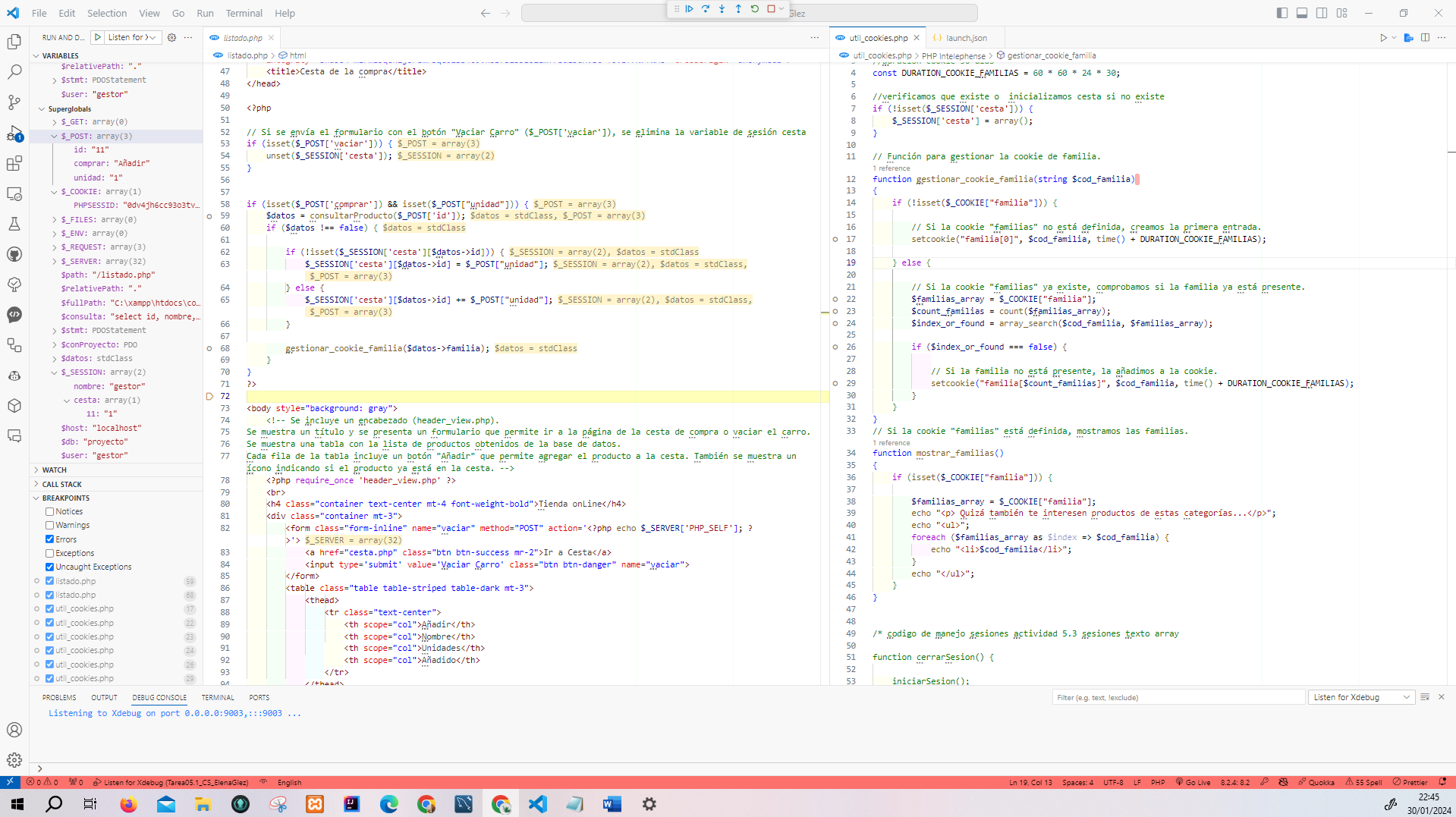
5- Asegúrate de que tienes bien configurado XDebug y que eres capaz de realizar depuración en tu código.

Sitúa puntos de ruptura en las instrucciones en las que añades una nueva cookie para registrar los códigos de familia y comprueba en tiempo de depuración si $\_COOKIE contiene el valor que acabas de añadir a la cookie, justo después de crearla. Incluye una captura del depurador en el Visual Studio Code donde se observe:

* la línea de código que se acaba de ejecutar (señálala expresamente o indica en un texto de qué línea se trata)
* el valor de la variable que contenga el código de familia que se acaba de añadir
* el contenido del array superglobal $\_COOKIE. Explica en la propia imagen por qué no está presente el código de familia que se acaba de añadir a la cookie



6. Sitúa puntos de ruptura en la instrucción session\_start(). Muestra en sendas capturas:

a)**valor de la directiva session.auto\_start**

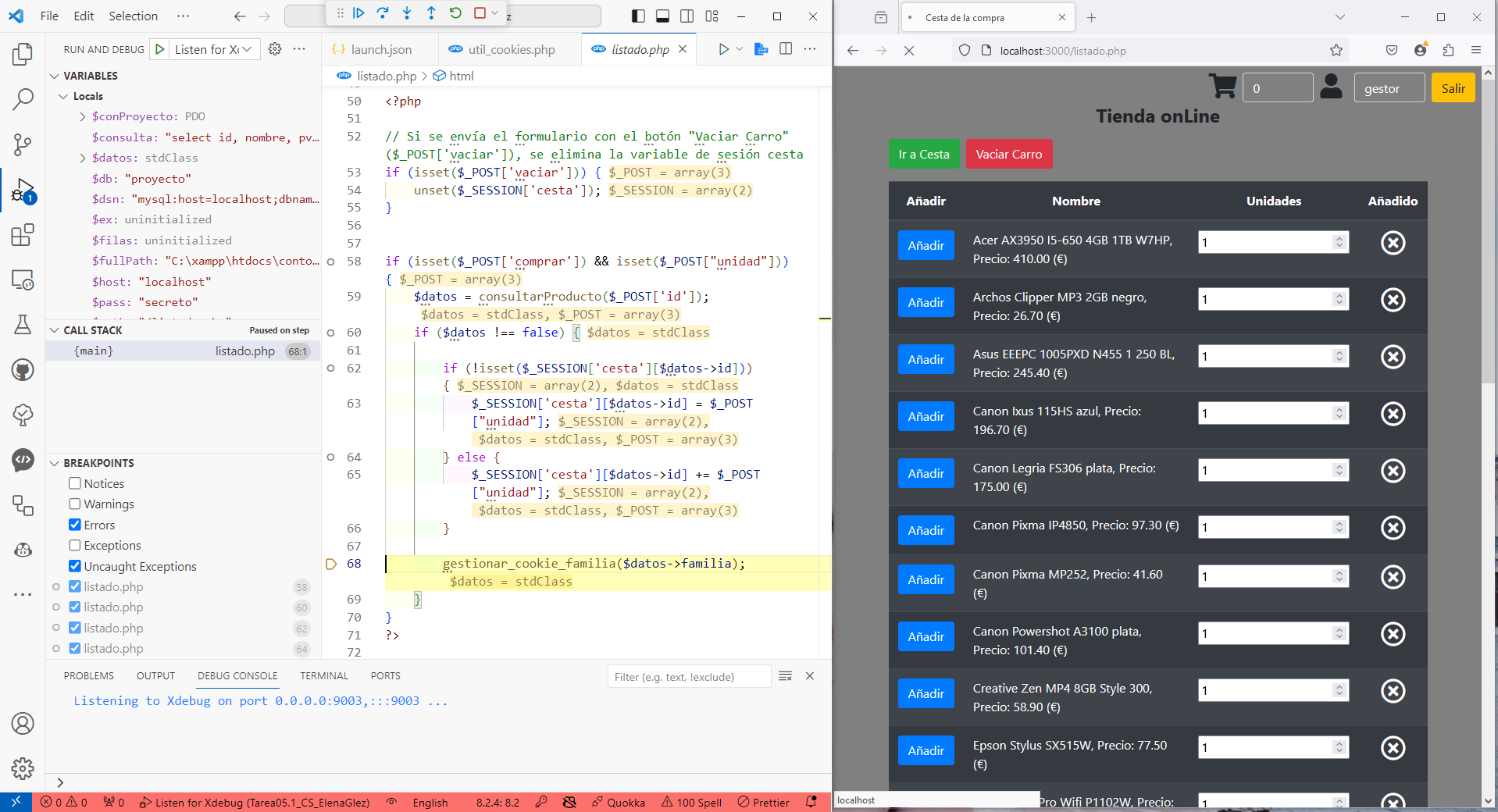
$autoStart = ini\_get("session.auto\_start"); su valor es 0 como se puede ver en la captura.

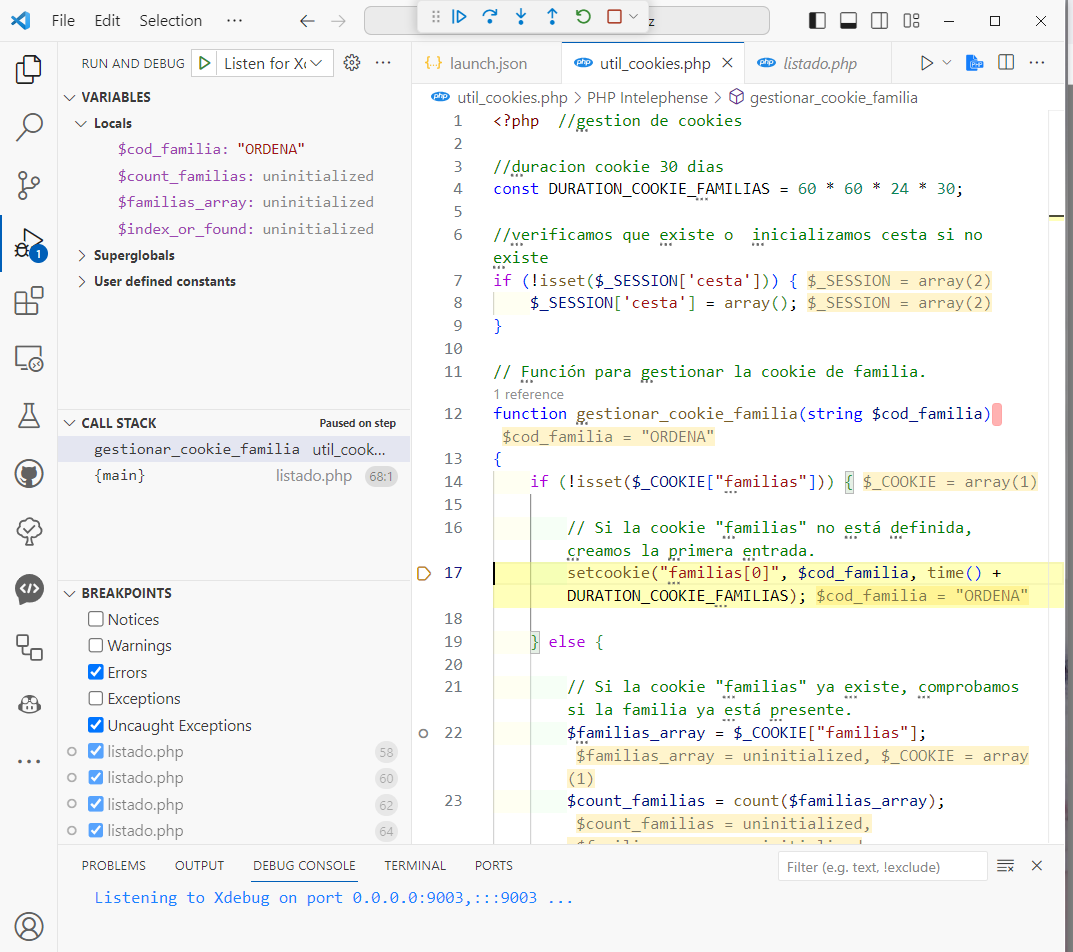
b)**antes de ejecutarse session\_start(), ¿existe $\_SESSION? ¿Por qué?**No, como podemos ver en las capturas $\_SESSION no existe ya que la directiva “session.auto\_start: bool” especifica si el módulo de sesión inicia una sesión automáticamente cuando arranque una petición. Por defecto es 0 (deshabilitado).

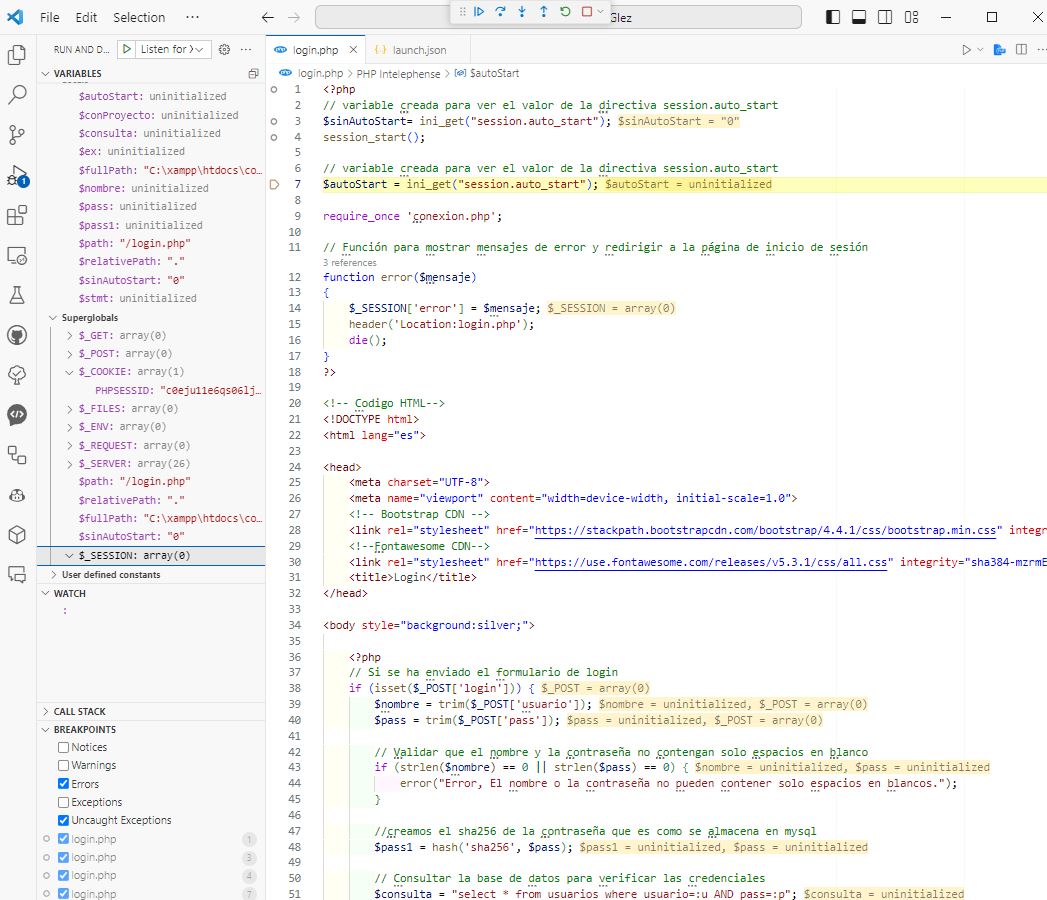
c)**justo después de ejecutarse session\_start() ¿existe $\_SESSION?¿Está vacía?¿Por qué?**

Despues de ejecutarse session\_start() existe la array superglobal $\_SESSION pero está vacia porque php aún no tiene los datos de retorno necesarios para devolverlos.

Despues de SESSION\_START() PHP llamará a los gestores de almacenamiento de sesiones open y read. Éstos serán un gestor de almacenamiento proporcionado por omisión o por extensiones de PHP (como SQLite o Memcached); o pueden ser un gestor personalizado como está definido en session\_set\_save\_handler(). La llamada de retorno read recuperará cualquier información s de sesión existente (almacenada en un formato serializado especial) y será deserializada y usada para rellenar automáticamente la variable superglobal $\_SESSION cuando la llamada de retorno read devuelva la información de sesión guardada a la gestión de sesiones de PHP.







**justo después de ejecutarse session\_start() ¿existe $\_SESSION?¿Está vacía? ¿Por qué?**

Después de ejecutarse session\_start() existe la array superglobal $\_SESSION pero está vacía porque php aún no tiene los datos de retorno necesarios para devolverlos.

Después de session\_start() PHP llamará a los gestores de almacenamiento de sesiones open y read. Éstos serán un gestor de almacenamiento proporcionado por omisión o por extensiones de PHP (como SQLite o Memcached); o pueden ser un gestor personalizado como está definido en session\_set\_save\_handler(). La llamada de retorno read recuperará cualquier información s de sesión existente (almacenada en un formato serializado especial) y será deserializada y usada para rellenar automáticamente la variable superglobal $\_SESSION cuando la llamada de retorno read devuelva la información de sesión guardada a la gestión de sesiones de PHP.