

**Desarrollar software a partir de la integración de
sus módulos componentes.**

GA8-220501096-AA1-EV01

Nombre: John Jairo Zamudio Agudelo

Guía: 2799467

Instituto: Centros de servicios financieros

Formación: Análisis y desarrollo de software

Sena 2025

Introducción

El presente documento corresponde al desarrollo de un módulo web que permite al usuario **visualizar y actualizar sus datos personales**, integrando las capas de frontend, backend y base de datos.

Este módulo forma parte de un sistema mayor y fue construido aplicando buenas prácticas de programación, mecanismos de seguridad (hashing de contraseñas) y organización por capas, cumpliendo con los lineamientos de la metodología de desarrollo por componentes.

Requerimientos del sistema

Funcionales

- El usuario debe poder visualizar sus datos actuales (nombre, correo, teléfono, dirección).
- El usuario debe poder modificar sus datos y actualizarlos.
- El sistema debe validar contraseñas seguras antes de permitir cambios.
- Los cambios deben almacenarse correctamente en la base de datos.

No funcionales

- Arquitectura por capas (frontend, backend y base de datos).
- Manejo de contraseñas seguro mediante **bcrypt**.
- Interfaz responsive y clara.
- Código organizado, documentado y reutilizable.

Arquitectura del sistema

La aplicación está organizada en capas:

- **Frontend (HTML, CSS, JS)**: interfaz para editar datos y validaciones en tiempo real.
- **Backend (Node.js + Express)**: lógica de negocio, controladores y rutas REST.
- **Base de datos (MySQL)**: almacenamiento de usuarios e información asociada.

Flujo general:

Usuario → Frontend (formulario) → API REST (Express) → MySQL

Diagramas

Diagrama de clases

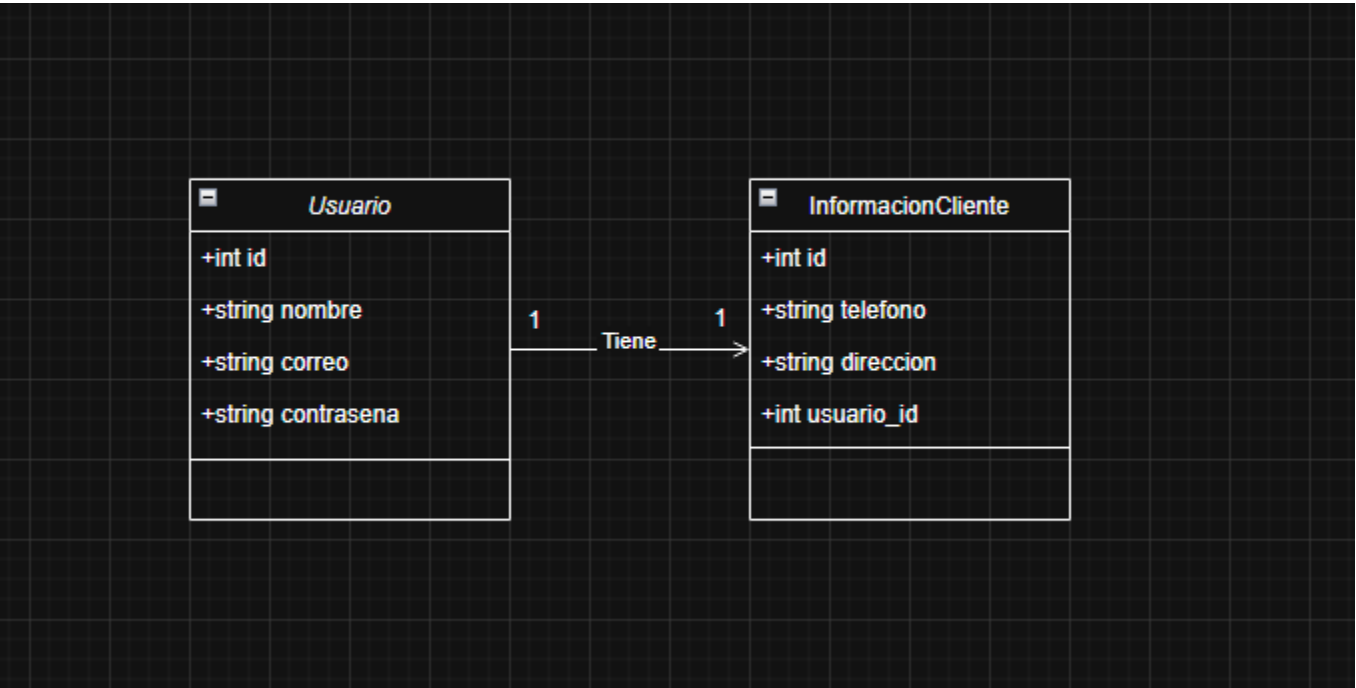


Diagrama de paquetes

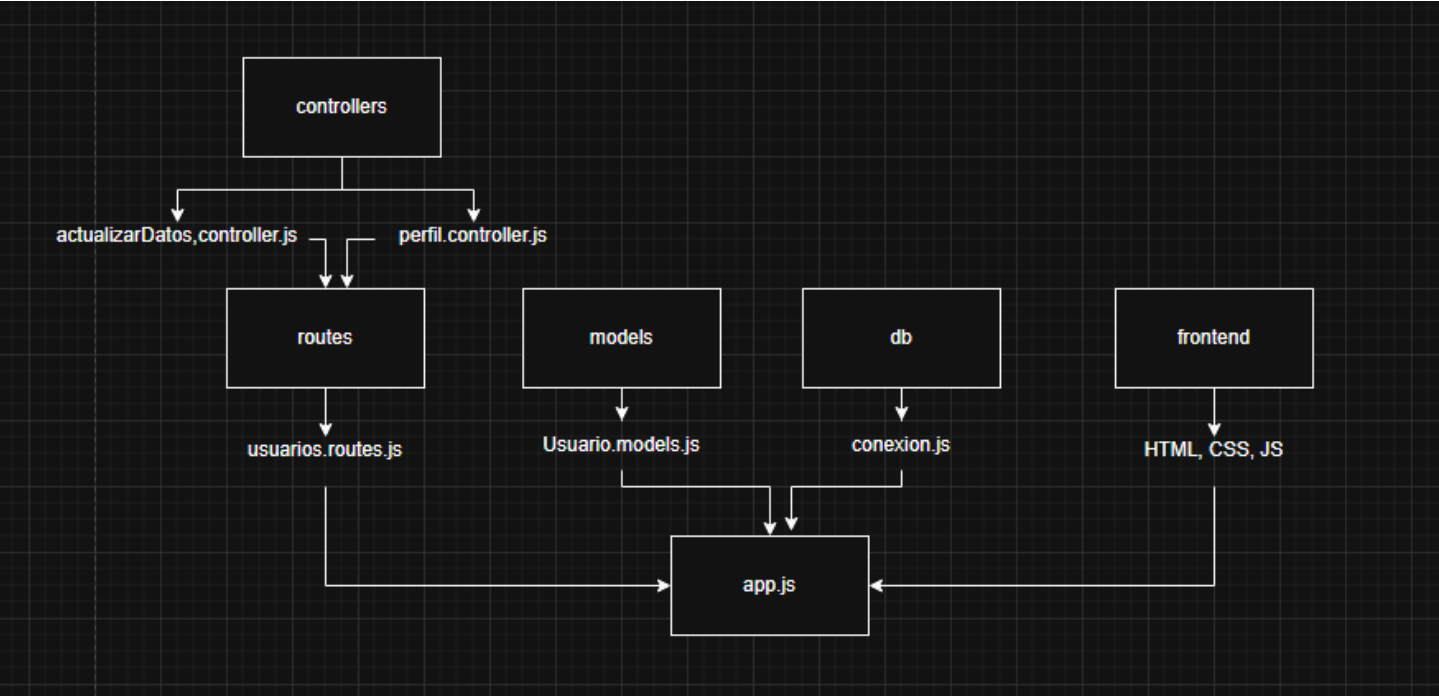
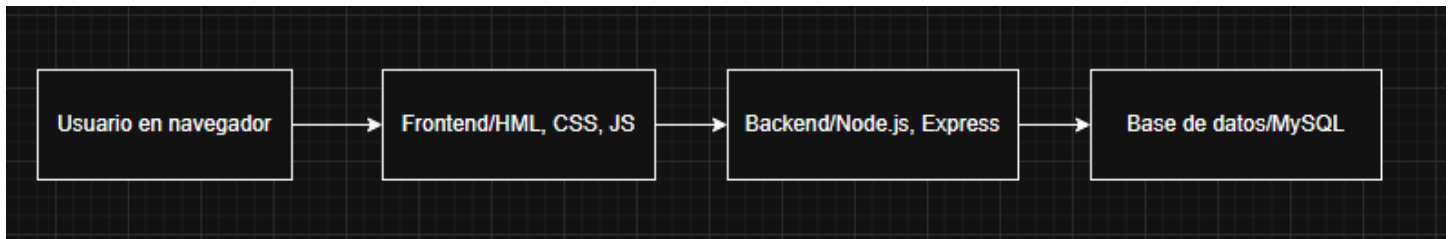
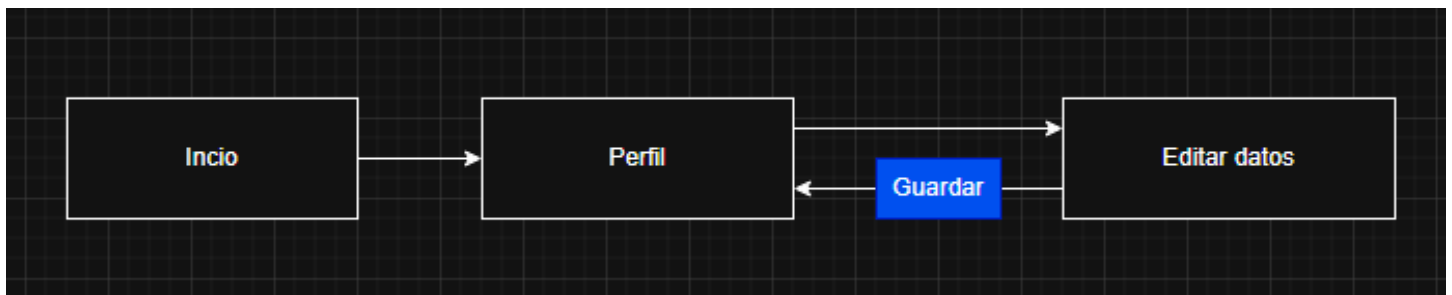


Diagrama de componentes



Mapa de navegación



5. Desarrollo de módulos

El módulo se compone de varios archivos que trabajan juntos:

- **Frontend**
 - cambiarDatos.html: formulario de perfil.
 - cambiarDatos.css: estilos responsive.
 - nav.css: barra de navegación.
 - cambiarDatos.js: lógica de validación y envío de datos.
 - mostrarSecciones.js: manejo de visibilidad de secciones.
 - bienvenido.html: página inicial.

- **Backend**

- app.js: configuración del servidor Express.
- usuarios.routes.js: rutas para actualizar y obtener datos.
- actualizarDatos.controller.js: lógica de actualización.
- perfil.controller.js: lógica de obtención de perfil.
- conexion.js: conexión a MySQL.
- usuario.model.js: funciones auxiliares.

- **Base de datos**

- modulo-cambiar-datos.sql: creación de tablas usuarios e informacion_cliente.

Buenas prácticas aplicadas

- Separación en capas (MVC simplificado).
- Validaciones en frontend y backend.
- Contraseñas protegidas con bcrypt.
- Código comentado y con nombres de fácil entendimiento.
- Reutilización de componentes (ejemplo: funciones de validación).
- Uso de un **repositorio Git** para control de versiones. <https://github.com/Spiner007/modulo-cambiar-datos.git>

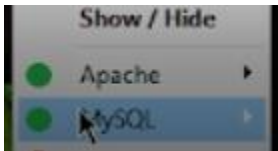
Pruebas unitarias e integración

Se realizaron pruebas en **Postman**:

- GET /api/usuarios/obtener-datos: devuelve datos del usuario.
- PUT /api/usuarios/actualizar-datos: actualiza información personal y contraseña.

También pruebas en frontend: validación de contraseña en tiempo real y campos obligatorios.

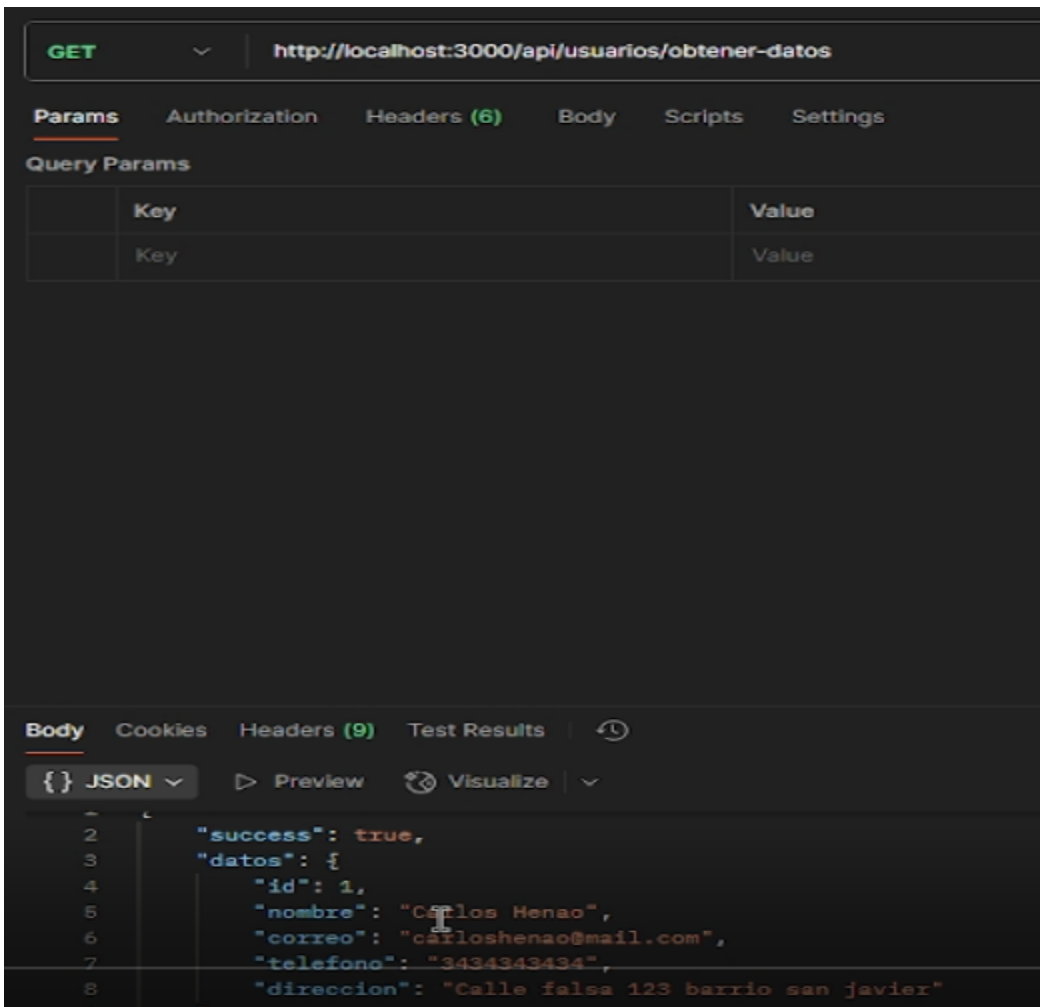
Apache + MYSQL



Servidor corriendo

```
PowerShell 7.5.2
Loading personal and system profiles took 679ms.
C:\Users\John\Desktop [master +29 ~0 -0 !]> set-location modulo-cambiar-datos
C:\Users\John\Desktop\modulo-cambiar-datos [master +29 ~0 -0 !]> set-location backend
C:\Users\John\Desktop\modulo-cambiar-datos\backend [master +29 ~0 -0 !]> node app.js
Servidor escuchando en http://localhost:3000
```

Obtener los datos



Actualizar datos

The screenshot shows a REST client interface with a PUT request to `http://localhost:3000/api/usuarios/actualizar-datos`. The request body is a JSON object with the following fields:

```
{
  "nombre": "Carlos Henao",
  "correo": "carloshenao@mail.com",
  "telefono": "343454334344",
  "direccion": "calle falsa 123 barrio buenos aires",
  "nuevaContrasena": "CarlosHenao123!",
  "confirmarContrasena": "CarlosHenao123!"
}
```

The response is also in JSON format, indicating success:

```
{
  "success": true,
  "message": "Datos actualizados correctamente.",
  "datos": {
    "id": 1,
    "nombre": "Carlos Henao",
    "correo": "carloshenao@mail.com",
    "telefono": "343454334344",
    "direccion": "calle falsa 123 barrio buenos aires"
  }
}
```

Contraseña carácter especial

The screenshot shows a REST client interface with a PUT request to `http://localhost:3000/api/usuarios/actualizar-datos`. The request body is a JSON object with the following fields:

```
{
  "nombre": "Carlos Henao",
  "correo": "carloshenao@mail.com",
  "telefono": "343454334344",
  "direccion": "calle falsa 123 barrio buenos aires",
  "nuevaContrasena": "CarlosHenao123",
  "confirmarContrasena": "CarlosHenao123!"
}
```

The response body shows the result of the request:

```
{
  "success": false,
  "message": "La contraseña debe contener al menos un carácter especial."
}
```

The error message indicates that the password "CarlosHenao123" does not contain at least one special character.

Contraseña al menos un numero

The screenshot shows a REST client interface with the following details:

- URL:** `http://localhost:3000/api/usuarios/actualizar-datos`
- Method:** `PUT`
- Body Type:** `raw` (selected)
- Body Content:**

```
1 {  
2   "nombre": "Carlos Henao",  
3   "correo": "carloshenao@mail.com",  
4   "telefono": "343454334344",  
5   "direccion": "calle falsa 123 barrio buenos aires",  
6   "nuevaContrasena": "CarlosHenao!",  
7   "confirmarContrasena": "CarlosHenao123!"  
8 }
```
- Response Body:**

```
1 {  
2   "success": false,  
3   "message": "La contraseña debe contener al menos un número."
```

contraseña debe contener al menos una mayúscula

The screenshot shows a REST client interface with a PUT request to `http://localhost:3000/api/usuarios/actualizar-datos`. The request body is a JSON object with the following fields: `nombre`, `correo`, `telefono`, `direccion`, `nuevaContrasena`, and `confirmarContrasena`. The response body shows `success: false` and a message: `"La contraseña debe contener al menos una mayúscula."`

PUT `http://localhost:3000/api/usuarios/actualizar-datos`

Params Authorization Headers (8) **Body** Scripts Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ▾

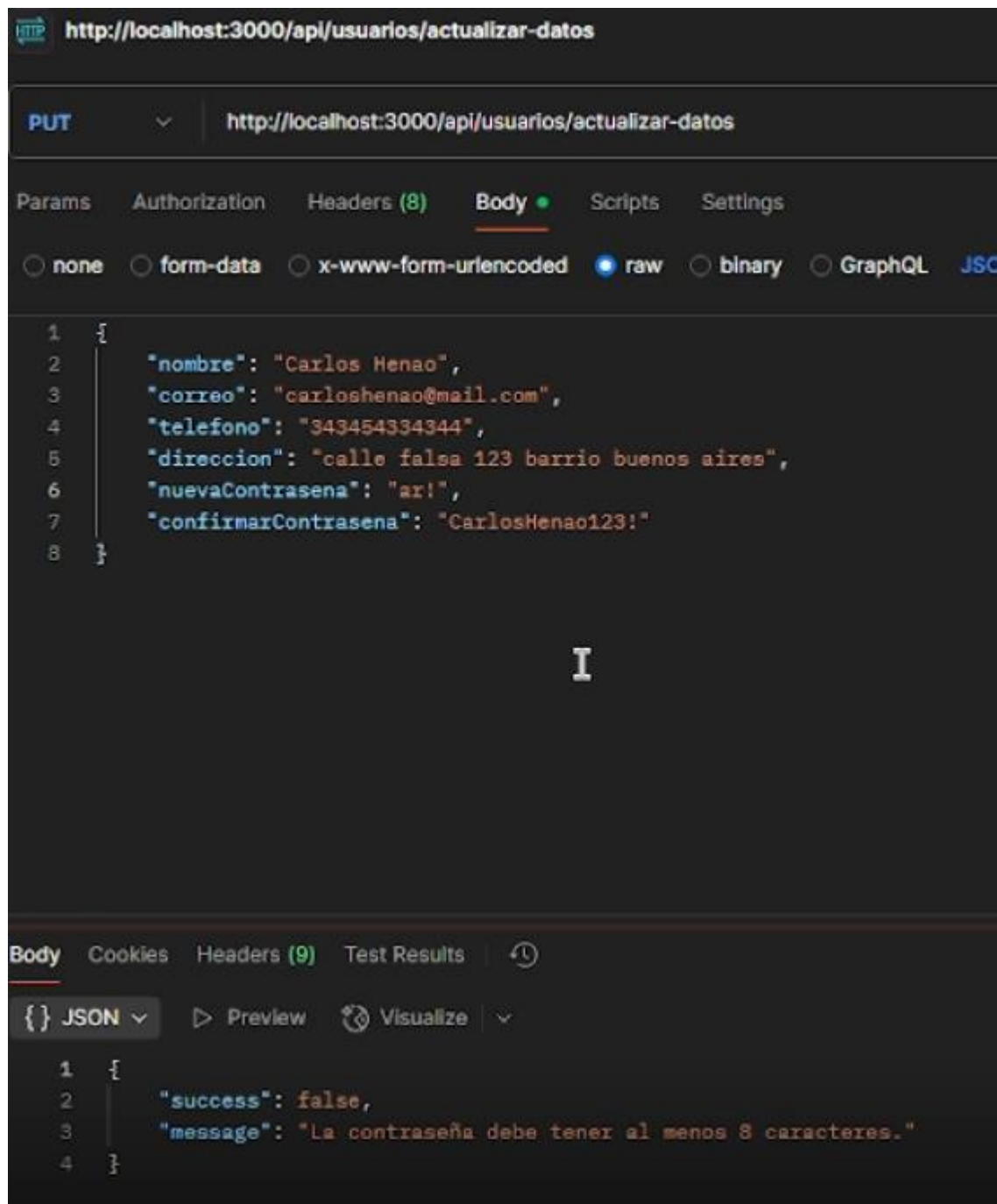
```
1 {
2   "nombre": "Carlos Menao",
3   "correo": "carloshenao@mail.com",
4   "telefono": "343454334344",
5   "direccion": "calle falsa 123 barrio buenos aires",
6   "nuevaContrasena": "arlosenao123!",
7   "confirmarContrasena": "CarlosMenao123!"
8 }
```

Body Cookies Headers (9) Test Results ↻

{ } JSON ▾ ▶ Preview ↻ Visualize ▾

```
1 {
2   "success": false,
3   "message": "La contraseña debe contener al menos una mayúscula."
4 }
```

Contraseña al menos con 8 caracteres



HTTP PUT http://localhost:3000/api/usuarios/actualizar-datos

Params Authorization Headers (8) **Body** Scripts Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSC

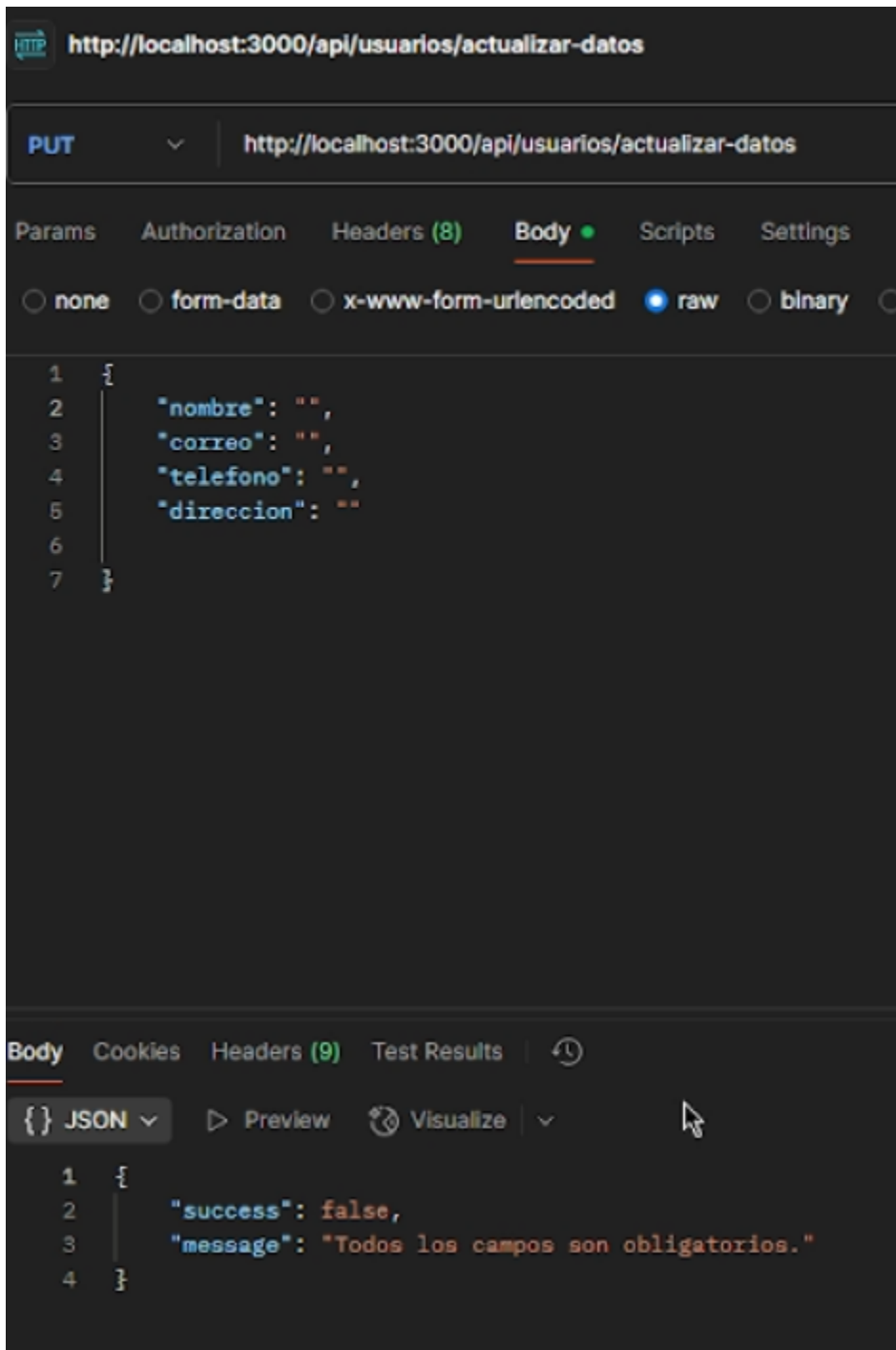
```
1 {
2   "nombre": "Carlos Henao",
3   "correo": "carloshenao@mail.com",
4   "telefono": "343454334344",
5   "direccion": "calle falsa 123 barrio buenos aires",
6   "nuevaContraseña": "ar!",
7   "confirmarContraseña": "CarlosHenao123!"
8 }
```

Body Cookies Headers (9) Test Results ↻

{ } JSON ▾ ▶ Preview 🔄 Visualize ▾

```
1 {
2   "success": false,
3   "message": "La contraseña debe tener al menos 8 caracteres."
4 }
```

Contraseña Todos los campos son obligatorios







HTTP **PUT** `http://localhost:3000/api/usuarios/actualizar-datos`

Params Authorization Headers (8) **Body** Scripts Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐

```
1 {
2   "nombre": "",
3   "correo": "",
4   "telefono": "",
5   "direccion": ""
6 }
7 }
```

Body Cookies Headers (9) Test Results 

{ } JSON  Preview  Visualize 

```
1 {
2   "success": false,
3   "message": "Todos los campos son obligatorios."
4 }
```

Base de datos (modulo-cambiar-datos) tabla **información_cliente**

	id	usuario_id	telefono	direccion
▶	1	1	343454334344	calle falsa 123 barrio buenos aires
•	<input type="text" value="NULL"/>	<input type="text" value="NULL"/>	<input type="text" value="NULL"/>	<input type="text" value="NULL"/>

Base de datos (modulo-cambiar-datos) tabla **usuarios**

	id	nombre	correo	contrasena
▶	1	Carlos Henao	carloshenao@mail.com	\$2b\$10\$GJN1CbxJwW3K9ydXioGe.E/80z0UFg...
•	<input type="text" value="NULL"/>	<input type="text" value="NULL"/>	<input type="text" value="NULL"/>	<input type="text" value="NULL"/>

Datos actualizados correctamente

Nombre

Correo

Teléfono

Dirección + Barrio

ID del Usuario: 1

! Rellene este campo.

localhost:3000 dice

Datos actualizados correctamente.

Aceptar

EDITAR INFORMACIÓN

Contraseña con un carácter especial

Online Validat... Online FlowChart &... Tur... localhost:3000 dice

La contraseña debe contener al menos un carácter especial.

Aceptar

EDITAR INFORMACIÓN

Nombre Carlos Henao Teléfono 2332322323 ID del Usuario: 1

Contraseña con almenos un número

Online Validat... Online FlowChart &... Tur... localhost:3000 dice

La contraseña debe contener al menos un número.

Aceptar

EDITAR INFORMACIÓN

Nombre Carlos Henao Teléfono 2332322323 ID del Usuario: 1

Correo carloshenao@mail.com Dirección + Barrio calle falsa 123 barrio buenos aires

Nueva Contraseña Confirmar Contraseña

caracteres
yúscula
mero
rácter especial

Contraseña al menos una mayúscula

localhost:3000 dice

La contraseña debe contener al menos una mayúscula.

Aceptar

EDITAR INFORMACIÓN

Nombre

Henao

Teléfono

2332322323

ID del Usuario

Correo

henao@mail.com

Dirección + Barrio

calle falsa 123 barrio buenos aires

Contraseña

Confirmar Contraseña

Intereses

Contraseña al menos con 8 caracteres

localhost:3000 dice

La contraseña debe tener al menos 8 caracteres.

Aceptar

EDITAR INFORMACIÓN

Nombre

Carlos Henao

Teléfono

2332322323

ID del Usuario

Correo

carloshenao@mail.com

Dirección + Barrio

calle falsa 123 barrio buenos aires

Nueva Contraseña

..

Confirmar Contraseña

..

Contraseña no coinciden

Nueva Contraseña	Confirmar Contraseña
<input type="password"/>	<input type="password"/>
<div>✓ 8+ caracteres</div> <div>✓ Mayúscula</div> <div>✓ Número</div> <div>✓ Carácter especial</div>	<div>! Las contraseñas no coinciden</div>

Configuración del entorno

- **IDE:** Visual Studio Code.
- **Servidor:** Node.js v22.14.0.
- **Framework:** Express.js.
- **Base de datos:** MySQL 8.0.
- **Gestor de dependencias:** NPM 10.9.2.
- **Control de versiones:** Git.

Conclusiones

- Se construyó un módulo funcional para edición de perfil.
- Se aplicaron principios de arquitectura por capas y buenas prácticas de programación.
- Se integraron correctamente frontend, backend y base de datos.
- Se validó la seguridad en el manejo de contraseñas.
- El módulo es escalable y se puede integrar fácilmente con otros componentes del sistema.

Referencias web

Microsoft. (2024). *Documentación oficial de Express.js en Node.js*:

<https://expressjs.com/es/>

MySQL. (2024). *MySQL 8.0 Reference Manual*:

<https://dev.mysql.com/doc/>