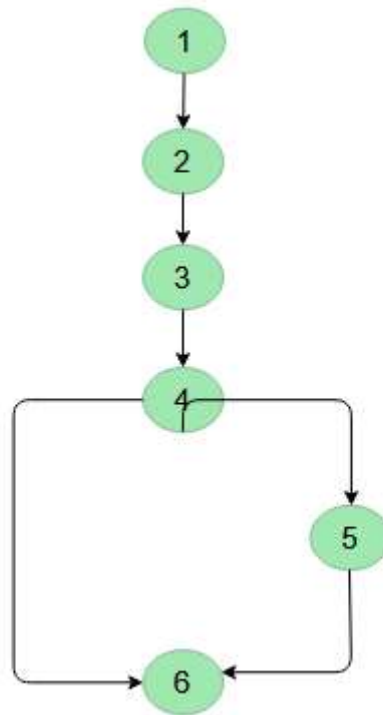
    if (res.data.length > 0) {
      setNombre(res.data[0].PRO_NOMBRE);
      setProvid(res.data[0].PROV_ID);
      setCatid(res.data[0].CAT_ID);
      setDescripcion(res.data[0].PRO_DESCRIPCION);

      setStock(res.data[0].PRO_STOCK);
      setPrecio(res.data[0].PRO_PRECIO);
      setIva(res.data[0].PRO_IVA);
      setEstado(res.data[0].PRO_ESTADO);
    } else {
      // Si no se encuentra ningún producto con el id proporcionado
      console.log("No se encontró ningún producto con el ID: " + idAux);
    }
  } catch (error) {
    // Manejo de errores
    console.error("Error al obtener los datos del producto:", error);
  }
}
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 4, 5, 6.

R2: 1, 2, 3, 4, 5.

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G) = \text{número de nodos predcados(decisiones)} + 1$
 $V(G) = 1 + 1 = 2$
- $V(G) = A - N + 2$
 $V(G) = 6 - 6 + 2 = 2$

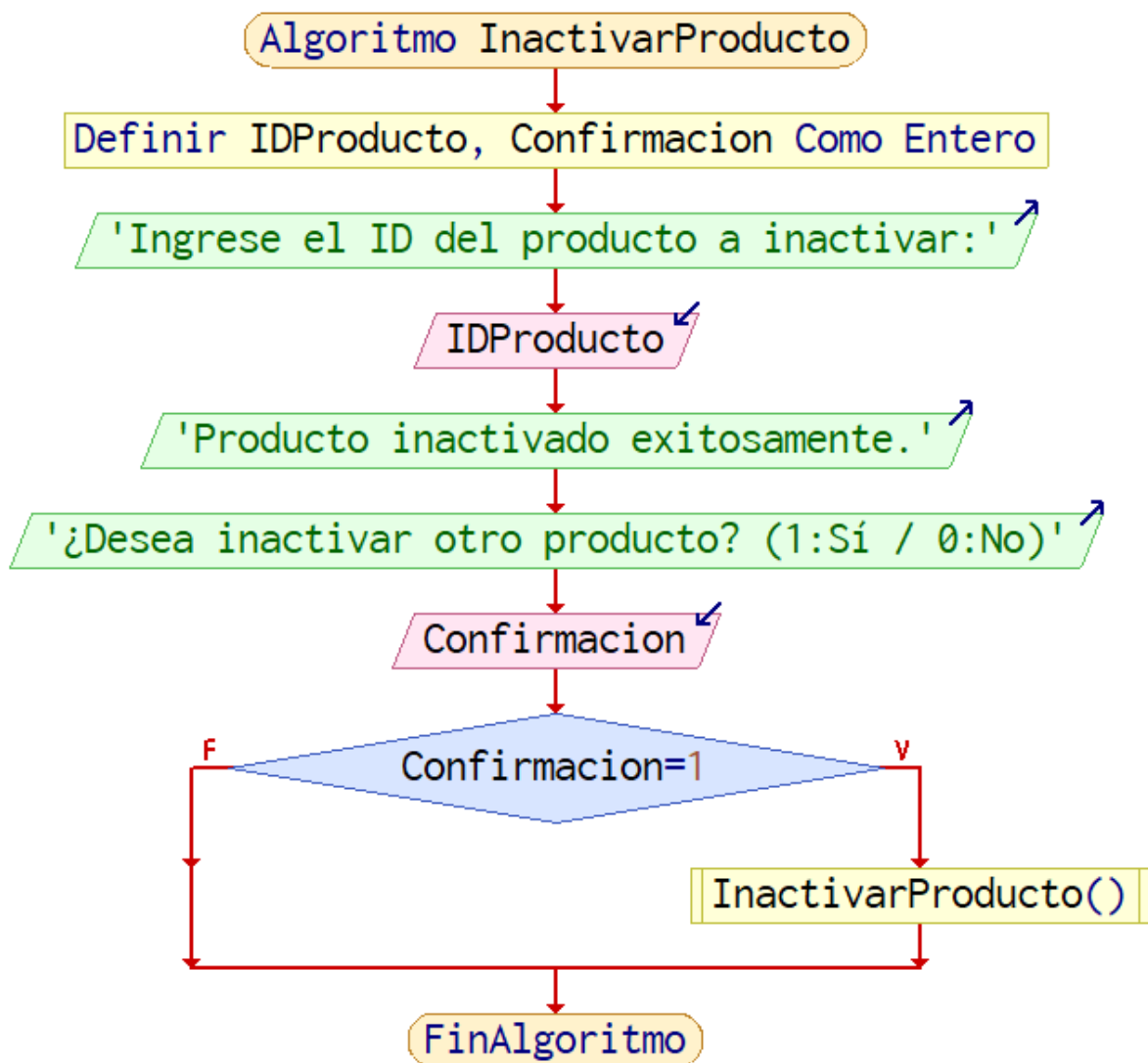
INACTIVAR PRODUCTO

CÓDIGO FUENTE

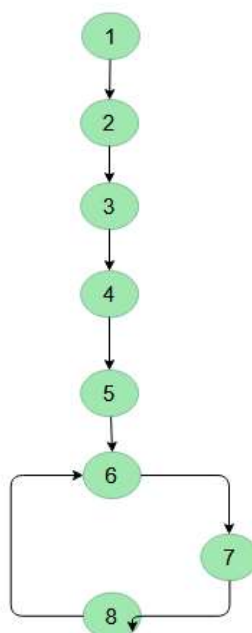
```

const deleteProductos = async (id) => {
  await axios.put(URI2 + id, {
    PRO_ESTADO: "Inactivo",
  });
  getProductos();
};
  
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 4, 5, 6, 7, 8.

R2: 1, 2, 3, 4, 5, 6, 7.

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G) = \text{número de nodos predichados(decisiones)} + 1$
 $V(G) = 1 + 1 = 2$
- $V(G) = A - N + 2$
 $V(G) = 8 - 8 + 2 = 2$

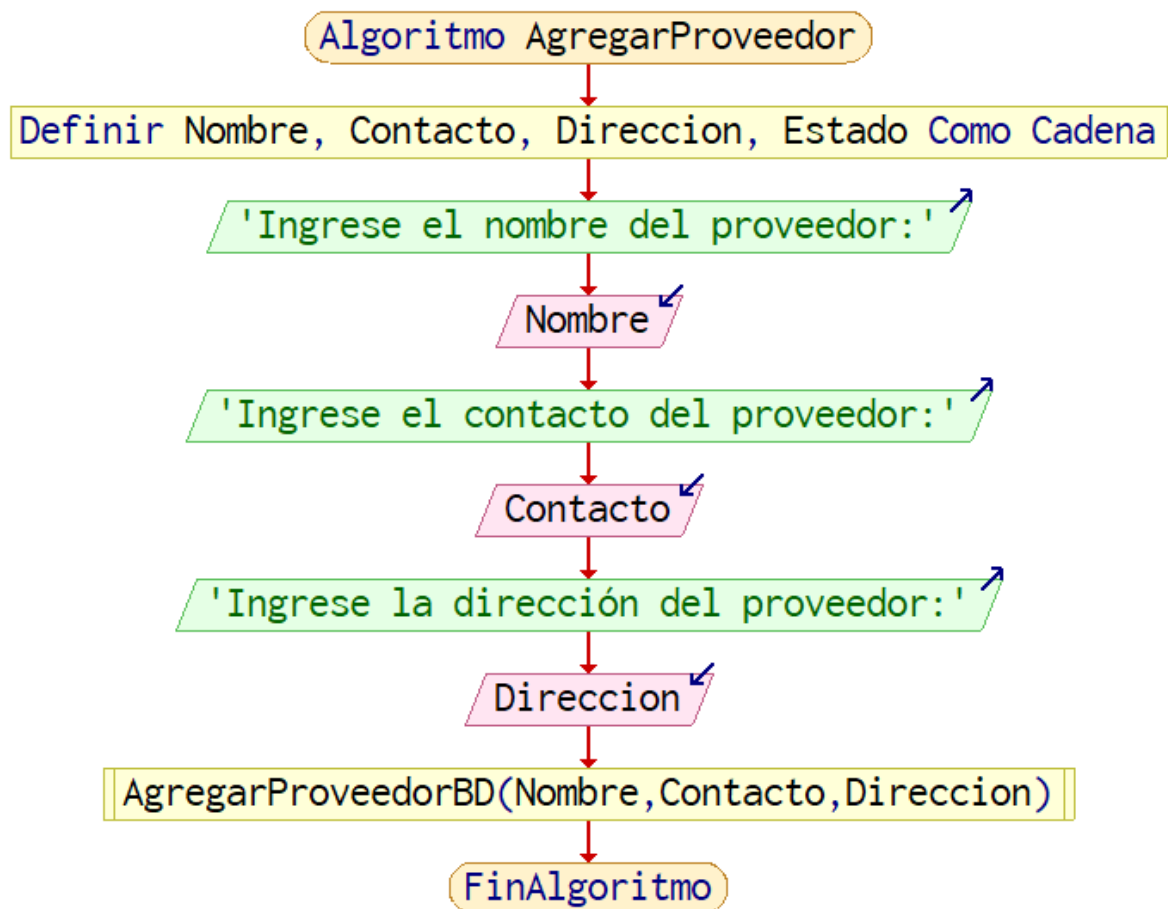
AGREGAR NUEVO PROVEEDOR

CÓDIGO FUENTE

```
const RegistroProveedor = () => {
  useEffect(() => {
    document.title = "PROVEEDORES"
  });
  const [selectedProveedorName, setSelectedProveedorName] = useState(""); // Estado para el
  nombre del proveedor seleccionado
  const [selectedEstado, setSelectedEstado] = useState(""); // Estado para filtrar por estado
  // Estado para controlar si se muestra el modal o no
  const [showModal, setShowModal] = useState(false);
  const [showEditModal, setEditShowModal] = useState(false);
  //AGREGAMOS ESTADOS PARA CARGAR LOS PROVEEDORES
  const [searchTerm, setSearchTerm] = useState("");
  const [proveedores, setProveedores] = useState([]);
  //ESTADOS PARA LOS DATOS
  const navigate = useNavigate();
  const { id } = useParams();
  const [Ruc, setRuc] = useState("");
  const [Nombre, setNombre] = useState("");
  const [Email, setEmail] = useState("");
  const [Contacto, setContacto] = useState("");
  const [Direccion, setDireccion] = useState("");
  const [Estado, setEstado] = useState("");

  useEffect(() => {
    getProveedores();
  }, []);
}
```

DIAGRAMA DE FLUJO



Grafo:



RUTAS

R1: 1, 2, 3, 4, 5, 6, 7

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G) = \text{número de nodos predicaados(decisiones)} + 1$

- $V(G)=0+1=1$
 $V(G) = A - N + 2$
 $V(G)= 6- 7 + 2 = 1$

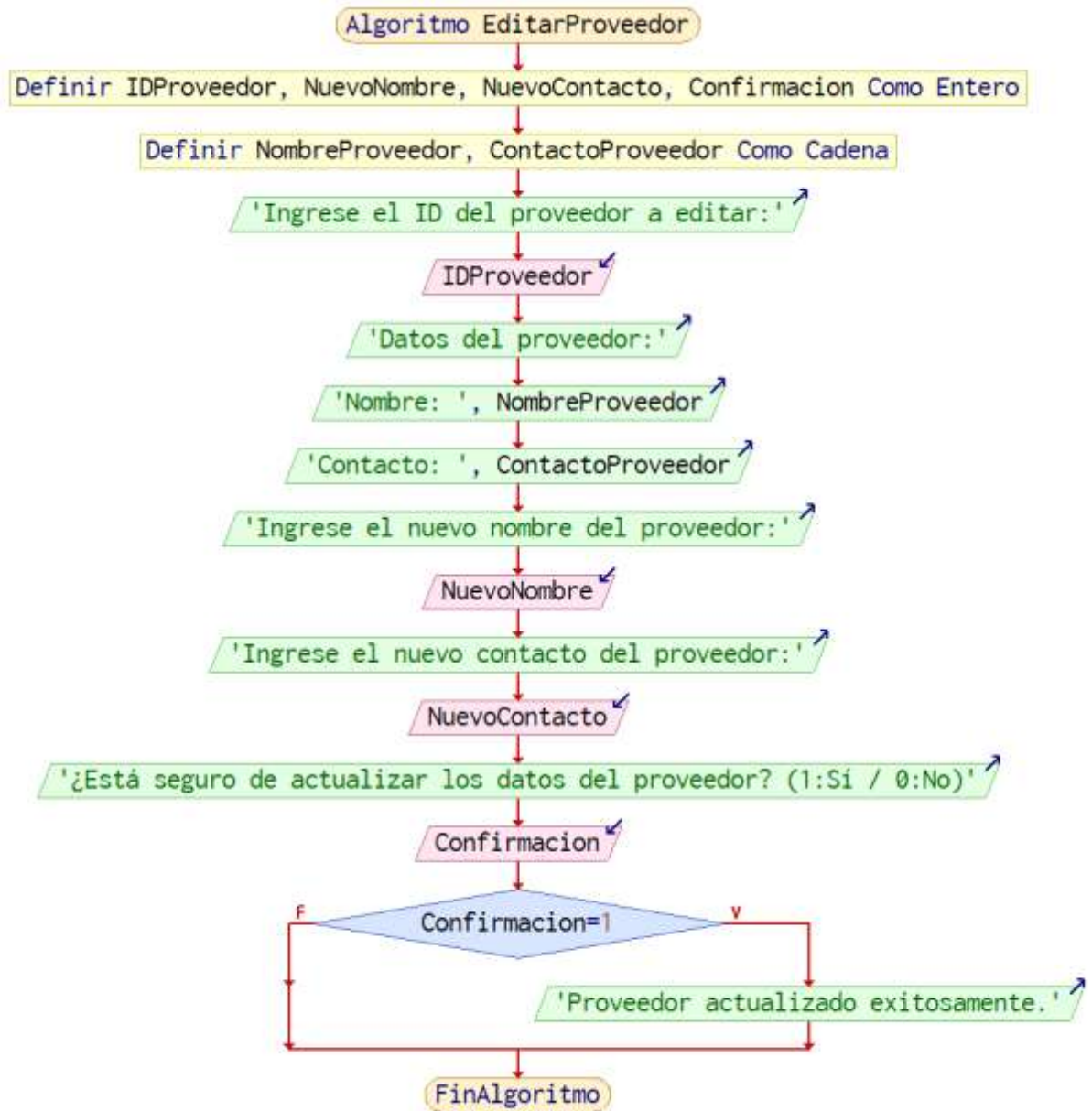
EDITAR PROVEEDOR

CÓDIGO FUENTE

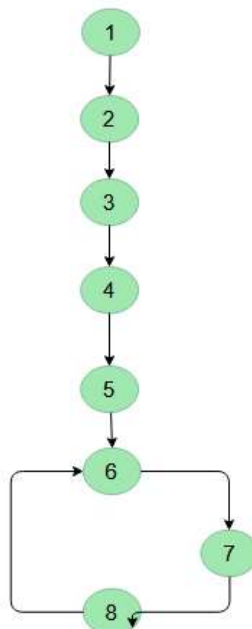
```
const update = async (e) => {
  e.preventDefault();
  // Validar los datos
  if (!validarFormulario()) {
    return;
  }

  try {
    const idAux = localStorage.getItem("id");
    await axios.put(URI + idAux, {
      PROV_RUC: Ruc,
      PROV_NOMBRE: Nombre,
      PROV_EMAIL: Email,
      PROV_CONTACTO: Contacto,
      PROV_DIRECCION: Direccion,
      PROV_ESTADO: Estado,
    });
    setEditShowModal(false);
    getProveedores();
  } catch (error) {
    console.error("Error al Actualizar Datos del Proveedor:", error);
  }
};
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 4, 5, 6, 7, 8.

R2: 1, 2, 3, 4, 5, 6, 7, 9.

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G) = \text{número de nodos predichados(decisiones)} + 1$
 $V(G) = 1 + 1 = 1$
- $V(G) = A - N + 2$
 $V(G) = 8 - 8 + 2 = 1$

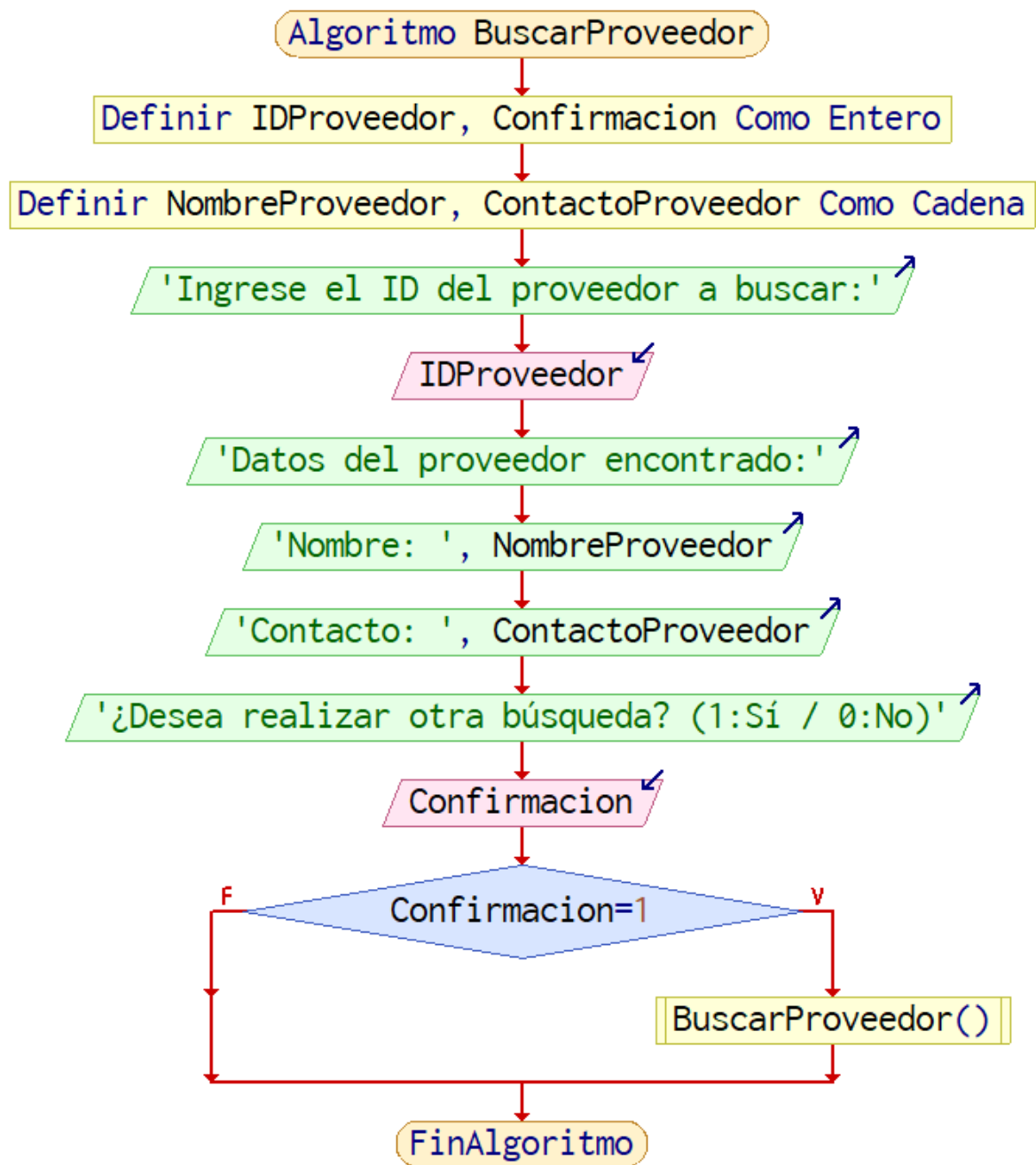
BUSCAR PROVEEDOR

CÓDIGO FUENTE

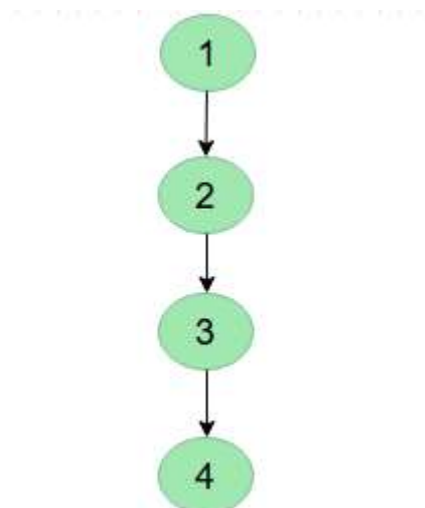
```
//funcion para cargar los datos del cliente por el ID enviado mediante la seleccion del boton editar de las tablas
const getProveedorById = async () => {
  try {
    var idAux = localStorage.getItem("id");
    const res = await axios.get(URI + idAux);

    if (res.data.length > 0) {
      setRuc(res.data[0].PROV_RUC);
      setNombre(res.data[0].PROV_NOMBRE);
      setEmail(res.data[0].PROV_EMAIL);
      setContacto(res.data[0].PROV_CONTACTO);
      setDireccion(res.data[0].PROV_DIRECCION);
      setEstado(res.data[0].PROV_ESTADO);
    } else {
      console.log("No se encontró ningún proveedor con el ID: " + idAux);
    }
  } catch (error) {
    // Manejo de errores
    console.error("Error al obtener los datos del proveedor:", error);
  }
};
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: R1: 1, 2, 3, 4

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

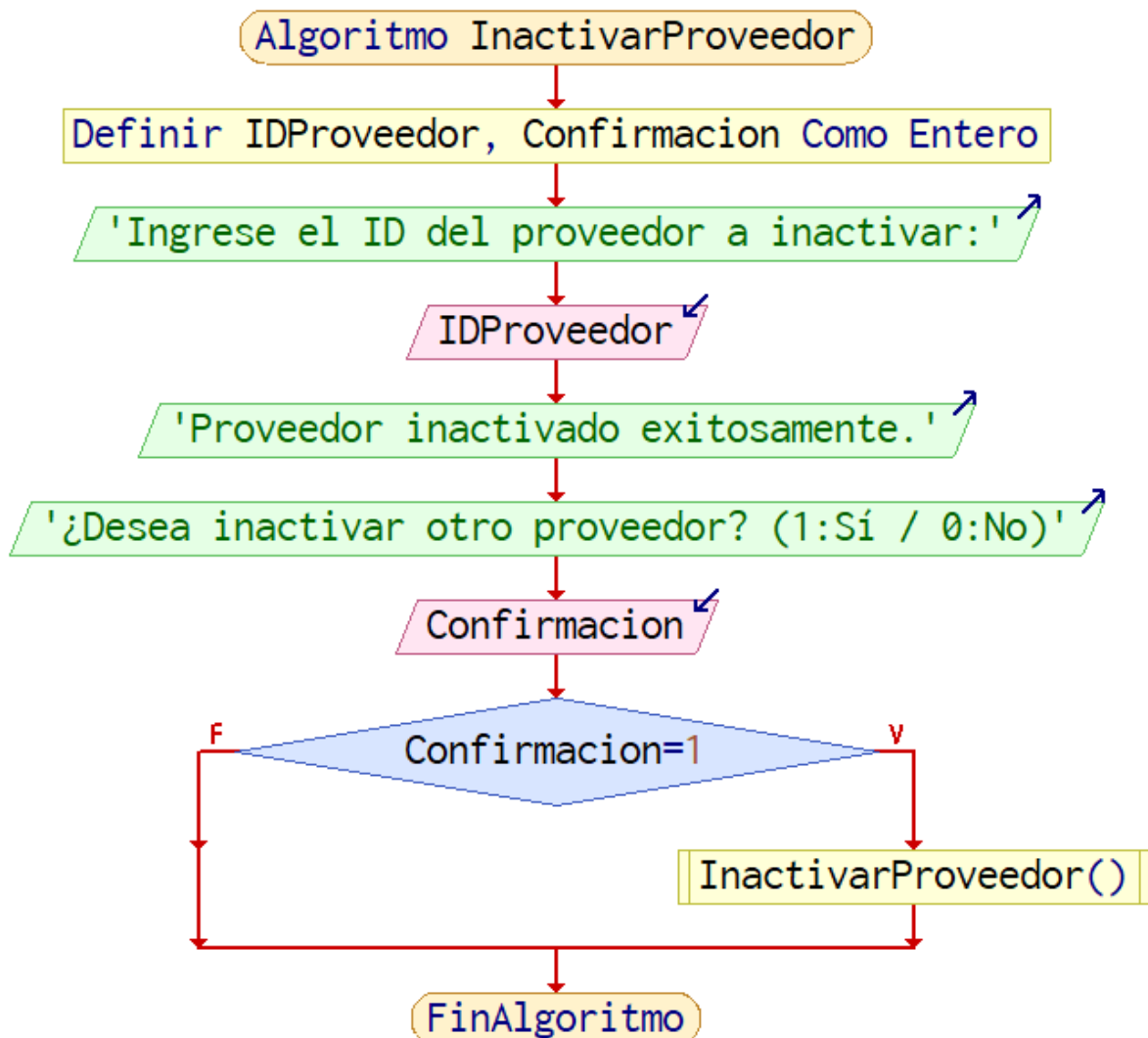
- $V(G) = \text{número de nodos predichados(decisiones)} + 1$
 $V(G) = 0 + 1 = 1$
- $V(G) = A - N + 2$
 $V(G) = 3 - 4 + 2 = 1$

INACTIVAR PROVEEDOR

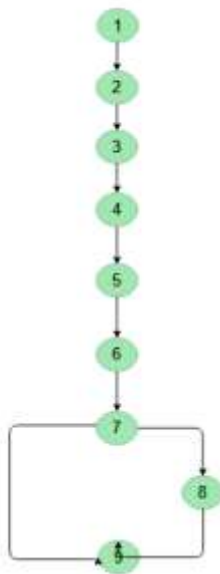
CÓDIGO FUENTE

```
const deleteProveedores = async (id) => {  
  await axios.put(URI + id, {  
    PROV_ESTADO: "Inactivo",  
  });  
  getProveedores();  
};
```

DIAGRAMA DE FLUJO



GRAFO



RUTAS

R1: 1, 2, 3, 4, 5, 6, 7, 8.

R2: 1, 2, 3, 4, 5, 6, 7, 9.

COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- $V(G) = \text{número de nodos predicados(decisiones)} + 1$
 $V(G) = 1 + 1 = 2$
- $V(G) = A - N + 2$ $V(G) =$
 $9 - 9 + 2 = 2$