



S.E.P. TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO de Tuxtepec

**PROGAMACION EN AMBIENTE CLIENTE-
SERVIDOR**

“Mini poke Api”

PRESENTA:

EUGENIO SIXTO NAYELI 22350398

DOCENTE:

DR. JULIO AGUILAR CARMONA

CARRERA:

INGENIERÍA INFORMÁTICA

AGOSTO/DICIEMBRE 2025

INTRODUCCION

El desarrollo de aplicaciones web modernas requiere la integración adecuada entre la interfaz gráfica, la lógica del servidor y el almacenamiento de datos. En este contexto, el proyecto Mini Pokémon fue diseñado con el propósito de poner en práctica los conocimientos adquiridos en la materia de Programación en Ambiente Cliente-Servidor.

El sistema se basa en una arquitectura REST, utilizando Node.js y Express.js para la creación del backend, así como una base de datos relacional MySQL para el almacenamiento persistente de la información. Además, se integra la API pública PokeAPI para obtener datos oficiales de los Pokémon, lo cual permite optimizar recursos y enriquecer la experiencia del usuario.

Tecnologías implementadas

Node.js permite ejecutar JavaScript en el servidor con un modelo asíncrono, lo que mejora el rendimiento del sistema. Express.js facilita la creación de endpoints REST y el manejo de rutas.

El frontend fue desarrollado con HTML5, CSS3 y JavaScript, permitiendo una interfaz dinámica y responsiva. La base de datos MySQL asegura la integridad y consistencia de los datos, mientras que PokeAPI provee información externa actualizada sobre los Pokémon.

Estructura del proyecto

El proyecto se divide en dos partes principales: backend y frontend. El backend se organiza en carpetas como controllers, models, routes y config, siguiendo el patrón MVC. Esta separación facilita el mantenimiento y la escalabilidad.

El frontend consume la API mediante peticiones HTTP y presenta la información de forma interactiva al usuario.

Funcionamiento del backend

El servidor inicia desde el archivo principal app.js, donde se configuran los middlewares, las rutas y el puerto de escucha. Los controladores contienen la lógica de negocio, validan datos y se comunican con la base de datos. Los modelos definen las consultas SQL necesarias para interactuar con las tablas de usuarios, partidas y Pokémon.

Funcionamiento del Frontend

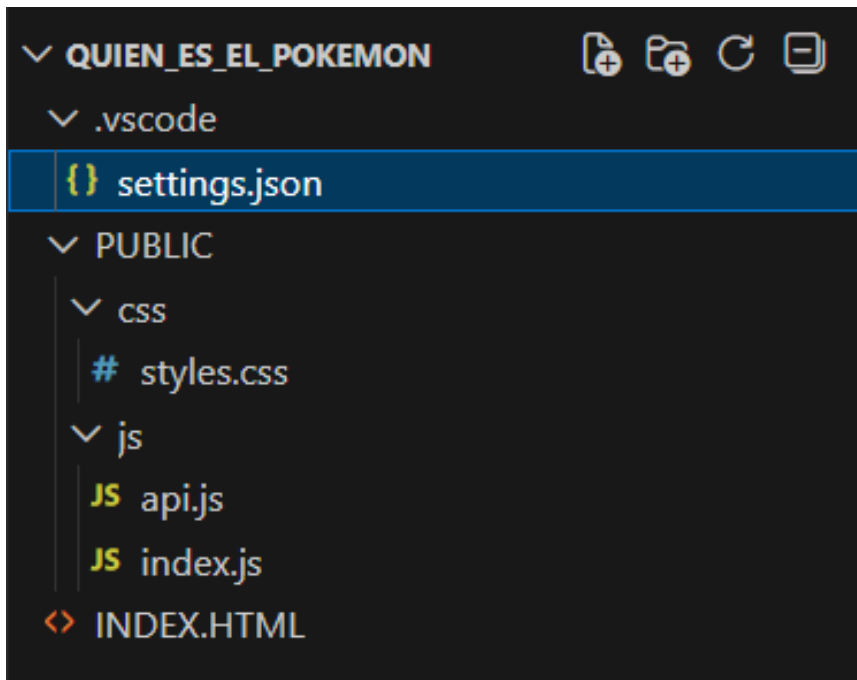
La interfaz gráfica permite al usuario interactuar con el juego ¿Quién es ese Pokémon?. Mediante JavaScript se gestionan los eventos, se muestran opciones aleatorias y se valida la respuesta seleccionada.

El diseño visual se apoya en CSS3, ofreciendo una experiencia atractiva y clara

Reporte del Sistema Mini Pokémon

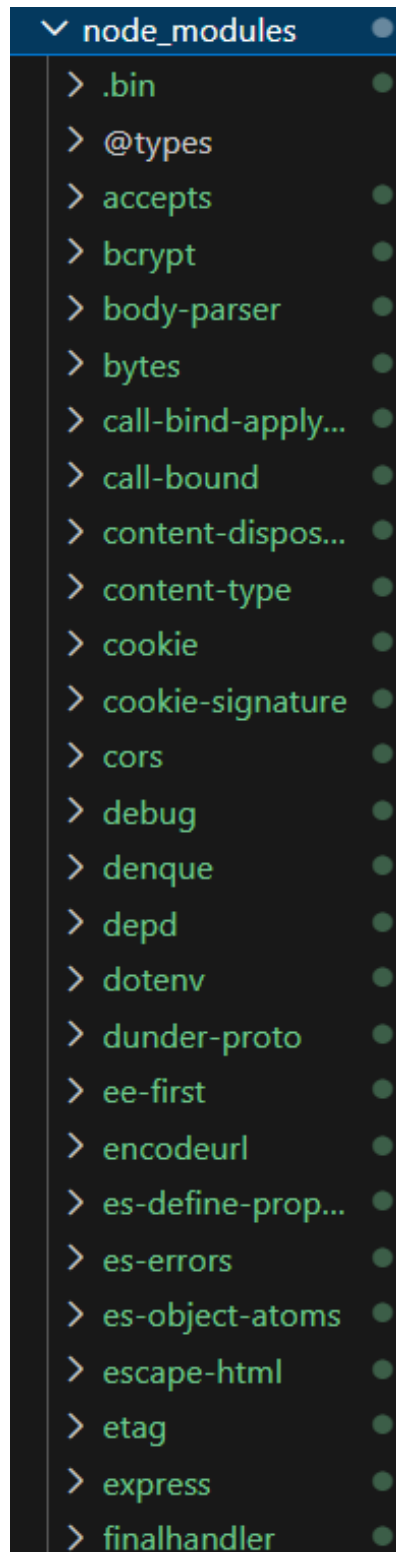


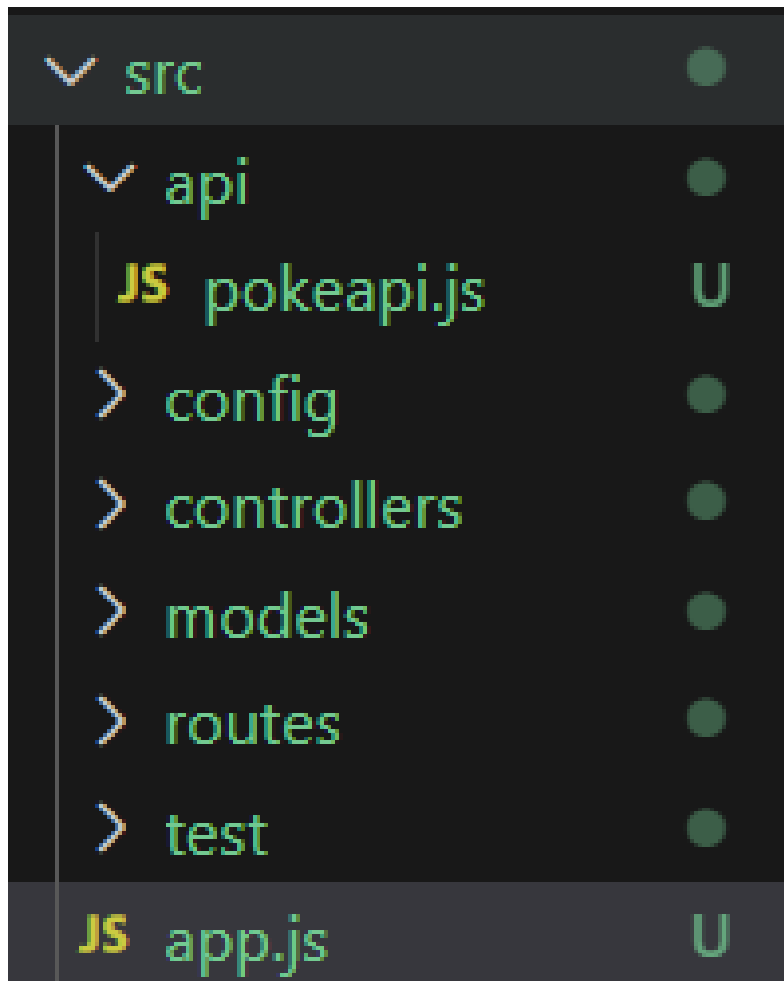
En esta imagen se aprecia el código principal del servidor, encargado de inicializar Express, registrar rutas y habilitar middlewares necesarios para la comunicación cliente-servidor.



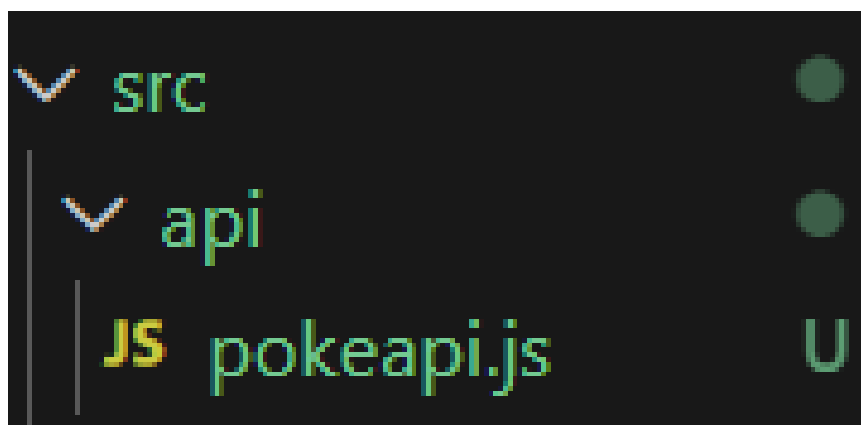
Aquí se observa la configuración de conexión a la base de datos, donde se centralizan las credenciales y parámetros necesarios para acceder a MySQL de forma segura.

La captura corresponde a un controlador, el cual recibe las peticiones HTTP, valida la información y coordina las operaciones con la base de datos.

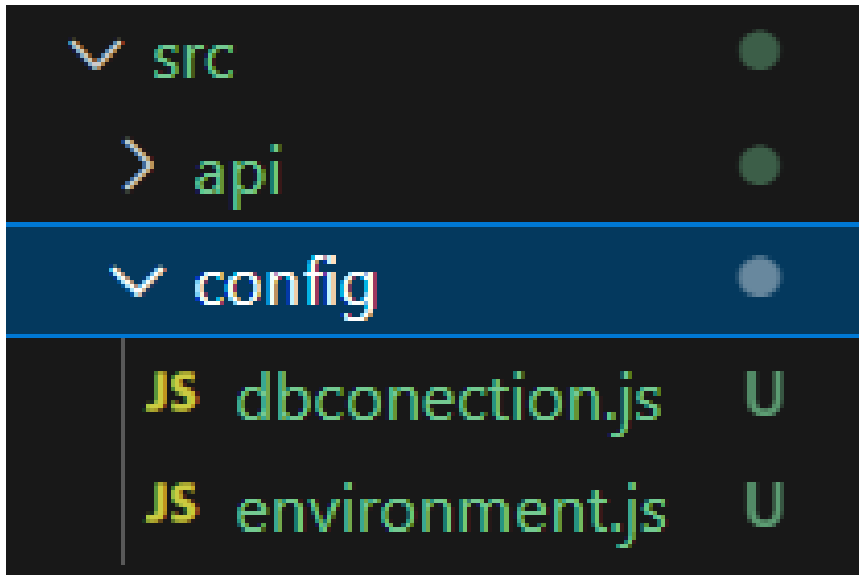




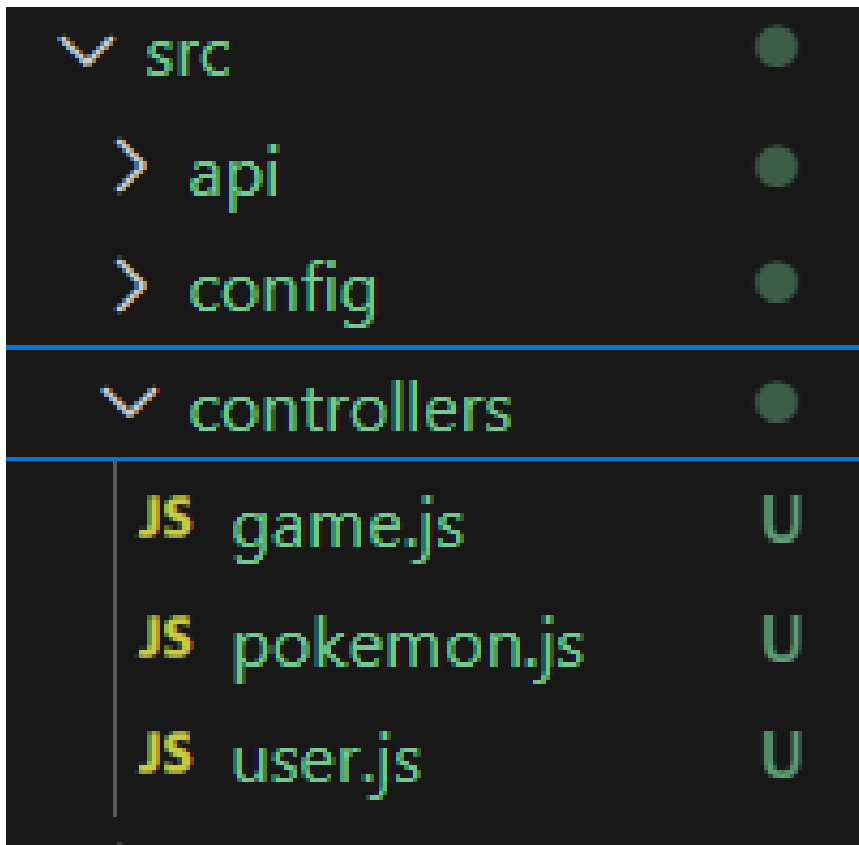
En esta imagen se muestra la interfaz gráfica del sistema en ejecución, donde el usuario interactúa con el juego y selecciona la respuesta correcta.



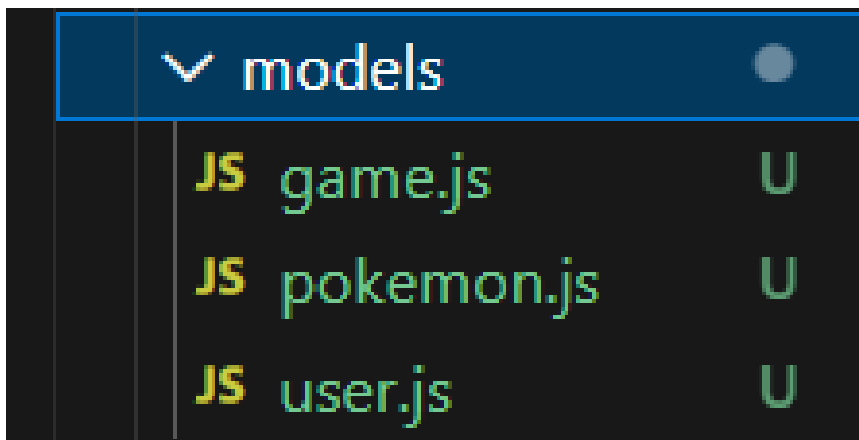
La siguiente captura muestra la estructura general del proyecto, donde se observa la separación entre backend y frontend, lo cual permite un mejor orden del código y facilita su mantenimiento.



En esta imagen se aprecia el código principal del servidor, encargado de inicializar Express, registrar rutas y habilitar middlewares necesarios para la comunicación cliente-servidor.



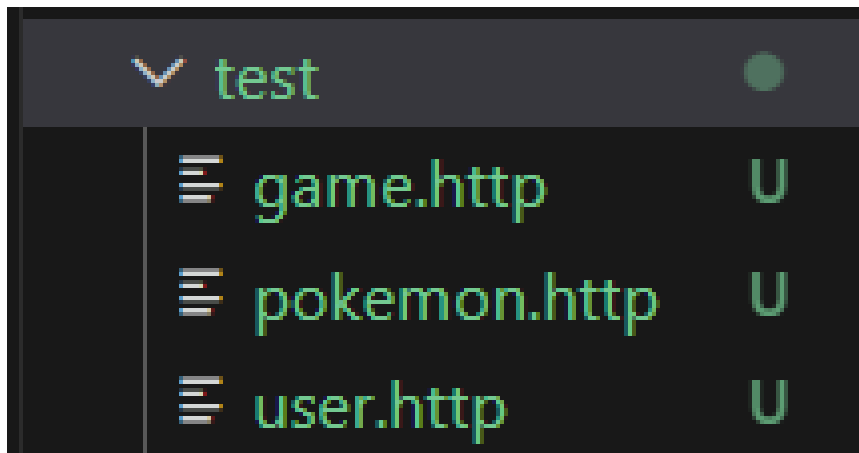
Aquí se observa la configuración de conexión a la base de datos, donde se centralizan las credenciales y parámetros necesarios para acceder a MySQL de forma segura.



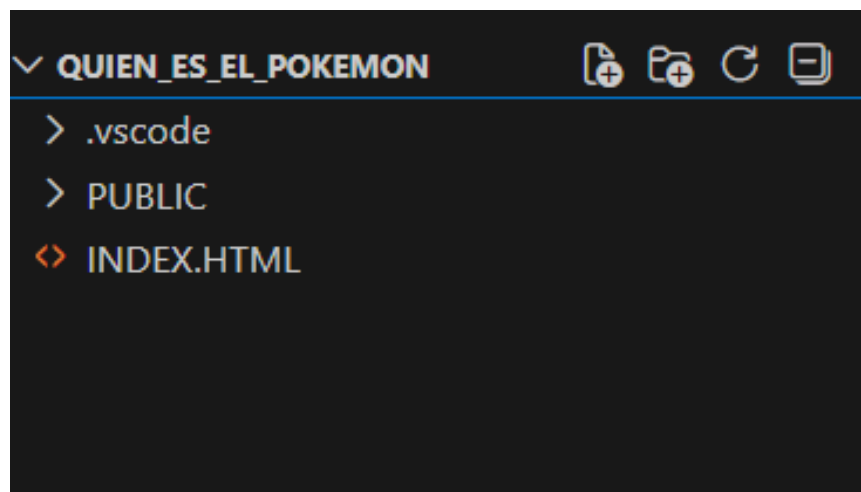
La captura corresponde a un controlador, el cual recibe las peticiones HTTP, valida la información y coordina las operaciones con la base de datos.

✓	routes	●
JS	game.js	U
JS	pokemon.js	U
JS	user.js	U

En esta imagen se muestra la interfaz gráfica del sistema en ejecución, donde el usuario interactúa con el juego y selecciona la respuesta correcta.



La siguiente captura muestra la estructura general del proyecto, donde se observa la separación entre backend y frontend, lo cual permite un mejor orden del código y facilita su mantenimiento.

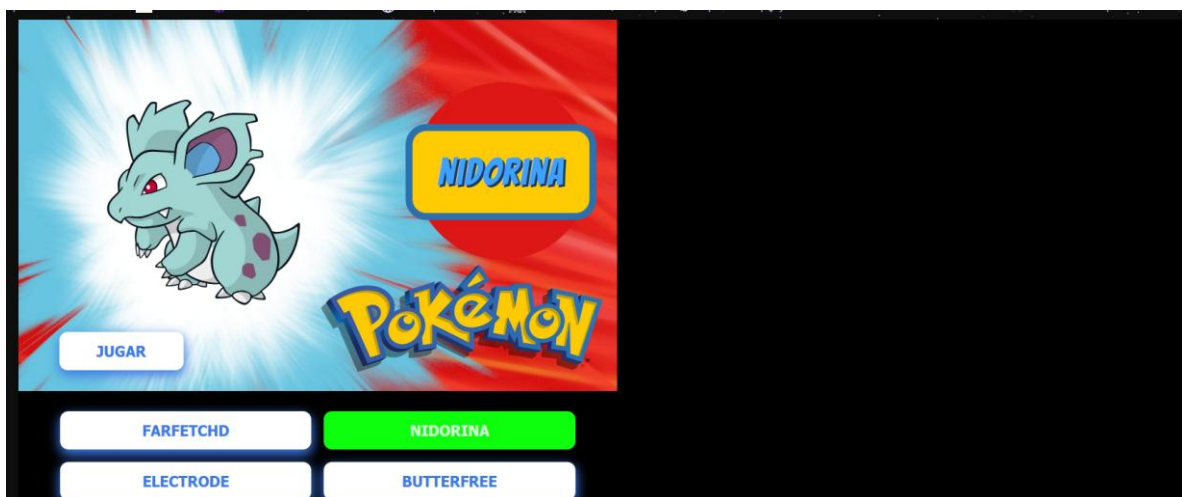
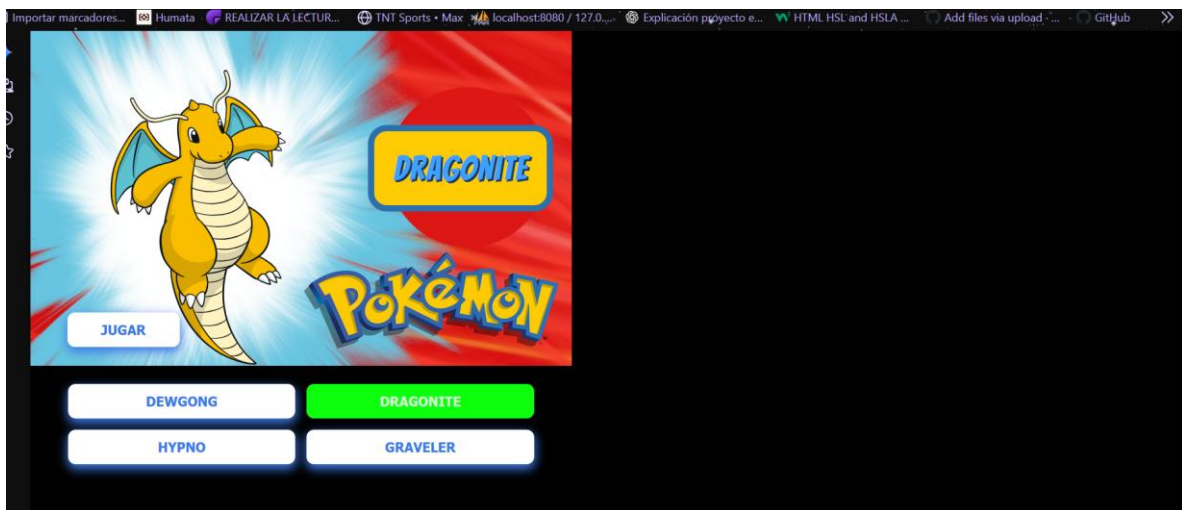


```

<> INDEX.HTML X
<> INDEX.HTML > ...
1  <!DOCTYPE html>
2  <html lang="es">
3
4  <head>
5      <meta charset="UTF-8">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <link rel="stylesheet" href="./PUBLIC/css/styles.css">
8      <link rel="stylesheet" href="https://fonts.googleapis.com/css2?family=Bangers&display=swap">
9
10
11     <title>¿QUIEN ES ESTE POKEMON?</title>
12 </head>
13
14 <body>
15     <main class="fetching">
16         <div id="pokemon-container">
17             
19             <img id="pokemon-image" src="" />
20         </div>
21
22         <div id="answer">
23             <div id="bg-overlay"> </div>
24             <div id="text-overlay"> </div>
25         </div>
26         <section id="controls">
27             <button id="play">JUGAR</button>
28             <div id="choices"></div>
29         </section>
30
31     <script src="./public/js/api.js"> </script>
32     <script src="./PUBLIC/js/index.js"> </script>
33
34 </body>
35 </html>

```

En esta imagen se aprecia el código principal del servidor, encargado de inicializar Express, registrar rutas y habilitar middlewares necesarios para la comunicación cliente-servidor.



Se muestra la creación del juego donde se presenta la combinación de adivinar el pokémon en el cual si es error te marcara color rojo y si es el correcto nos marcara color verde

CONCLUSION

El desarrollo del sistema Mini Pokémon permitió aplicar de manera práctica los conocimientos adquiridos en la materia de Programación en Ambiente Cliente-Servidor, integrando conceptos fundamentales como la arquitectura en capas, el consumo de APIs REST y el uso de bases de datos relacionales. A lo largo del proyecto se logró una correcta separación entre el frontend, el backend y la base de datos, lo cual facilita el mantenimiento del sistema y permite que este pueda escalarse o modificarse sin afectar su funcionamiento general.

La implementación del backend con Node.js y Express.js demostró la importancia de contar con un servidor eficiente y organizado, capaz de gestionar múltiples peticiones, validar información y comunicarse de forma segura con la base de datos MySQL. Asimismo, el uso de controladores, rutas y modelos permitió estructurar el código de manera clara, siguiendo buenas prácticas de desarrollo de software y mejorando la legibilidad del proyecto.