

**DISEÑO Y DESARROLLO DE SERVICIOS WEB-PROYECTO  
GA7-220501096-AA5-EV01**

**NOMBRE DEL APRENDIZ**

Carlos Arturo Santos Castellar

**Instructor**

ALIRIO BETANCOURTGARCIA

**FICHA 3070302**

Servicio Nacional de Aprendizaje - SENA

Centro de Materiales y Ensayos - CME

Análisis y Desarrollo de Software – ADSO

2025

## **TABLA DE CONTENIDO**

|  |    |
|--|----|
| 1. INTRODUCCIÓN.....   | 3  |
| 2. OBJETIVO.....   | 4  |
| 3. EVIDENCIA DE DESEMPEÑO: GA7-220501096-AA5-EV01 DISEÑO Y DESARROLLO DE SERVICIOS WEB-PROYECTO..... | 4  |
| 4. DESARROLLO DE EVIDENCIA.....  | 5  |
| 5. CONCLUSIONES.....   | 10 |
| 6. REFERENCIAS.....  | 10 |

## **INTRODUCCIÓN**

El desarrollo de aplicaciones web modernas requiere la integración de diferentes tecnologías que permitan crear interfaces funcionales y sistemas capaces de procesar información. En esta actividad se construyó una solución completa basada en un frontend desarrollado con HTML, CSS y JavaScript, conectado a un backend construido con Node.js y Express.

El propósito principal es implementar un sistema básico que permita registrar usuarios e iniciar sesión mediante solicitudes HTTP, evidenciando el funcionamiento del flujo cliente servidor y la validación de datos. A través de esta práctica se ponen en uso conceptos fundamentales de programación web, comunicación mediante APIs y manejo de respuestas en formato JSON.

## **OBJETIVO**

Implementar un sistema web funcional que permita el registro y la autenticación de usuarios mediante la construcción de una API en Node.js y su integración con una interfaz gráfica, demostrando el uso adecuado de solicitudes HTTP, estructura cliente servidor y procesamiento de datos.

## **EVIDENCIA DE DESEMPEÑO: GA7-220501096-AA5-EV01 DISEÑO Y DESARROLLO DE SERVICIOS WEB-PROYECTO**

Tomando como referencia lo visto en el componente formativo “Construcción de API” que pueden encontrar en: Contenido del curso -> PROYECTO -> Fase 3 Ejecución -> Actividad de proyecto 7 -> Material de formación, realizar el diseño y la codificación de un servicio web para el proyecto que ha venido desarrollando, teniendo en cuenta lo siguiente:

Se requiere realizar un servicio web para un registro y un inicio de sesión. El servicio recibirá un usuario y una contraseña, si la autenticación es correcta saldrá un mensaje de autenticación satisfactoria en caso contrario debe devolver error en la autenticación.

El código debe contener comentarios.

**NOTA:** Puede seguir las instrucciones dadas en el componente formativo “Construcción de API” para el desarrollo de esta evidencia o también puede utilizar las herramientas de desarrollo que hayan seleccionado para el desarrollo de su proyecto.

### **Lineamientos generales para la entrega de la evidencia:**

- Productos para entregar: carpeta comprimida que debe tener los siguientes archivos: archivos del proyecto y archivo con enlace del repositorio, la carpeta comprimida debe tener el nombre del aprendiz y número de la evidencia así:  
NOMBRE\_APELLIDO\_AA5\_EV01.
- Describa los pasos a seguir para el desarrollo de la evidencia a través de capturas de pantalla y explique cada una.
- Extensión: ZIP, RAR.

Para hacer el envío de la evidencia remítase al área de la actividad correspondiente y acceda al espacio: diseño y desarrollo de servicios web - caso. GA7-220501096-AA5-EV01

## **DESARROLLO DE EVIDENCIA:**

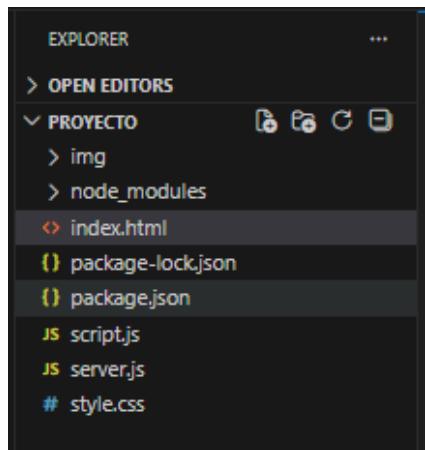
DISEÑO Y DESARROLLO DE SERVICIOS WEB-PROYECTO.

### **1. Creación de la estructura del proyecto**

Primero se creó la carpeta del proyecto con dos partes principales:

**(Frontend, Backend)**

**Captura 1:** Estructura de carpetas del proyecto



**Explicación:**

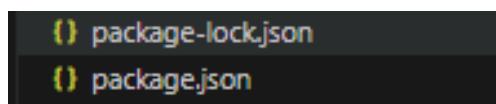
Aquí se muestra la organización inicial del proyecto donde se separan los archivos del cliente (interfaz) y el servidor (API). Esto permite trabajar de forma ordenada.

### **2. Instalación y configuración del servidor con Node.js**

En una terminal se ejecutaron los siguientes comandos:

- **npm init -y**
- **npm install express cors**

**Captura 2:** Terminal instalando dependencias



**Explicación:**

Se creó el archivo package.json e instalamos los paquetes necesarios para que el servidor pueda funcionar.

### **3. Creación del archivo server.js**

Se desarrolló la API con dos rutas principales:

- **register** Guarda usuarios en memoria
- **login** Valida el usuario y contraseña

### Captura 3: Código del servidor (server.js)



```
index.html JS server.js X
JS server.js > ...
1 const express = require("express");
2 const app = express();
3 const cors = require("cors");
4
5 // Middleware
6 app.use(cors());
7 app.use(express.json());
8
9 // "Base de datos" temporal en memoria
10 let usuarios = [];
11
12 // REGISTRO
13 app.post("/register", (req, res) => {
14   const { usuario, password } = req.body;
15   if (!usuario || !password) {
16     return res.status(400).json({ message: "Usuario y contraseña requeridos." });
17   }
18   const existe = usuarios.find(u => u.usuario === usuario);
19   if (existe) {
20     return res.status(409).json({ message: "El usuario ya existe." });
21   }
22   usuarios.push({ usuario, password });
23   return res.json({ message: "Registro exitoso." });
24 });
25
26 // LOGIN
27 app.post("/login", (req, res) => {
28   const { usuario, password } = req.body;
29   const user = usuarios.find(
30     u => u.usuario === usuario && u.password === password
31   );
32   if (!user) {
33     return res
34       .status(401)
35       .json({ message: "Error en la autenticación." });
36   }
37   return res.json({ message: "Autenticación satisfactoria." });
38 });
39
40 app.listen(3000, () => {
41   console.log("API escuchando en http://localhost:3000");
42 });
43
```

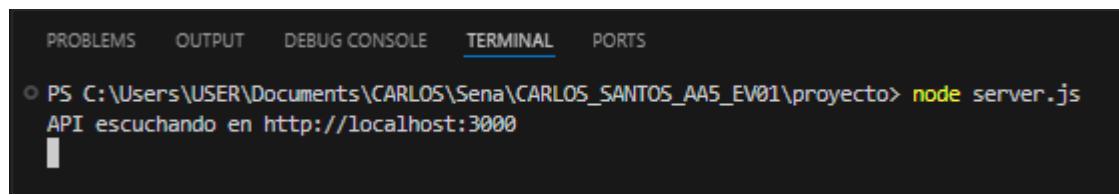
#### Explicación:

En esta captura se evidencia la programación del servidor en Node.js. Aquí se definen las rutas que permiten registrar usuarios e iniciar sesión. La “base de datos” utilizada es temporal.

#### 4. Ejecución del servidor

Para iniciar la API se utilizó: **node server.js**

#### Captura 4: Mensaje en consola “API escuchando en <http://localhost:3000>”



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\USER\Documents\CARLOS\Sena\CARLOS_SANTOS_AA5_EV01\ proyecto> node server.js
API escuchando en http://localhost:3000
```

#### Explicación:

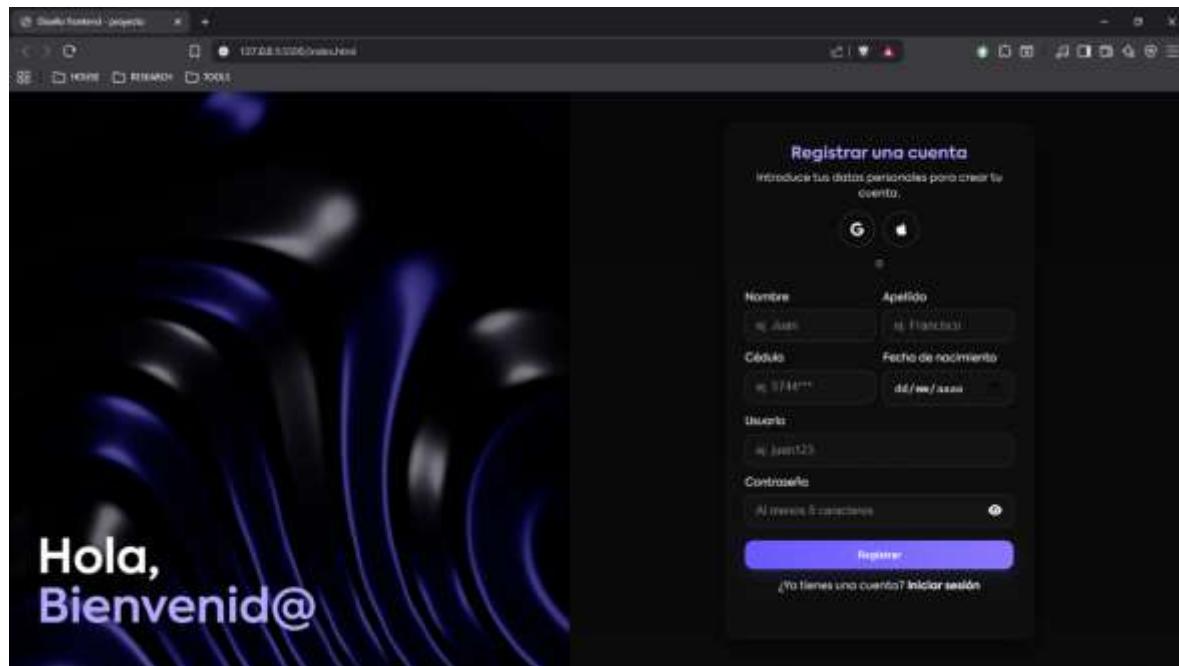
Esta imagen demuestra que el servidor está corriendo correctamente y listo para recibir solicitudes del frontend.

#### 5. Desarrollo de la interfaz de registro

Se diseñó un formulario moderno con los campos:

**(Nombre, Apellido ,Cédula, Fecha de nacimiento, Usuario, Contraseña)**

#### Captura 5: Formulario de registro en el navegador



#### Explicación:

Aquí se observa la interfaz donde el usuario ingresa sus datos. Esta sección envía la información a la ruta /register mediante fetch().

#### 6. Registro exitoso

Cuando el usuario completa el formulario, la interfaz muestra un mensaje:

“Registro exitoso.”

#### Captura 6: Mensaje de registro exitoso en pantalla

## Registrar una cuenta

Introduce tus datos personales para crear tu cuenta.

O

Nombre: carlos Apellido: santos

Cédula: 1004462287 Fecha de nacimiento: 25/02/2002

Usuario: C\_santos

Contraseña:  Mostrar

**Registrar**

¿Ya tienes una cuenta? [Iniciar sesión](#)

Registro exitoso. Ya puedes iniciar sesión.

The screenshot shows a registration form titled "Registrar una cuenta". It includes fields for Name (carlos, santos), ID (1004462287), Birthdate (25/02/2002), Username (C\_santos), and Password (represented by a masked input). There are also links for existing users to log in and a success message at the bottom.

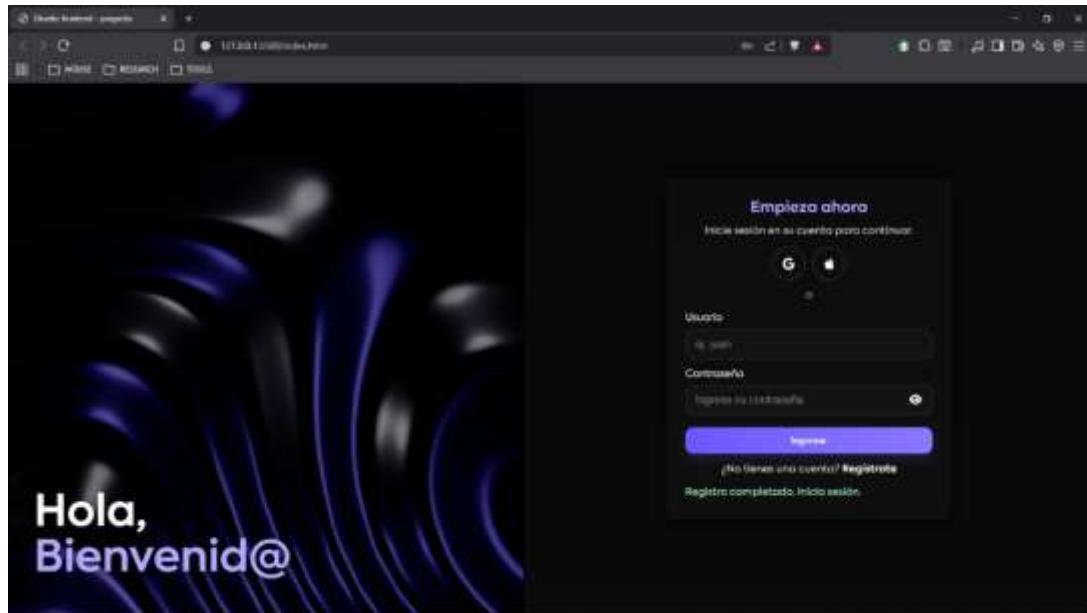
### Explicación:

Esto confirma que el formulario se comunica correctamente con el servidor y el usuario queda almacenado temporalmente en el arreglo.

### 7. Desarrollo de la interfaz de inicio de sesión

Se creó un formulario que solicita: (**Usuario ,Contraseña**)

**Captura 7:** Pantalla de inicio de sesión



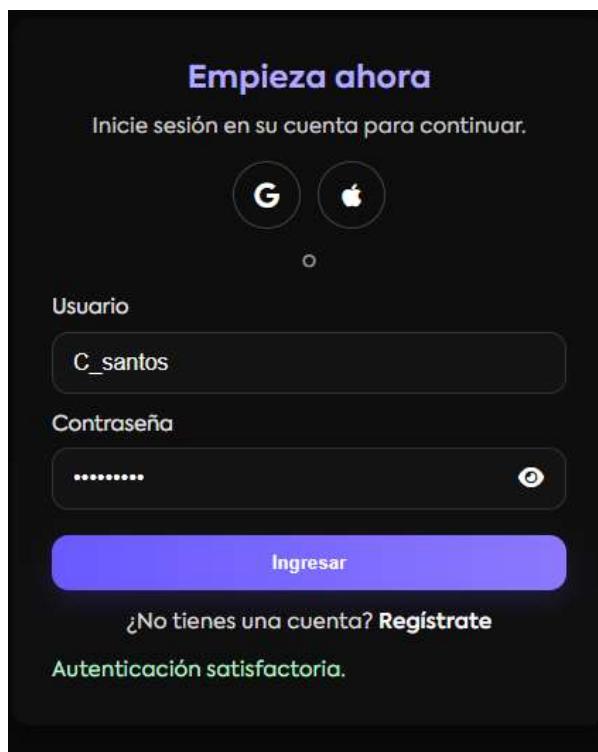
**Explicación:**

Este formulario envía credenciales al endpoint /login. Si coinciden, el servidor responde exitosamente.

## 8. Prueba de inicio de sesión

El usuario ingresa los datos registrados anteriormente.

**Captura 8:** “Autenticación satisfactoria.”



**Explicación:**

Esta captura evidencia que:

1. El servidor recibió las credenciales
2. Las validó correctamente
3. La API respondió afirmativamente
4. El frontend interpretó la respuesta y mostró el mensaje al usuario

## **CONCLUSIONES**

La realización de esta actividad permitió comprender y aplicar los principios esenciales del desarrollo web full-stack, integrando correctamente el frontend con un backend basado en Express. Se logró implementar un flujo completo de registro e inicio de sesión, validar información y manipular datos en memoria, cumpliendo con los requerimientos establecidos.

Este ejercicio fortalece habilidades prácticas en la creación de APIs, uso de JavaScript tanto en el cliente como en el servidor y comprensión del funcionamiento de las solicitudes HTTP. Además, sienta bases importantes para el desarrollo de sistemas más avanzados con almacenamiento persistente y mayores niveles de seguridad.

## **REFERENCIAS**

SENA. (s.f.). *Tema 1* [Página web].

<https://zajuna.sena.edu.co/Repositorio/Titulada/institution/SENA/Tecnologia/228118/Contenido/OVA/CF33/index.html#/curso/tema1>

SENA. (s.f.). *Tema 2* [Página web].

<https://zajuna.sena.edu.co/Repositorio/Titulada/institution/SENA/Tecnologia/228118/Contenido/OVA/CF33/index.html#/curso/tema2>

SENA. (s.f.). *Tema 3* [Página web].

<https://zajuna.sena.edu.co/Repositorio/Titulada/institution/SENA/Tecnologia/228118/Contenido/OVA/CF33/index.html#/curso/tema3>