

Semana 2. Desarrollo inicial

Jessica Lorena Galviz López

SENA

Análisis y desarrollo de software

2977395

Pasto

2025

**TABLA DE CONTENIDO**

1. Introducción
2. Carpeta comprimida sobre lo realizado en Visual Studio
3. Códigos de registro.html y registro.js
4. Imágenes
5. Códigos de habitaciones.html y habitaciones.js
6. Imágenes
7. Códigos de reservas.html y reservas.js
8. Imágenes
9. Comentarios de los archivos
10. Conclusión

**INTRODUCCION**

En este proyecto trabaje en Visual Studio para desarrollar la estructura básica de un sistema de Gestión para un Hotel. Para ello cree varios archivos que permite organizar y controlar diferentes aspectos del funcionamiento del Hotel. Los archivos **registro.html** y **registro.js** se encargan de gestionar la información de los usuarios que se registran en el sistema. Por otro lado, **habitaciones.html** y **habitaciones.js** permiten visualizar y manejar la información relacionada con las habitaciones disponibles, como su estado y características. Finalmente, **reservas.html** y **reservas.**js se enfoca en la gestión de reservas, permitiendo registrar y consultar las reservas realizadas por los usuarios. Cada archivocombina **HTML** y **JavaScript** para ofrecer una interacción dinámica y funcional dentro del sistema.

**CARPETA COMPRIMIDA SOBRE LA REALIZADO EN VISUAL STUDIO**

****

**CODIGOS DE REGISTRO.HTML Y REGISTRO.JS**

**Registro.html**

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<title>Registro de Reservas</title>

</head>

<body>

<h2>Registro de Reserva</h2>

<form id="form-registro">

<label>Nombre del cliente:</label>

<input type="text" id="nombre" required><br><br>

<label>Número de habitación:</label>

<input type="number" id="habitacion" required><br><br>

<label>Fecha de ingreso:</label>

<input type="date" id="fecha" required><br><br>

<button type="submit">Registrar</button>

</form>

<div id="resultado"></div>

<!-- Aquí conectas tu JS -->

<script src="registro.js"></script>

</body>

</html>

**Registro.js**

// registro.js

document.addEventListener("DOMContentLoaded", () => {

const formulario = document.getElementById("formRegistro");

formulario.addEventListener("submit", (e) => {

e.preventDefault();

const nombre = document.getElementById("nombre").value.trim();

const email = document.getElementById("email").value.trim();

const telefono = document.getElementById("telefono").value.trim();

const habitacion = document.getElementById("habitacion").value;

const contraseña = document.getElementById("contraseña").value;

if(nombre === "" || email === "" || telefono === "" || contraseña === "") {

alert("Por favor, completa todos los campos.");

return;

}

const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;

if(!emailRegex.test(email)) {

alert("Por favor, ingresa un correo válido.");

return;

}

const usuario = {

nombre,

email,

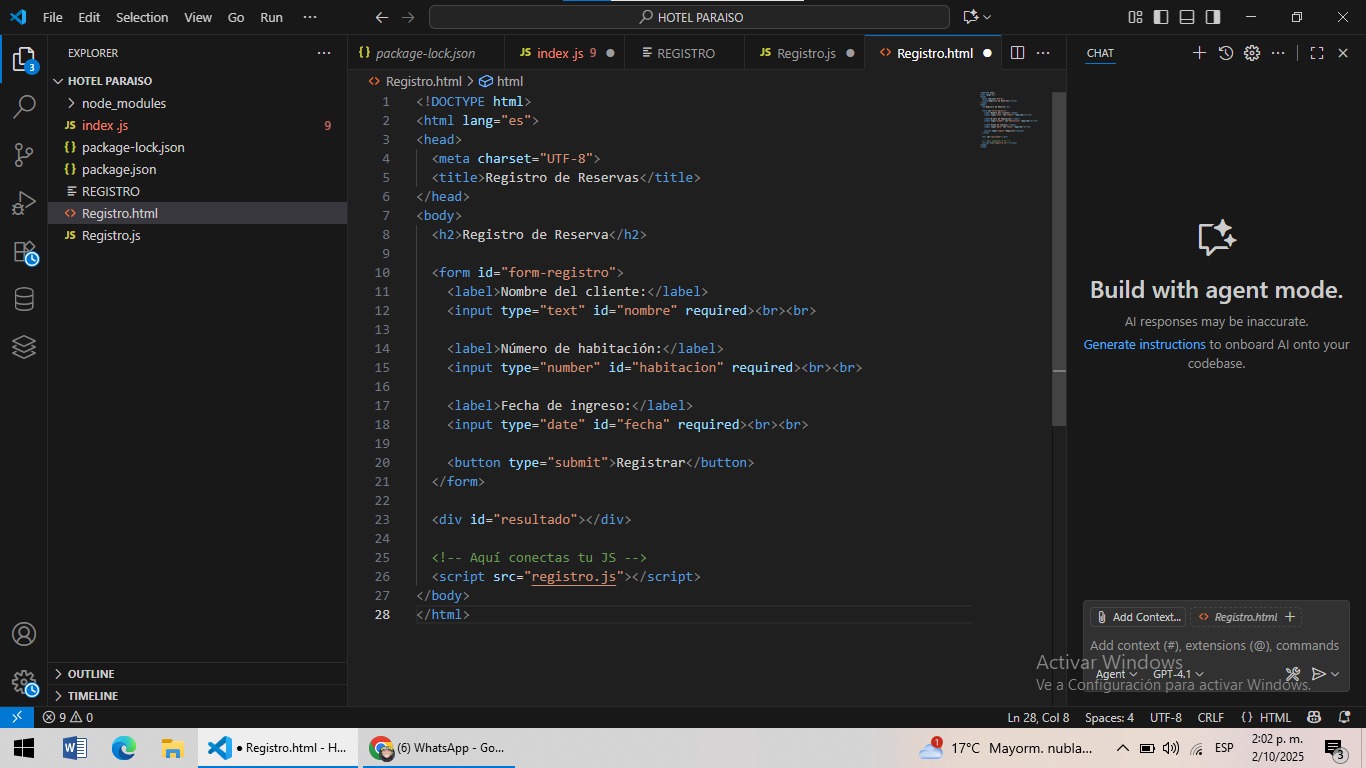
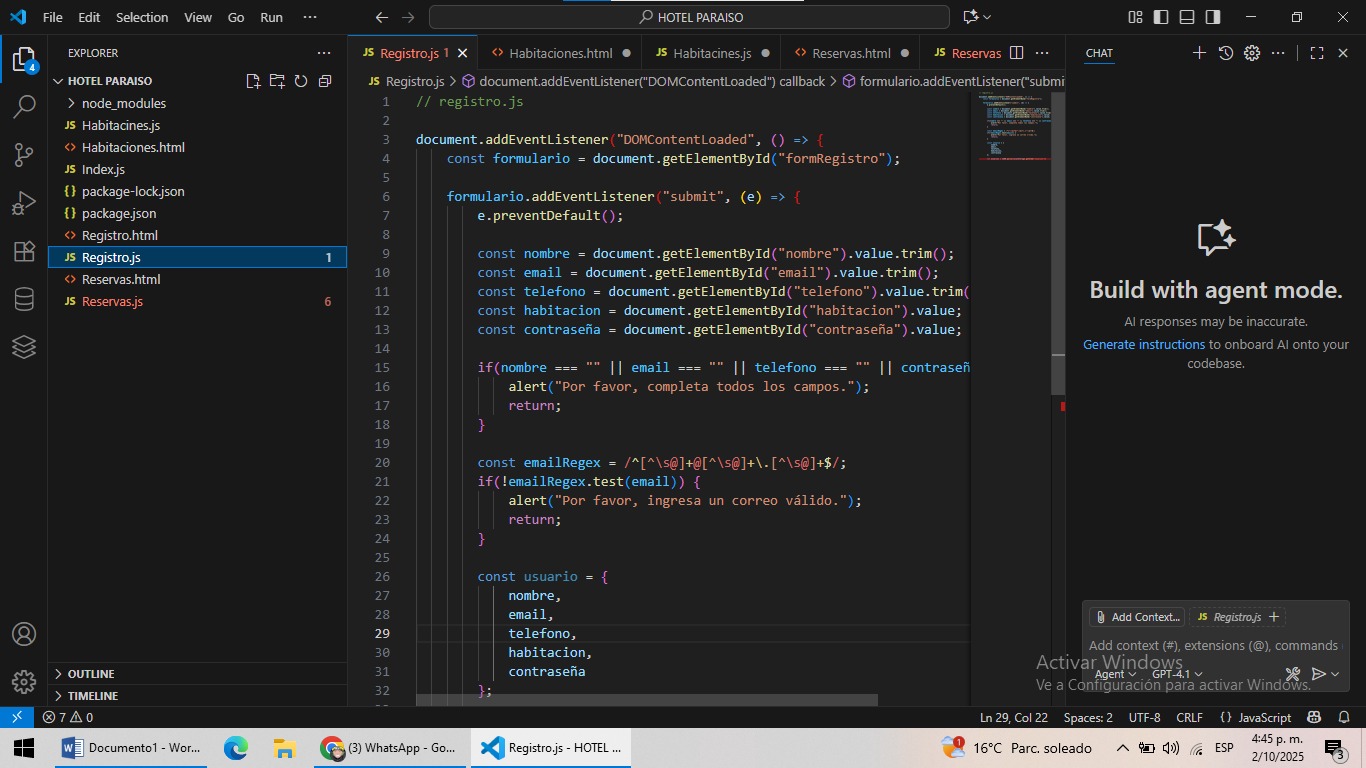
telefono,

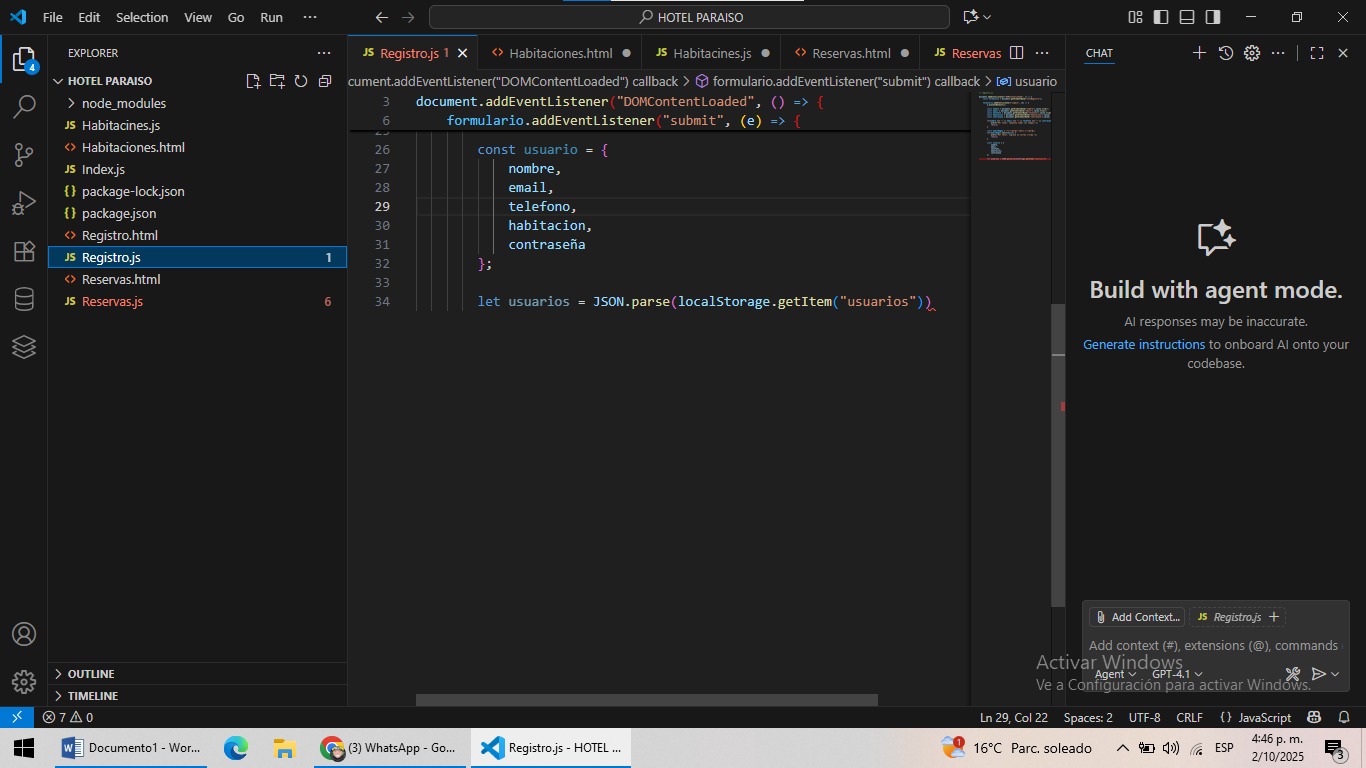
habitacion,

contraseña

};

let usuarios = JSON.parse(localStorage.getItem("usuarios"))

**IMÁGENES**

****

**CODIGOS DE HABITACONES.HTML Y HABITACIONES.JS**

**Habitaciones.html**

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<title>Habitaciones - Hotel Paraíso</title>

</head>

<body>

<h1>Gestión de Habitaciones</h1>

<div id="lista-habitaciones"></div>

<!-- Conexión al archivo JS -->

<script src="habitaciones.js"></script>

</body>

</html>

**Habitaciones.js**

// arreglo para almacenar las habitaciones

let habitaciones = [];

// función para agregar una habitación

function agregarHabitacion(numero, tipo, precio) {

const habitacion = {

numero: numero,

tipo: tipo,

precio: precio,

ocupada: false

};

habitaciones.push(habitacion);

console.log(Habitación ${numero} agregada correctamente.);

}

// función para mostrar todas las habitaciones

function mostrarHabitaciones() {

if (habitaciones.length === 0) {

console.log("No hay habitaciones registradas.");

return;

}

console.log("Listado de habitaciones:");

habitaciones.forEach(h => {

console.log(Número: ${h.numero}, Tipo: ${h.tipo}, Precio: ${h.precio}, Ocupada: ${h.ocupada});

});

}

// función para reservar una habitación

function reservarHabitacion(numero) {

const habitacion = habitaciones.find(h => h.numero === numero);

if (!habitacion) {

console.log(La habitación ${numero} no existe.);

return;

}

if (habitacion.ocupada) {

console.log(La habitación ${numero} ya está ocupada.);

return;

}

habitacion.ocupada = true;

console.log(Habitación ${numero} reservada correctamente.);

}

// función para liberar una habitación

function liberarHabitacion(numero) {

const habitacion = habitaciones.find(h => h.numero === numero);

if (!habitacion) {

console.log(La habitación ${numero} no existe.);

return;

}

if (!habitacion.ocupada) {

console.log(La habitación ${numero} ya está libre.);

return;

}

habitacion.ocupada = false;

console.log(Habitación ${numero} liberada correctamente.);

}

// Ejemplos de uso

agregarHabitacion(101, "Sencilla", 100000);

agregarHabitacion(102, "Doble", 150000);

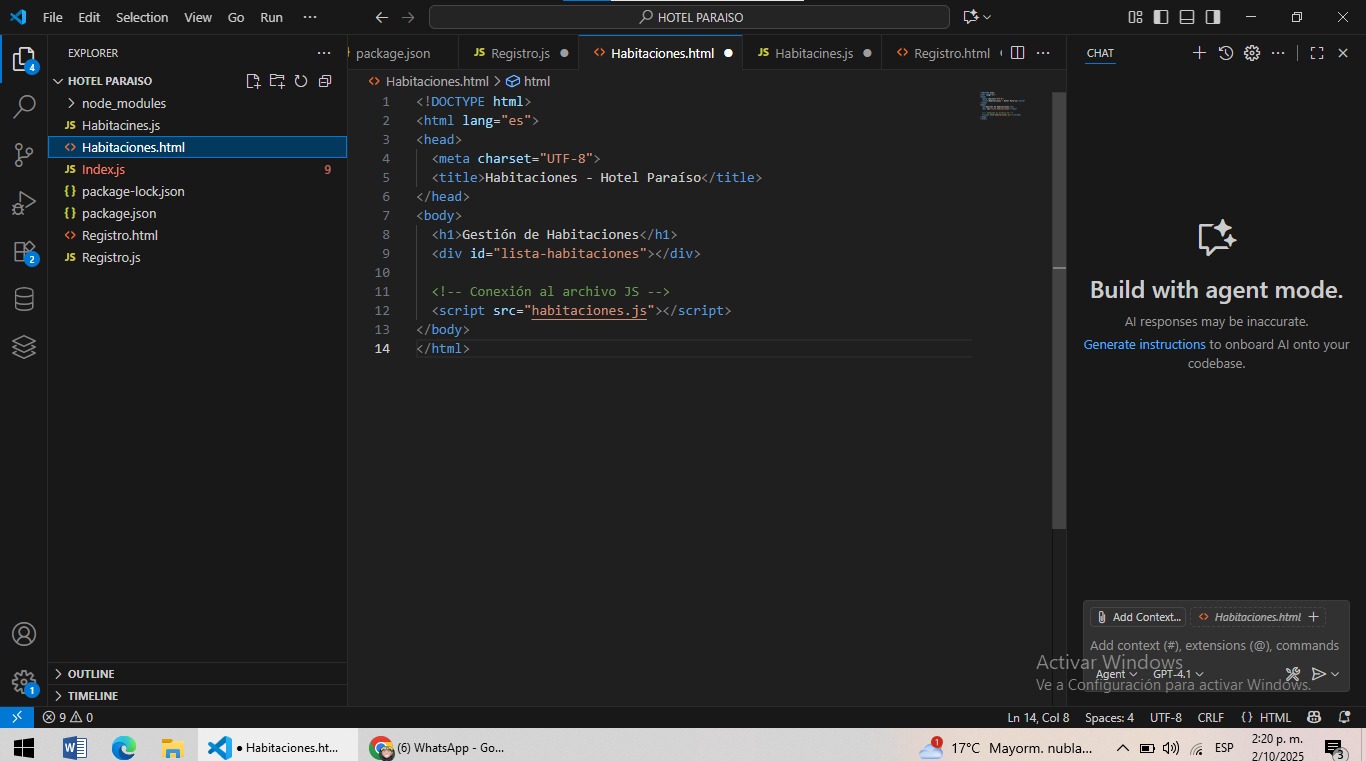
mostrarHabitaciones();

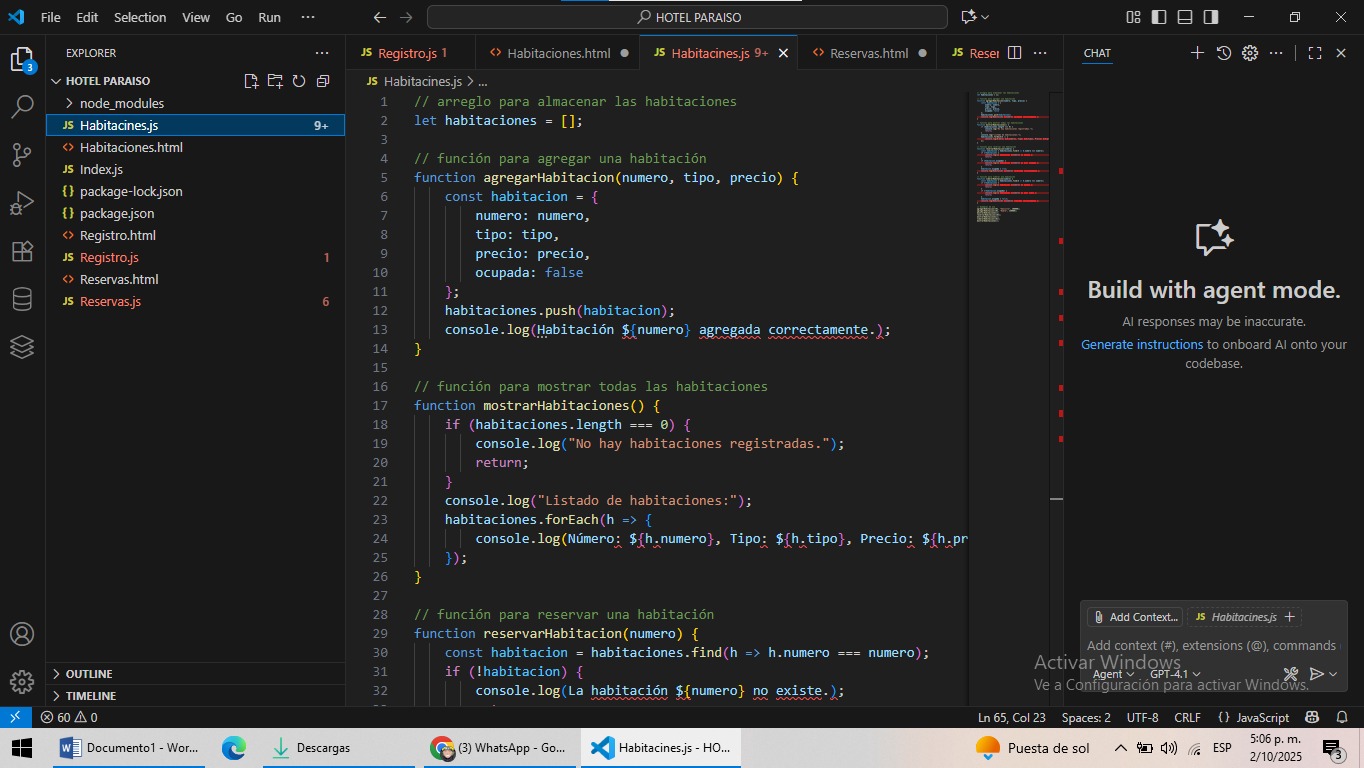
reservarHabitacion(101);

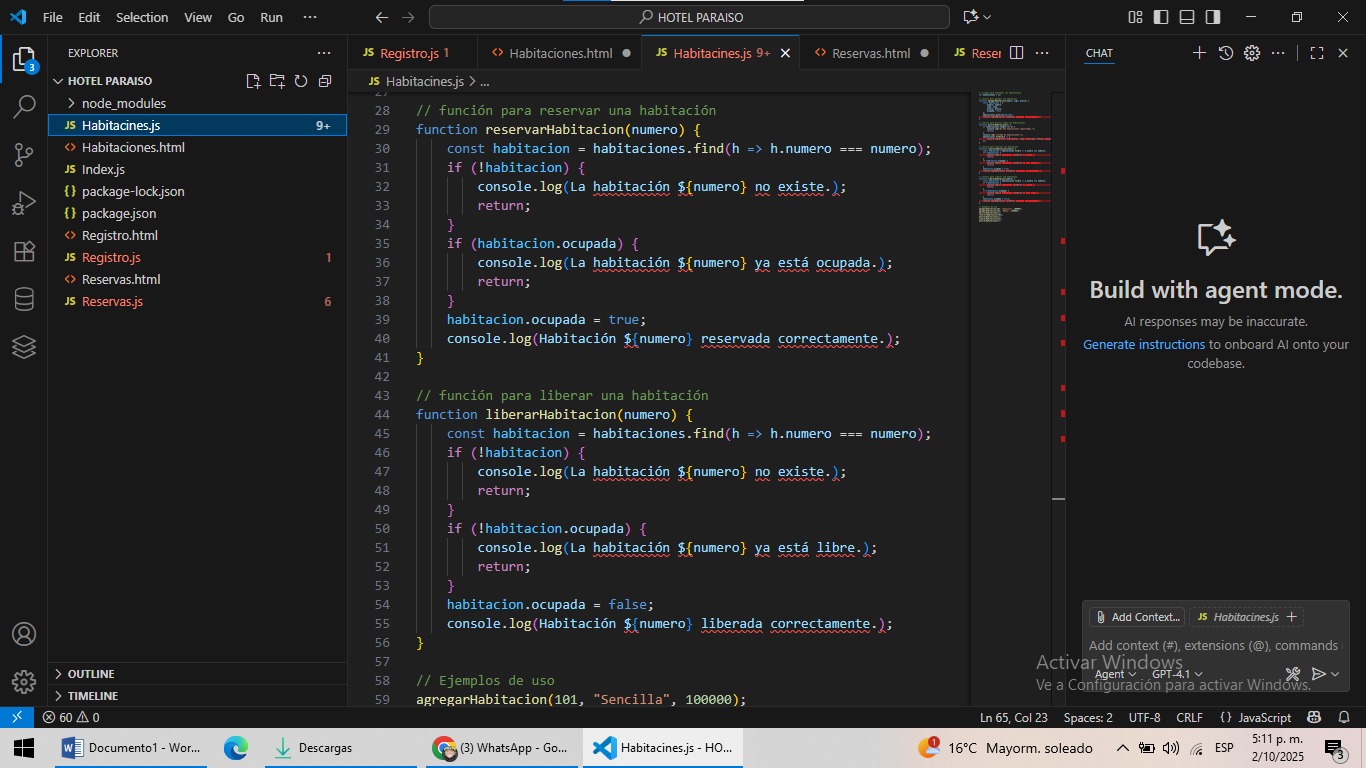
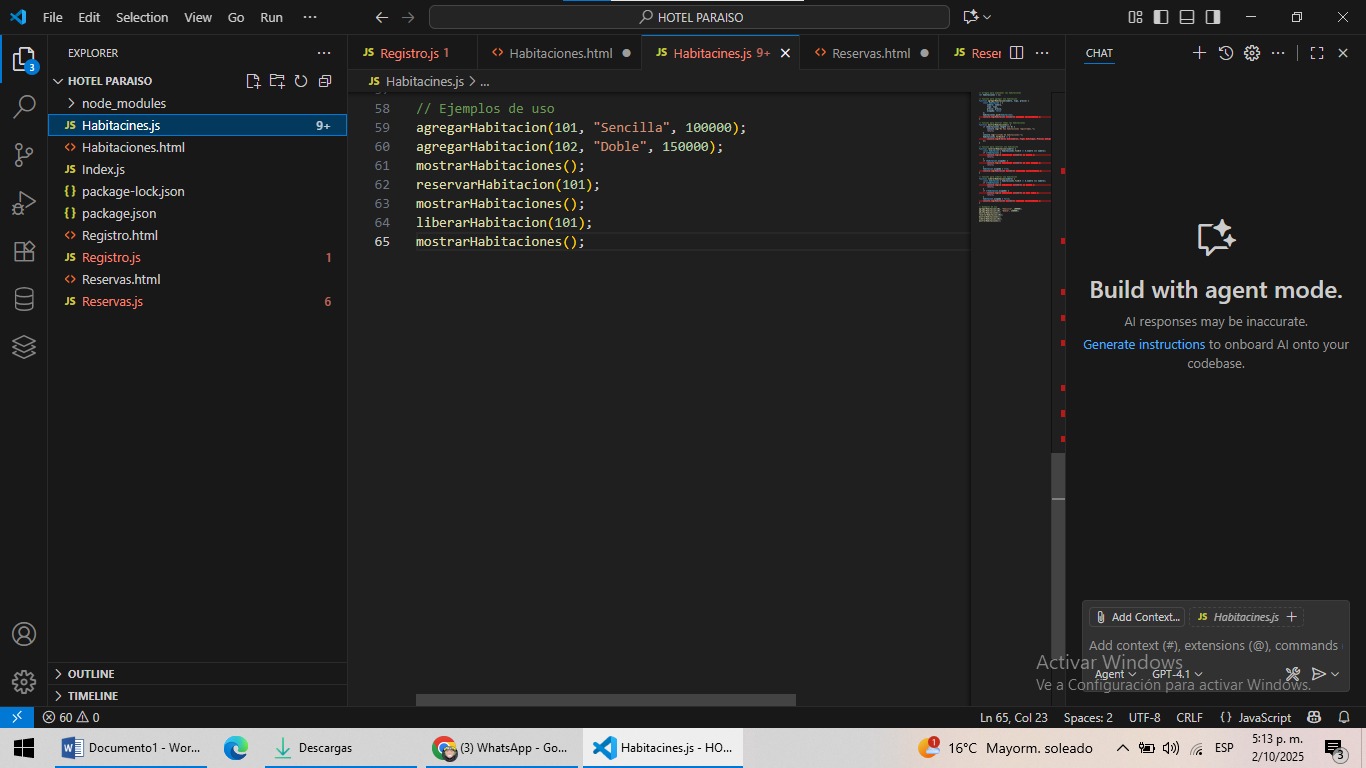
mostrarHabitaciones();

liberarHabitacion(101);

mostrarHabitaciones();

**IMÁGENES**

****

****

**CODIGOS DE RESEVAS.HTML Y RESERVAS.JS**

**Reservas.html**

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Reservas Hotel</title>

<link rel="stylesheet" href="reservas.css"> <!-- Si quieres agregar estilos -->

</head>

<body>

<h1>Reservas Hotel Paraíso</h1>

<form id="formReserva">

<label for="nombre">Nombre:</label>

<input type="text" id="nombre" required><br><br>

<label for="habitacion">Habitación:</label>

<select id="habitacion" required>

<option value="">Selecciona una habitación</option>

<option value="Sencilla">Sencilla</option>

<option value="Doble">Doble</option>

<option value="Suite">Suite</option>

</select><br><br>

<label for="fechaEntrada">Fecha de entrada:</label>

<input type="date" id="fechaEntrada" required><br><br>

<label for="fechaSalida">Fecha de salida:</label>

<input type="date" id="fechaSalida" required><br><br>

<button type="submit">Reservar</button>

</form>

<h2>Reservas registradas:</h2>

<ul id="listaReservas"></ul>

<script src="reservas.js"></script>

</body>

</html>

**Reservas.js**

// Seleccionamos los elementos del DOM

const formulario = document.getElementById('formReserva');

const lista = document.getElementById('listaReservas');

// Función para obtener reservas desde localStorage

function obtenerReservas() {

return JSON.parse(localStorage.getItem('reservas')) || [];

}

// Función para mostrar las reservas en la lista

function mostrarReservas() {

const reservas = obtenerReservas();

lista.innerHTML = ''; // Limpiar lista antes de mostrar

reservas.forEach((reserva, i) => {

const item = document.createElement('li');

item.textContent = ${reserva.nombre} - ${reserva.habitacion} - Entrada: ${reserva.fechaEntrada}, Salida: ${reserva.fechaSalida};

// Botón para eliminar

const btnEliminar = document.createElement('button');

btnEliminar.textContent = 'Eliminar';

btnEliminar.style.marginLeft = '10px';

btnEliminar.onclick = () => eliminarReserva(i);

item.appendChild(btnEliminar);

lista.appendChild(item);

});

}

// Función para agregar nueva reserva

formulario.addEventListener('submit', function(e) {

e.preventDefault(); // Evitar recargar la página

const nuevaReserva = {

nombre: document.getElementById('nombre').value,

habitacion: document.getElementById('habitacion').value,

fechaEntrada: document.getElementById('fechaEntrada').value,

fechaSalida: document.getElementById('fechaSalida').value

};

const reservas = obtenerReservas();

reservas.push(nuevaReserva);

localStorage.setItem('reservas', JSON.stringify(reservas));

formulario.reset(); // Limpiar formulario

mostrarReservas(); // Actualizar la lista

});

// Función para eliminar reserva por índice

function eliminarReserva(index) {

const reservas = obtenerReservas();

reservas.splice(index, 1);

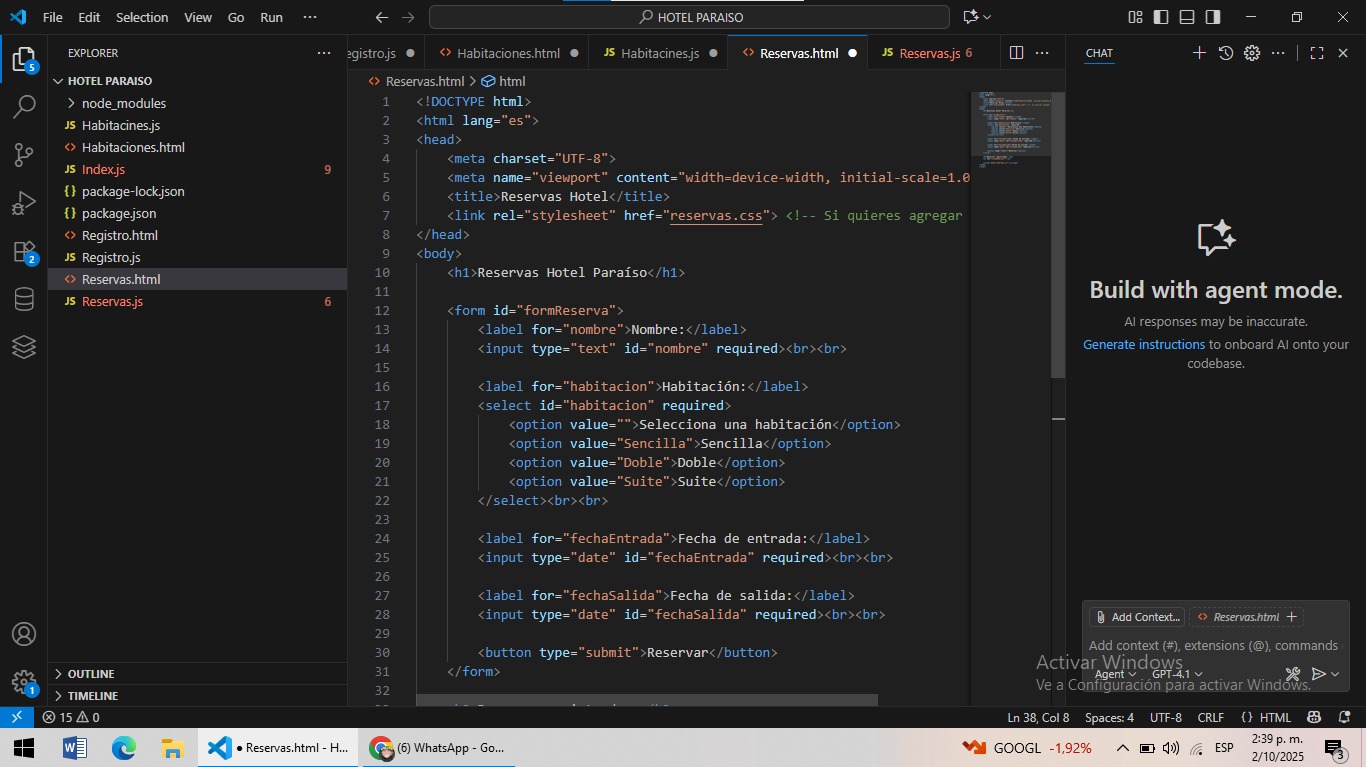
localStorage.setItem('reservas', JSON.stringify(reservas));

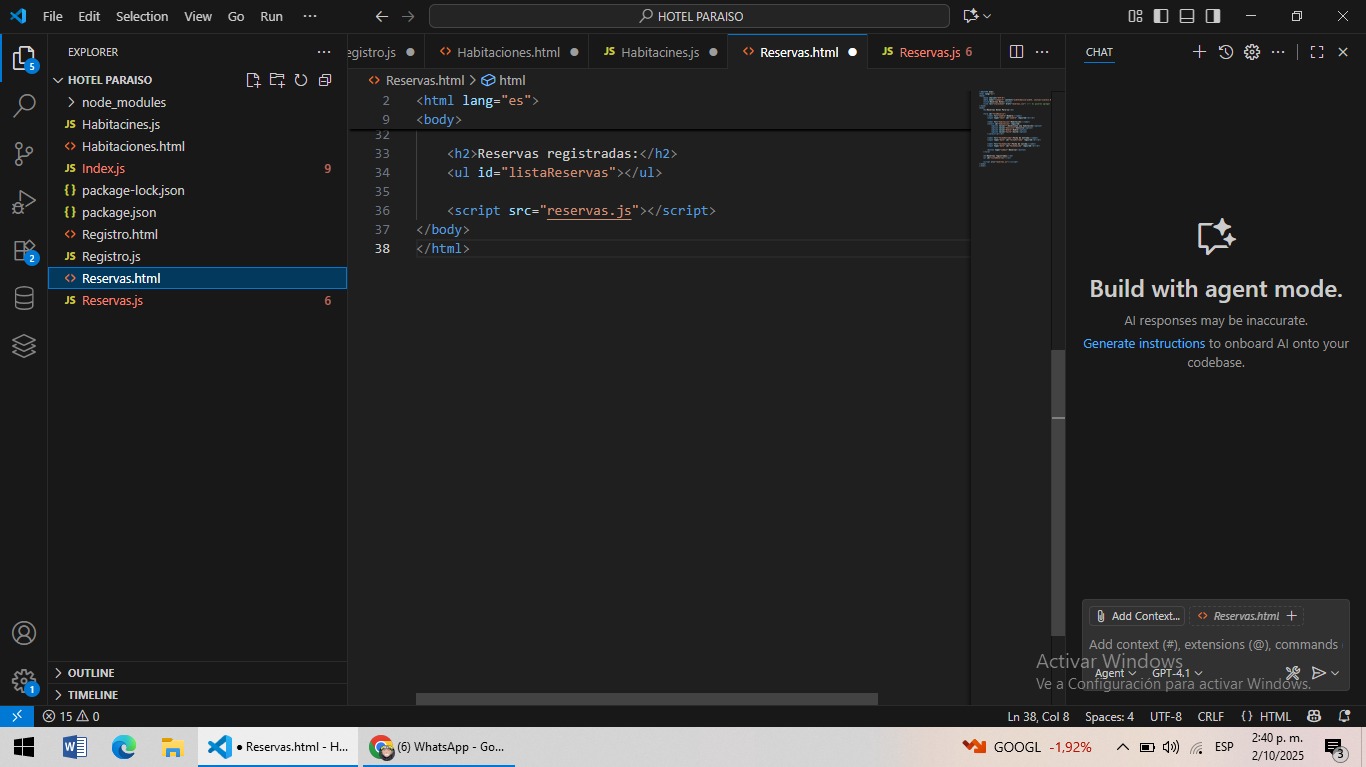
mostrarReservas();

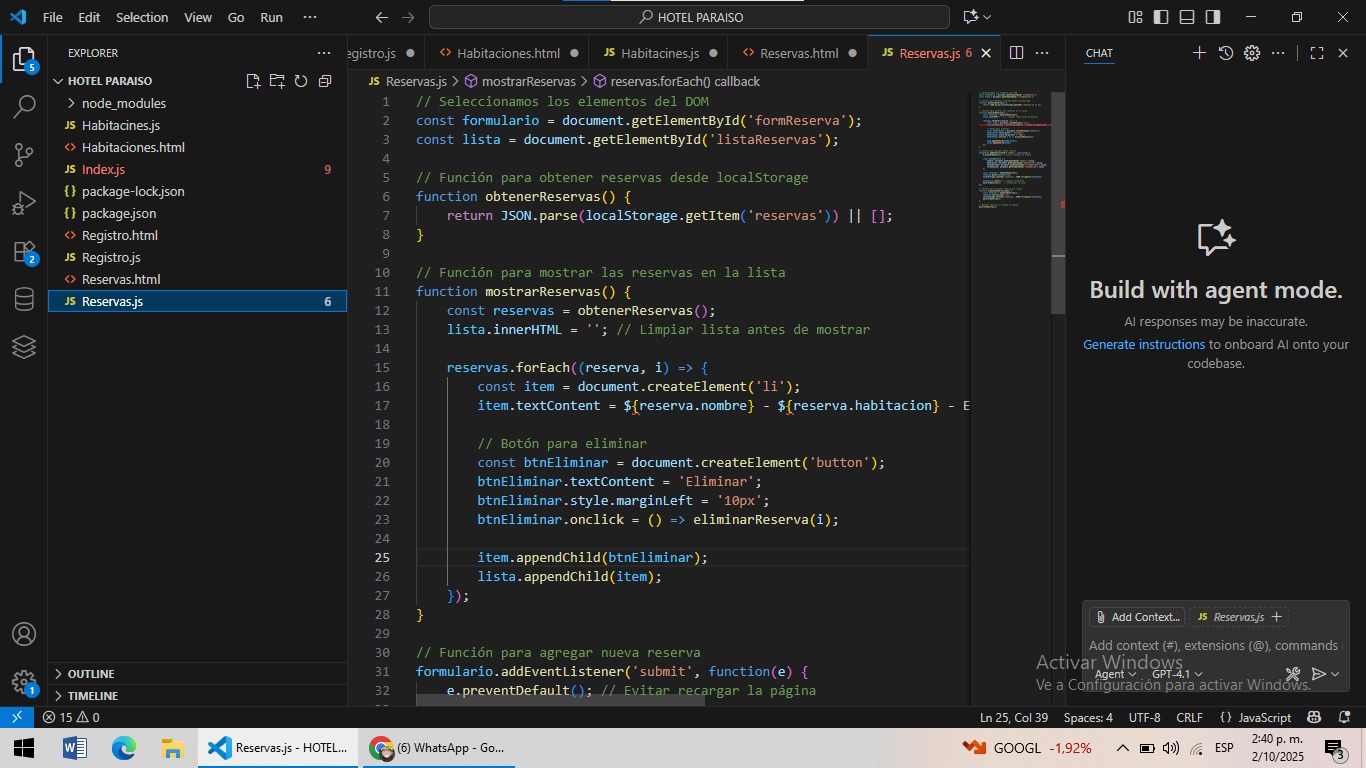
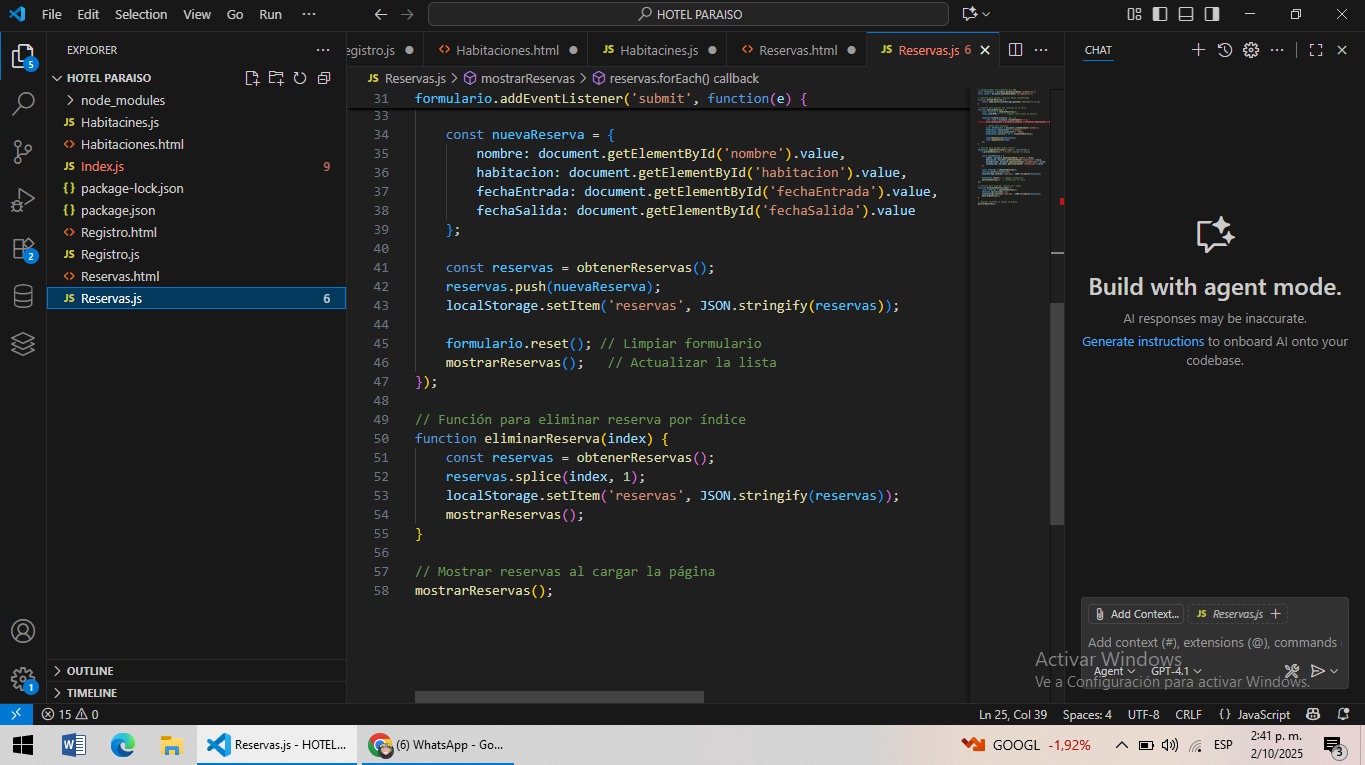
}

// Mostrar reservas al cargar la página

mostrarReservas();

**IMÁGENES**





**COMENTARIO DE LOS ARCHIVOS CREADOS**

En el desarrollo del proyecto Hotel Paraíso realizado en Visual Studio, se implementaron comentarios dentro de los archivos creados para el registro de clientes, la gestión de habitaciones y el control de reservas. Los comentarios cumplen la función de explicar de manera clara cada parte del código, permitiendo comprender su propósito y facilitar futuras modificaciones, gracias a esta práctica, se logra una mejor organización del trabajo, se evidencia el entendimiento de la lógica utilizada y se asegura que cualquier persona que revise el proyecto pueda entender con facilidad su funcionamiento.

**CONCLUSION**

En este proyecto se logró desarrollar un sistema básico de gestión para un hotel, que permite registrar habitaciones, gestionar reservas y controlar la ocupación de manera sencilla. A través de **HTML** y **JavaScript** se logró implementar la funcionalidad necesaria para almacenar, mostrar y actualizar la información de las habitaciones, brindando una herramienta práctica para la administración del hotel. Este trabajo permite comprender cómo interactúanlos archivos **HTML** y **JS** para crear aplicaciones web dinámicas y funcionales.