PRIMERA PARCIAL

Fecha: 29/08/24

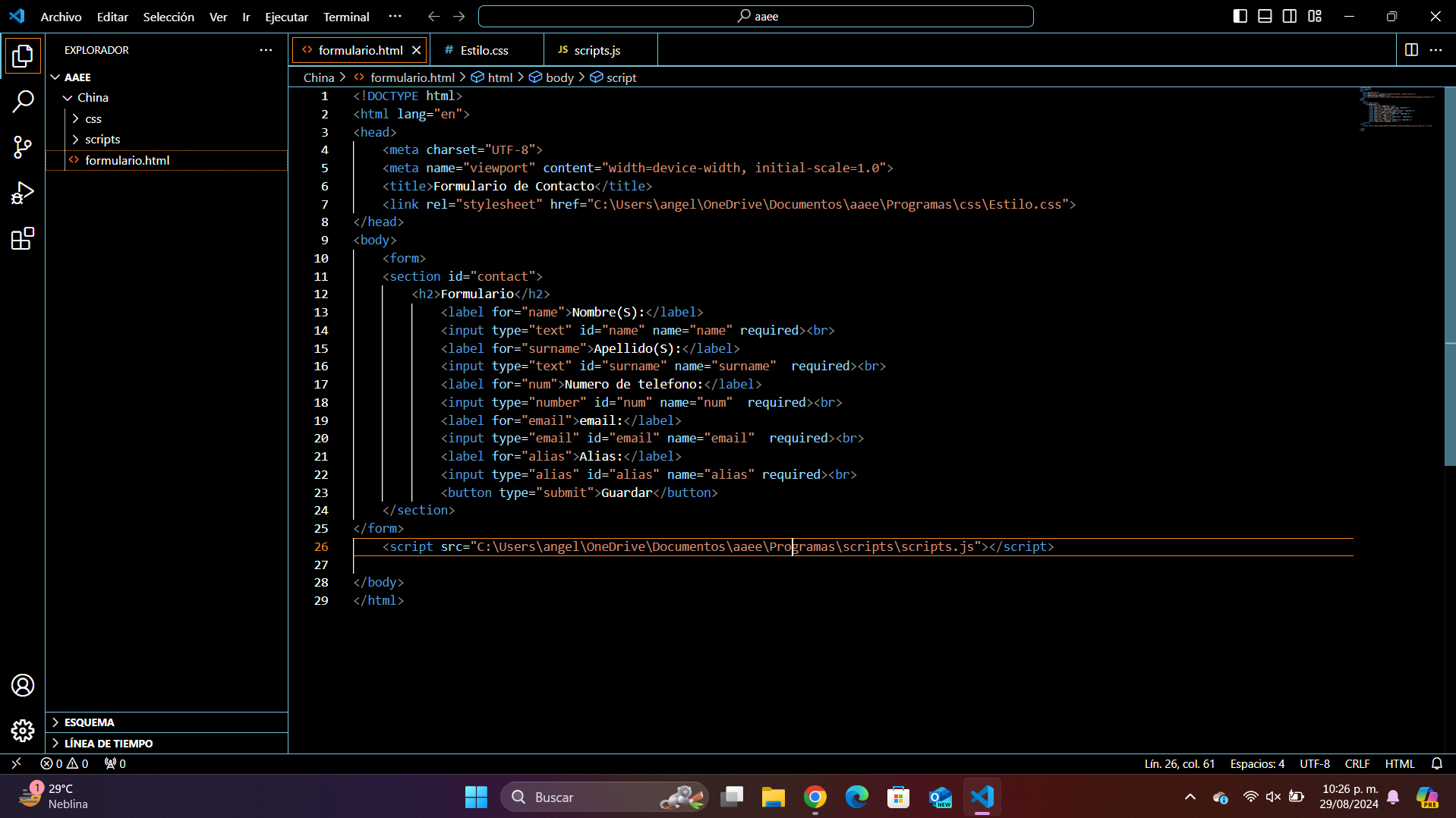
ALUMNO: Angel Moises Guzman Solis

MODULO: DESARROLLA APLICACIONES CON CONEXION A BASE DE DATOS

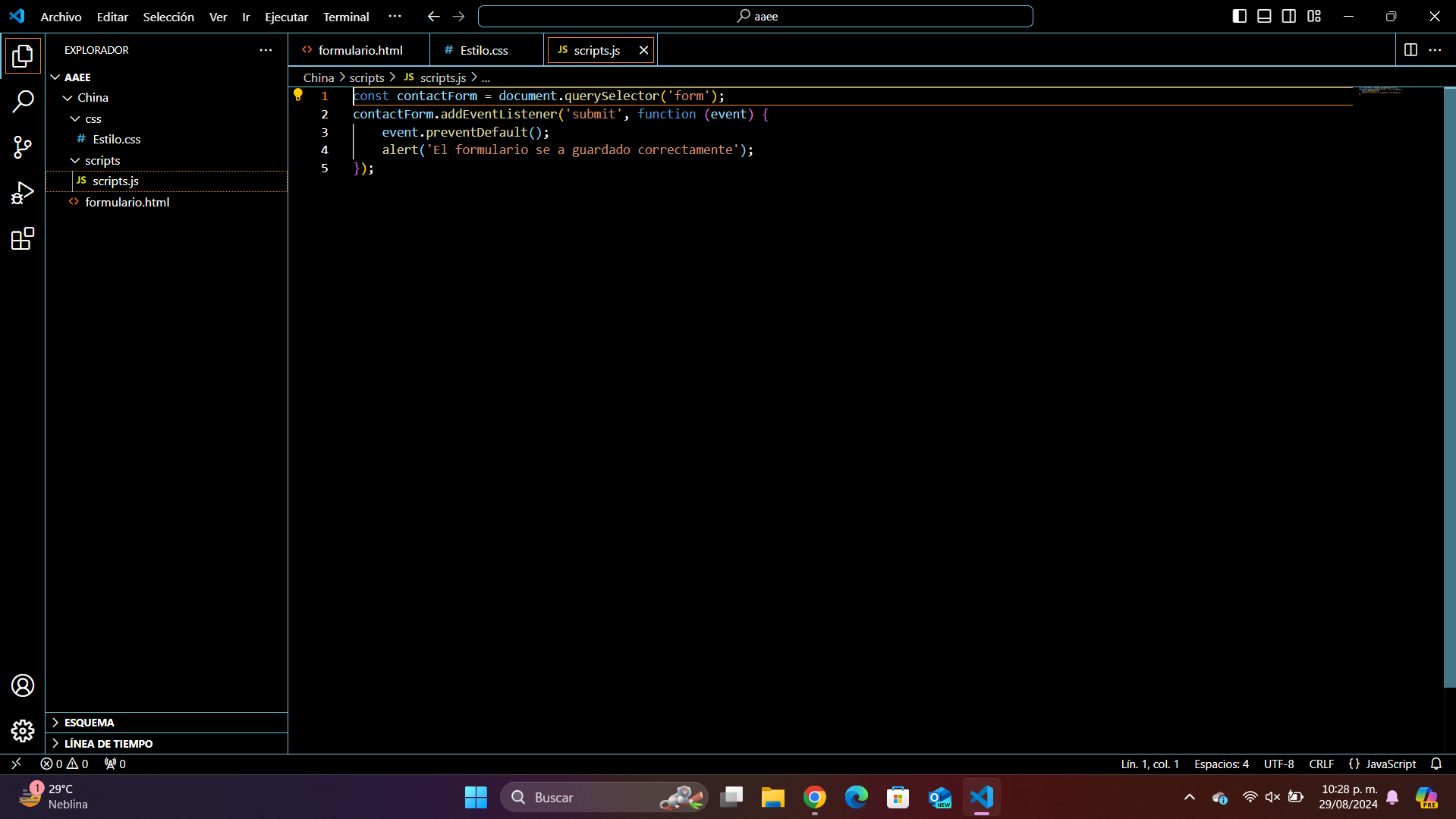
DOCENTE: GABRIEL IGNACIO CHINA CORTES

ESCUELA: CENTRO DE ESTUDIOS TECNOLOGICOS INDUSTRIAL Y DE SERVICIOS NO.084

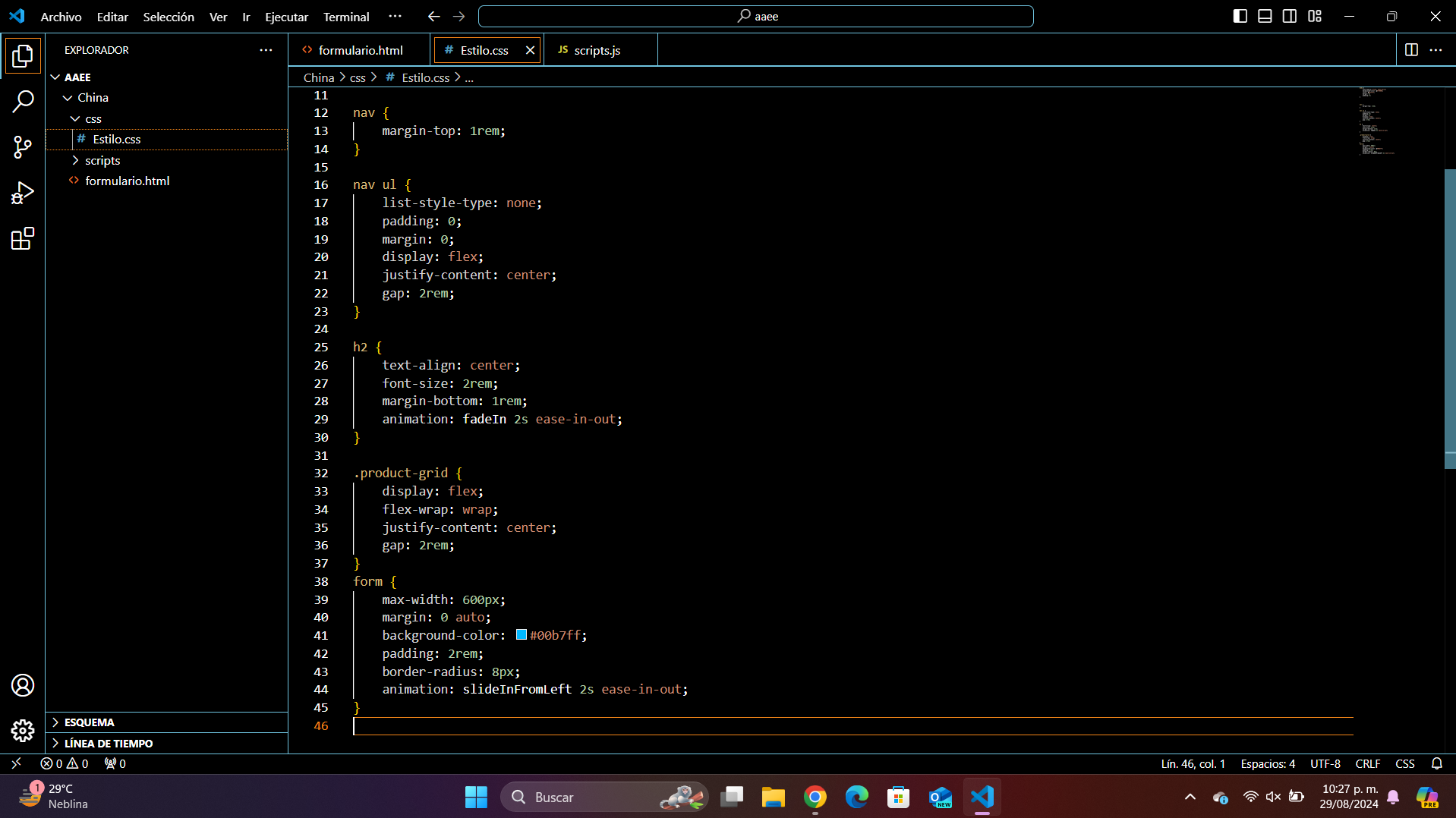
CAPTURAS DEL CODIGO HTML



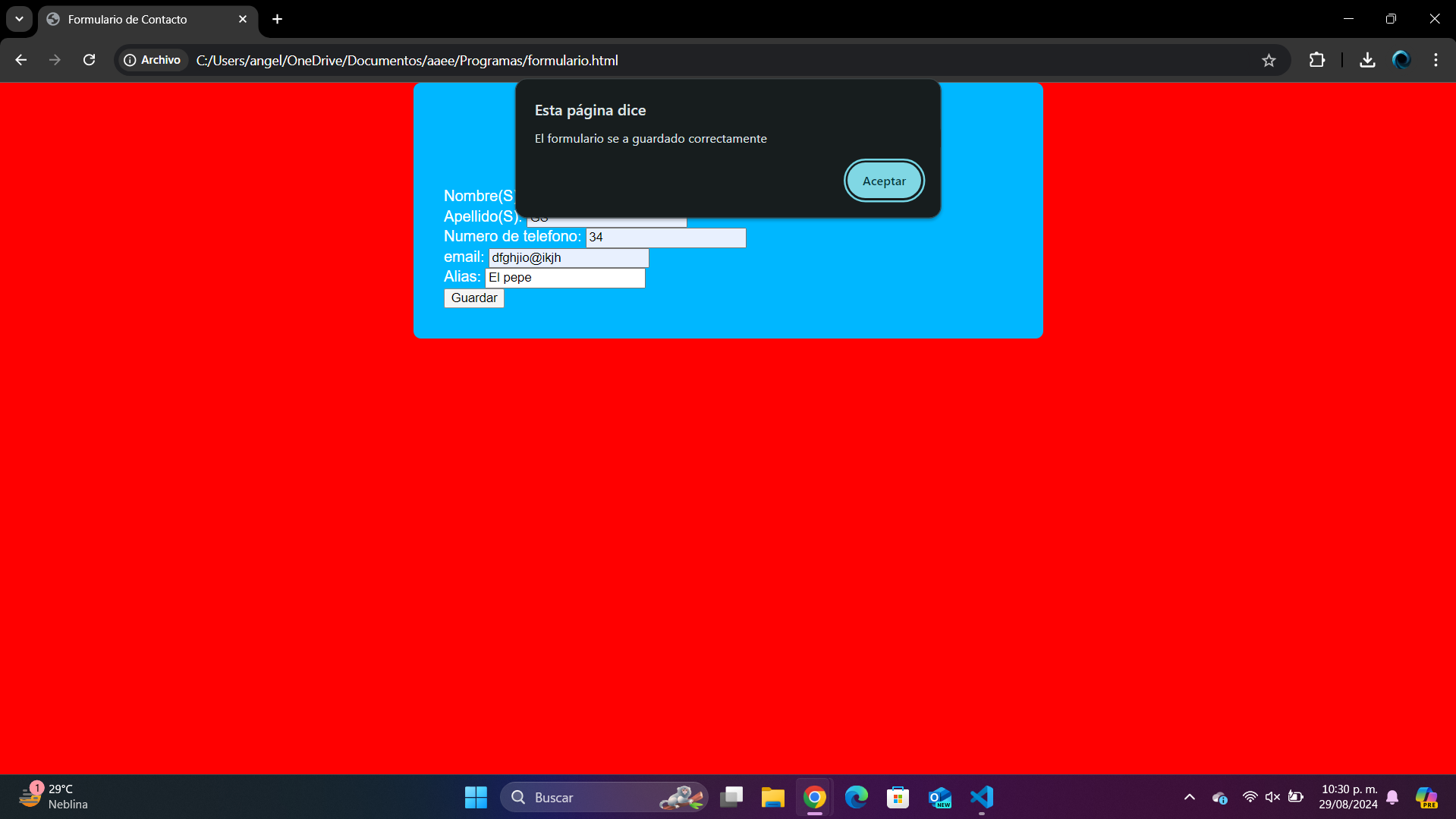
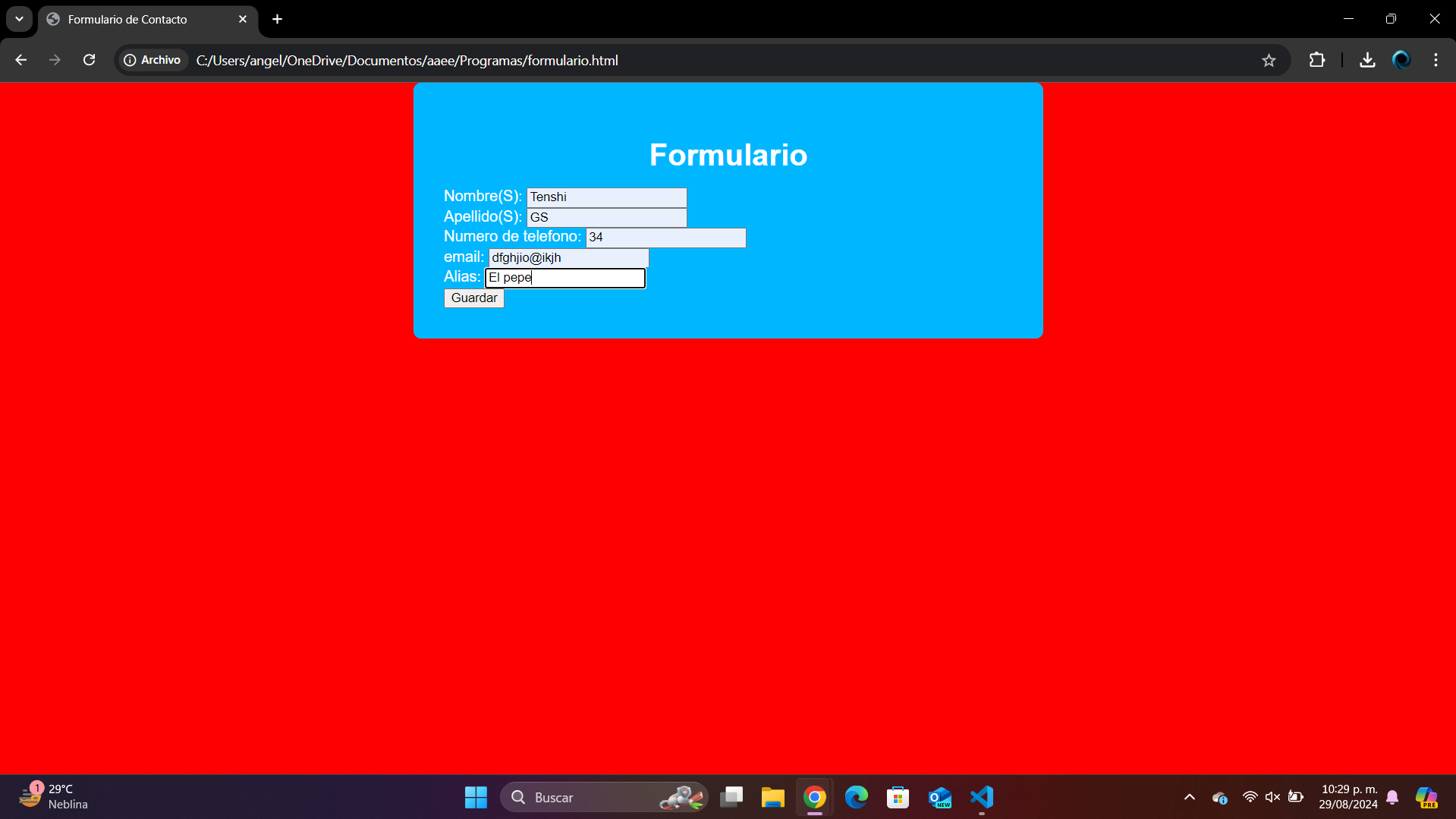
CAPTURAS DEL SCRIPT



CAPTURA DEL ESTILO



CAPTURA DE FUNCIONAMIENTO



5 BUENAS PRACTICAS ALA HORA DE PROGRAMAR

**1. Separación de Concerns (Separación de Responsabilidades)**

* HTML: estructura y contenido
* CSS: estilos y diseño
* JS: lógica y interacción
* PHP: lógica de negocio y backend

Cada tecnología tiene su propio propósito y debemos mantenerlas separadas para evitar confusiones y facilitar el mantenimiento del código.

**2. Code Organization and Structure (Organización y Estructura del Código)**

* Utiliza carpetas y archivos organizados para cada tecnología (e.g., **css**, **js**, **php**, etc.)
* Utiliza nombres de archivos y variables descriptivos y coherentes
* Utiliza comentarios y documentación para explicar el propósito del código

Un código organizado y estructurado es más fácil de leer, mantener y escalar.

**3. Code Reusability and Modularity (Reutilización y Modularidad del Código)**

* Utiliza funciones y módulos para evitar código duplicado
* Utiliza patrones de diseño para resolver problemas comunes
* Utiliza bibliotecas y frameworks para aprovechar el trabajo de otros desarrolladores

Un código reutilizable y modular es más eficiente y escalable.

**4. Error Handling and Debugging (Manejo de Errores y Depuración)**

* Utiliza try-catch blocks para manejar errores en JS y PHP
* Utiliza herramientas de depuración como console.log () o var\_dump() para identificar problemas
* Utiliza herramientas de desarrollo como el inspector de elementos en el navegador para depurar HTML y CSS

Un código que maneja errores y se depura fácilmente es más confiable y estable.

**5. Security and Best Practices (Seguridad y Buenas Prácticas)**

* Utiliza HTTPS para proteger la comunicación entre el cliente y el servidor
* Utiliza validación y sanitización de datos para evitar ataques de inyección SQL o XSS
* Utiliza contraseñas seguras y autenticación adecuada para proteger la información del usuario

Un código seguro y que sigue buenas prácticas es más confiable y protege la información del usuario.

En cuanto al funcionamiento, es importante tener en cuenta que cada tecnología tiene su propio ciclo de vida y flujo de trabajo:

* HTML: se carga en el navegador y se renderiza en la pantalla
* CSS: se carga en el navegador y se aplica a los elementos HTML
* JS: se carga en el navegador y se ejecuta en el lado del cliente
* PHP: se ejecuta en el servidor y devuelve una respuesta al navegador