



Universidad Tecmilenio

Campus Las Torres

Materia: Desarrollo FullStack

Avance del Proyecto

Alumno: Patricio Calvo

Matricula: 07097795

Enero / 2026

Link de Github:

<https://github.com/Diablo650cc/Desarrollo-FullStack/tree/main/Avance%20del%20Proyecto>

REPORTE DE ACTIVIDAD

Objetivo del Proyecto

Crear una aplicación web completa que permita a los usuarios registrarse, iniciar sesión y gestionar información mediante un sistema seguro y funcional.

Backend - Servidor y API

Se implementó un servidor con Node.js y Express que funciona como API RESTful. Este servidor realiza operaciones de base de datos MySQL (crear, leer, actualizar y eliminar registros). Se añadió autenticación mediante JSON Web Tokens (JWT) y se protegió la información con hashing de contraseñas usando bcryptjs.

Base de Datos

Se diseñó una base de datos MySQL con una tabla de usuarios que almacena nombre, correo electrónico, contraseñas protegidas y fecha de registro. La conexión se optimizó para manejar múltiples solicitudes simultáneas mediante un sistema de pooling.

Frontend - Interfaz de Usuario

Se desarrollaron tres páginas principales: inicio de sesión, registro de usuarios y panel de control (dashboard). Estas páginas utilizan HTML para estructura, CSS para diseño visual y JavaScript para funcionalidades interactivas y comunicación con el backend.

Sistema de Autenticación y Navegación

El sistema verifica automáticamente el estado de autenticación del usuario. Si un usuario no logueado intenta acceder al panel de control, es redirigido al login. Si un usuario ya autenticado intenta acceder a login o registro, es redirigido directamente al dashboard.

Arquitectura y Organización

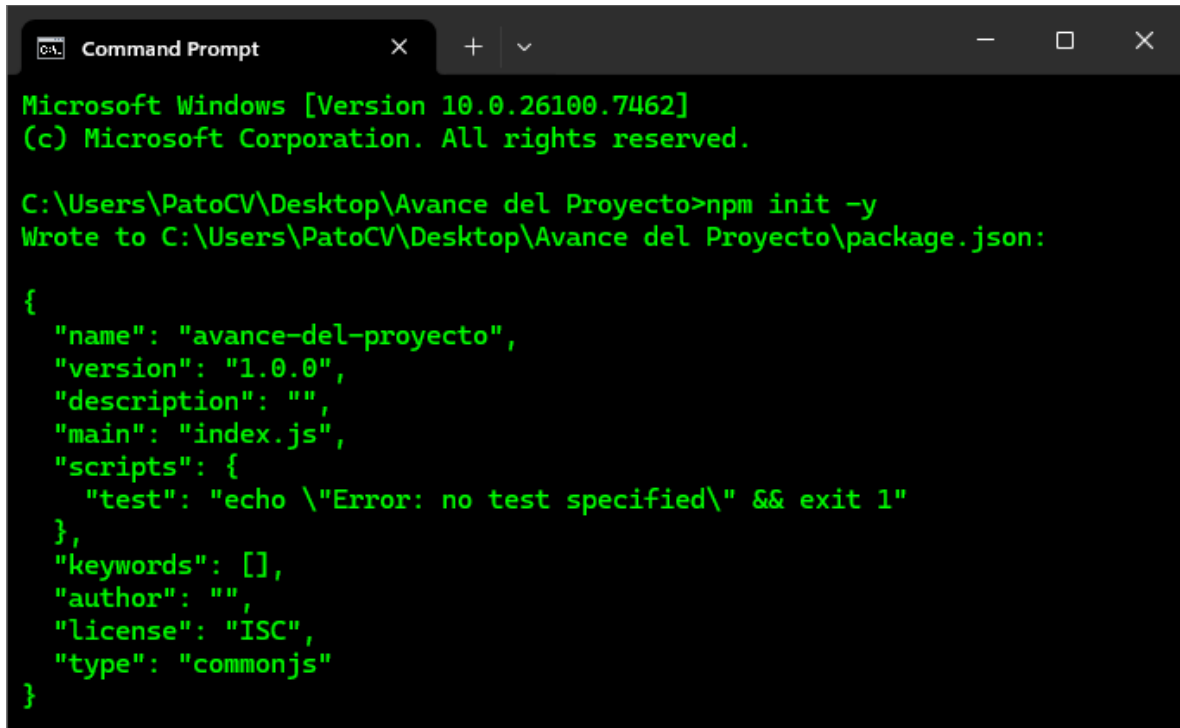
El proyecto se estructuró en módulos separados por carpetas: configuración de base de datos, middlewares de autenticación, rutas de la API y archivos estáticos del frontend. Se utilizaron variables de entorno para manejar información sensible de manera segura.

Resultado Final

Se obtuvo una aplicación web funcional que integra frontend y backend, con autenticación segura, gestión de base de datos y una interfaz de usuario operativa. El sistema demuestra los principios fundamentales del desarrollo web moderno.

A: CONFIGURACIÓN DEL PROYECTO

Inicializar el proyecto con npm

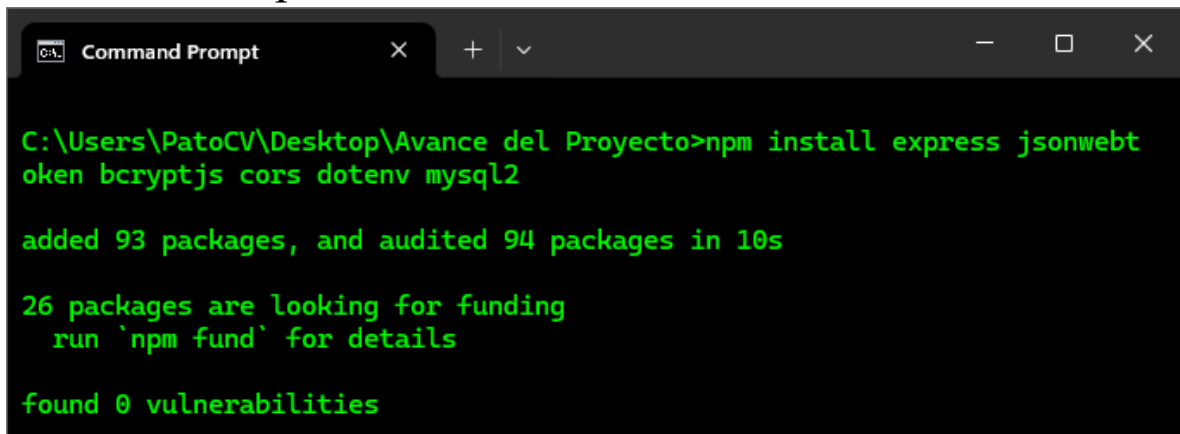


```
Microsoft Windows [Version 10.0.26100.7462]
(c) Microsoft Corporation. All rights reserved.

C:\Users\PatoCV\Desktop\Avance del Proyecto>npm init -y
Wrote to C:\Users\PatoCV\Desktop\Avance del Proyecto\package.json:

{
  "name": "avance-del-proyecto",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "type": "commonjs"
}
```

Instalar las dependencias necesarias



```
C:\Users\PatoCV\Desktop\Avance del Proyecto>npm install express jsonwebtoken bcryptjs cors dotenv mysql2

added 93 packages, and audited 94 packages in 10s

26 packages are looking for funding
  run `npm fund` for details

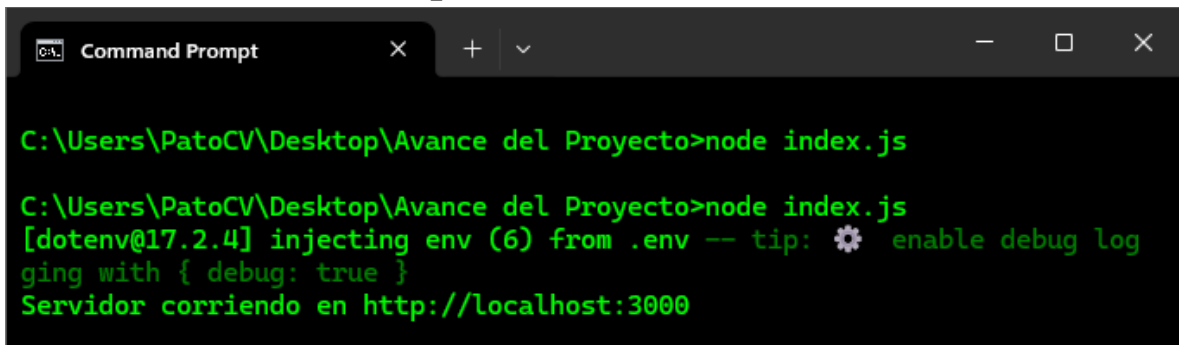
found 0 vulnerabilities
```

Estructura del proyecto

- index.js → punto de entrada del servidor
- package.json → lo creó npm init
- .env → variables de entorno (credenciales, claves)
- .gitignore → evitar subir cosas sensibles

B: BACKEND (API RESTFUL CON EXPRESS.JS)

Servidor básico en Express



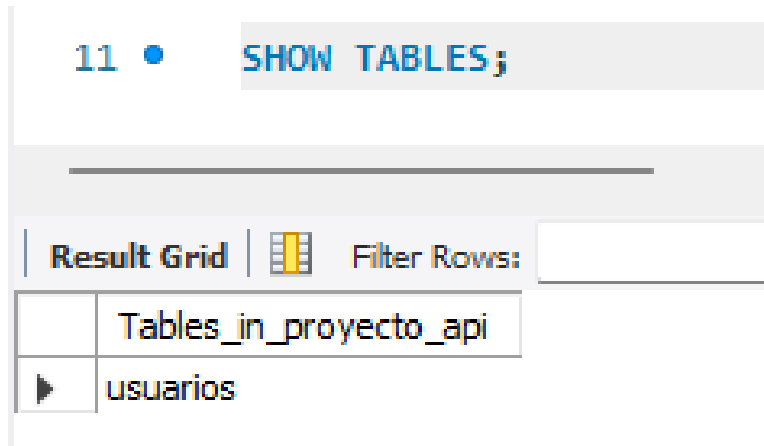
```
C:\Users\PatoCV\Desktop\Avance del Proyecto>node index.js

C:\Users\PatoCV\Desktop\Avance del Proyecto>node index.js
[dotenv@17.2.4] injecting env (6) from .env — tip: ⚙️ enable debug logging with { debug: true }
Servidor corriendo en http://localhost:3000
```

Crear la base de datos (MySQL)

```
1 • CREATE DATABASE proyecto_api;
2 • USE proyecto_api;
3
4 • CREATE TABLE usuarios (
5     id INT AUTO_INCREMENT PRIMARY KEY,
6     nombre VARCHAR(100) NOT NULL,
7     email VARCHAR(100) NOT NULL,
8     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
9 );
```

Comprobación

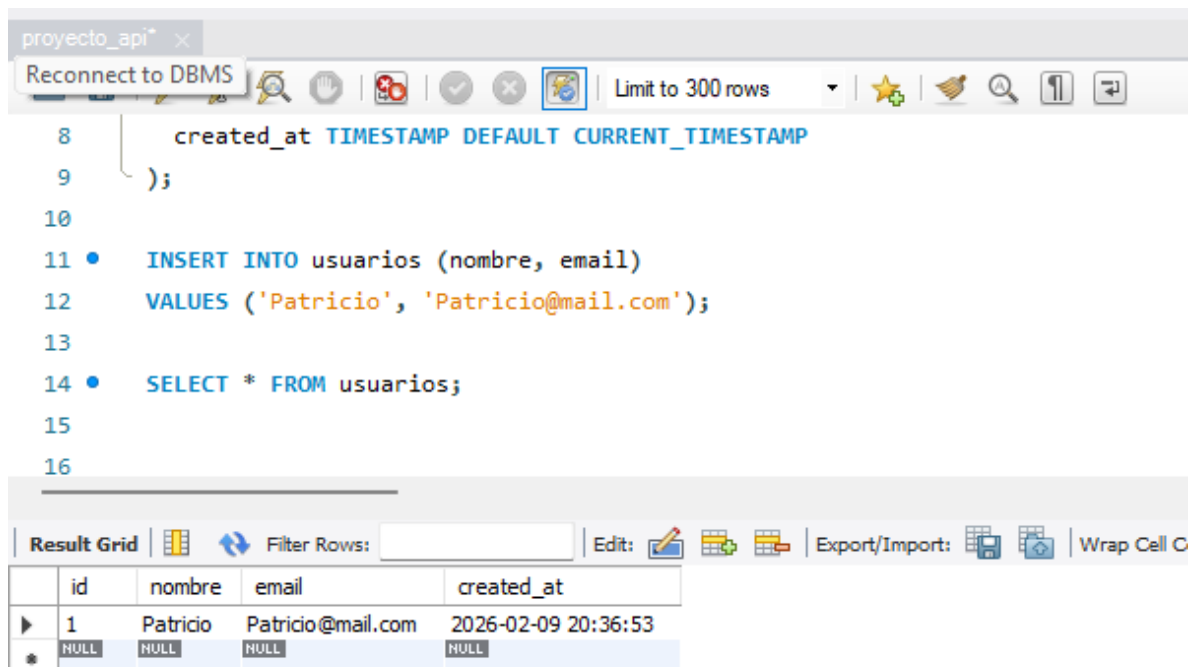


Levantar Servidor Backend

```
Command Prompt - node inc X + v
Microsoft Windows [Version 10.0.26100.7462]
(c) Microsoft Corporation. All rights reserved.

C:\Users\PatoCV\Desktop\Avance del Proyecto>node index.js
[dotenv@17.2.4] injecting env (6) from .env — tip: 📁 backup and recover secrets: https://dotenvx.com/ops
Servidor corriendo en http://localhost:3000
```

Insertar datos a la tabla



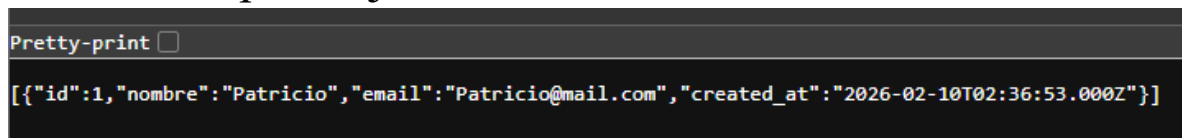
The screenshot shows a database client window titled "proyecto_api". The SQL editor contains the following code:

```
8      created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
9  );
10
11 • INSERT INTO usuarios (nombre, email)
12   VALUES ('Patricio', 'Patricio@mail.com');
13
14 • SELECT * FROM usuarios;
15
16
```

Below the editor is the "Result Grid" showing the results of the SELECT query. The grid has columns: id, nombre, email, and created_at. The first row shows the inserted data, and the second row shows NULL values.

	id	nombre	email	created_at
▶	1	Patricio	Patricio@mail.com	2026-02-09 20:36:53
*	NULL	NULL	NULL	NULL

Servidor Express ejecutándose



The screenshot shows a terminal window with a "Pretty-print" checkbox. The output is a JSON array representing the inserted user data.

```
[[{"id":1,"nombre":"Patricio","email":"Patricio@mail.com","created_at":"2026-02-10T02:36:53.000Z"}]]
```

Middleware Personalizado

```
middlewares > JS auth.middleware.js > ...
1  const jwt = require('jsonwebtoken');
2
3  function authMiddleware(req, res, next) {
4    const authHeader = req.headers.authorization;
5
6    if (!authHeader) {
7      return res.status(401).json({ mensaje: 'Token no proporcionado' });
8    }
9
10   const token = authHeader.split(' ')[1];
11
12   try {
13     const decoded = jwt.verify(token, process.env.JWT_SECRET);
14     req.usuario = decoded;
15     next();
16   } catch (error) {
17     return res.status(401).json({ mensaje: 'Token inválido' });
18   }
19 }
20
21 module.exports = authMiddleware;
22
```


C: FRONTEND (HTML, CSS, JAVASCRIPT)

Desarrollo del Frontend

```
public > login.html > html
1  <!DOCTYPE html>
2  <html lang="es">
3  <head>
4    <meta charset="UTF-8">
5    <title>Login</title>
6    <link rel="stylesheet" href="styles.css">
7  </head>
8
9  Click to add a breakpoint
10 <div class="container">
11   <h2>Iniciar sesión</h2>
12
13   <form id="loginForm">
14     <input type="email" id="email" placeholder="Correo electrónico">
15     <input type="password" id="password" placeholder="Contraseña">
16     <button type="submit">Entrar</button>
17   </form>
18
19   <p id="mensaje"></p>
20   <p>¿No tienes cuenta? <a href="registro.html">Regístrate aquí</a>
21 </div>
22
23 <script src="app.js"></script>
24 </body>
25 </html>
```

```
public > dashboard.html > ...
1  <!DOCTYPE html>
2  <html lang="es">
3  <head>
4    <meta charset="UTF-8">
5    <title>Dashboard</title>
6    <link rel="stylesheet" href="styles.css">
7  </head>
8  <body>
9
10 <div class="container">
11   <h2>Usuarios registrados</h2>
12   <ul id="listaUsuarios"></ul>
13   <button onclick="cerrarSesion()">Cerrar sesión</button>
14 </div>
15
16 <script src="app.js"></script>
17 </body>
18 </html>
```

```
public > register.html > @html
1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4 <meta charset="UTF-8">
5 <title>Dashboard</title>
6 <link rel="stylesheet" href="styles.css">
7 <style>
8   .dashboard-container {
9     width: 600px;
10    max-width: 90%;
11  }
12
13  .user-list {
14    list-style: none;
15    padding: 0;
16    max-height: 400px;
17    overflow-y: auto;
18  }
19
20  .user-list li {
21    background: #f8f9fa;
22    padding: 15px;
23    margin: 10px 0;
24    border-radius: 8px;
25    border-left: 4px solid #007bff;
26  }
27
28  .actions {
29    display: flex;
30    justify-content: space-between;
31    margin-top: 20px;
32  }
33
34  .btn-secondary {
```

Diseño Visual con CSS

```
public > # styles.css > ...
1 * {
2   box-sizing: border-box;
3 }
4
5 body {
6   font-family: 'Segoe UI', Arial, sans-serif;
7   background: linear-gradient(135deg, #e9eef5, #f5f7fa);
8   height: 100vh;
9   margin: 0;
10
11   display: flex;
12   justify-content: center;
13   align-items: center;
14 }
15
16 .container {
17   width: 340px;
18   background: #ffffff;
19   padding: 30px;
20   border-radius: 12px;
21   box-shadow: 0 15px 30px rgba(0, 0, 0, 0.12);
22 }
23
24 h2 {
25   text-align: center;
26   margin-bottom: 20px;
27   color: #333;
28 }
29
30 input {
31   width: 100%;
32   padding: 12px;
33   margin-top: 12px;
34   border-radius: 6px;
```

D: DESPLIEGUE

Comprobación de funcionamiento

The image displays two screenshots of a web application running on a browser at localhost:3000.

The top screenshot shows the "Login" page. The browser tab is titled "Login" and the address bar shows "localhost:3000/login.html". The page features a white card with the title "Iniciar sesión". It contains two input fields: "Correo electrónico" and "Contraseña". Below these fields is a blue button labeled "Entrar". At the bottom of the card, there is a link: "¿No tienes cuenta? [Regístrate aquí](#)".

The bottom screenshot shows the "Registro" page. The browser tab is titled "Registro" and the address bar shows "localhost:3000/registro.html". The page features a white card with the title "Registro de Usuario". It contains three input fields: "Nombre completo", "Correo electrónico", and "Contraseña (mínimo 6 caracteres)". Below these fields is a blue button labeled "Registrarse". At the bottom of the card, there is a link: "¿Ya tienes cuenta? [Inicia sesión aquí](#)".

