



Instituto Politécnico Nacional
Escuela Superior de Computo



Sistemas Distribuidos

Tarea 8

Prototipo de un sistema de comercio electrónico utilizando serverless

Nombre del alumno:

García Quiroz Gustavo Ivan

Grupo: 7CV4

Nombre del profesor: Guerrero Carlos Pineda

Fecha de entrega: 07/12/2025

ÍNDICE

1	Introducción	1
2	Objetivos	2
2.1	Objetivo general.....	2
2.2	Objetivos específicos	2
3	Nomenclatura y recursos creados en Azure	4
4	Preparación del entorno local	5
4.1	Resumen de software requerido.....	5
4.2	Instalación y configuración de MySQL en Windows 11	5
4.3	Creación de la base de datos “servicio_web” y tablas iniciales	13
4.4	Creación de tablas adicionales para el e-commerce	16
4.5	Creación de usuario local y privilegios	18
4.6	Instalación de Visual Studio Code	18
4.7	Instalación de .NET SDK	21
4.8	Instalación de Azure Functions Core Tools	22
4.9	Creación del proyecto Azure Functions en Visual Studio Code	23
4.10	Instalación de paquetes .NET requeridos	27
4.11	Configuración de variables locales y front-end	28
4.12	Pruebas locales iniciales del front-end y back-end	30
5	Desarrollo.....	34
6	Back-end: Funciones de Azure implementadas	34
6.1	Resumen de funciones y operaciones	34
6.2	alta_usuario (registro de usuarios).....	36
6.3	consulta_usuario (perfil del usuario)	37
6.4	modifica_usuario (actualización de perfil y foto)	38

6.5	borra_usuario (eliminación del perfil)	39
6.6	login y verifica_acceso (autenticación y seguridad)	40
6.7	alta_articulo (captura de stock)	42
6.8	consulta_articulos (búsqueda por palabra clave).....	43
6.9	compra_articulo (compra con transacción)	44
6.10	elimina_articulo_carrito_compra (devolver al stock y borrar del carrito)	45
6.11	elimina_carrito_compra (borrado total del carrito).....	46
6.12	modifica_carrito_compra (ajuste de cantidad +1/-1 con validaciones).....	47
7	Front-end: Aplicación web	49
7.1	Estructura de archivos y carga del front-end.....	49
7.2	Menú principal extendido	49
7.3	Pantalla “Captura de artículo”	50
7.4	Pantalla “Compra de artículos”	53
7.5	Pantalla “Artículos en el carrito”	54
7.6	Integración con WSClient.js	55
8	Despliegue en Azure	56
8.1	Creación de Azure Database for MySQL (t8mysql2022630278) e inicialización de esquema	56
8.2	Creación de Storage Account (t8af2022630278) y File Share (t8rca2022630278)	
	65	
8.3	Montaje en Function App (ruta /t8pm2022630278) y variable ROOT	71
8.4	Publicación de Azure Functions (t8ap2022630278).....	79
8.5	Pruebas de acceso a la aplicación: URL /api/Get?nombre=/prueba.html	81
9	Evidencias requeridas (capturas de pantalla completas con fecha y hora).....	83
9.1	Variables de entorno en Azure (Function App: Configuration).....	83
9.2	Pruebas en dispositivo móvil del front-end.....	84

9.2.1	Requerimiento 1: Pantalla “Captura de artículo”	86
9.2.2	Requerimiento 2: Pantalla “Compra de artículos” con búsqueda por palabra clave	88
9.2.3	Requerimiento 3: Agregar artículo al carrito desde “Compra de artículos”	90
9.2.4	Requerimiento 5: Pantalla “Artículos en el carrito” con total	92
9.2.5	Requerimiento 6: Modificación de cantidad en el carrito (+/-).....	93
9.2.6	Requerimiento 6 (extensión coherente) y 7: Eliminar artículo y eliminar carrito	97
9.2.7	Botón adicional “Comprar todo” (checkout)	99
9.2.8	Navegación entre pantallas y login/registro	101
9.3	Pruebas unitarias del despliegue del back-end con curl	104
9.4	Pruebas unitarias locales del back-end con curl (detalladas)	105
9.4.1	Prueba 0. Login y obtención de token (previo a las demás)	106
9.4.2	Prueba 0.1 Consulta de usuario para obtener id_usuario.....	107
9.4.3	Requerimiento 1 (back-end): alta_articulo	108
9.4.4	Requerimiento 2: consulta_articulos (búsqueda por palabra clave)	109
9.4.5	Requerimiento 3: compra_articulo	110
9.4.6	Requerimiento 4: elimina_articulo_carrito_compra	112
9.4.7	Requerimiento 5: elimina_carrito_compra	113
9.4.8	Requerimiento 6: modifica_carrito_compra (+1/-1).....	114
9.4.9	Función de apoyo: consulta_carrito	118
9.4.10	Checkout opcional: finaliza_compra (“Comprar todo”).....	119
10	Conclusiones	121
	Enlace del chat de la IA generativa	122
11	Referencias (Formato IEEE).....	123

1 Introducción

Este reporte documenta la Tarea 8 del curso de Sistemas Distribuidos: el desarrollo de un prototipo de sistema de comercio electrónico con arquitectura serverless utilizando Azure Functions (back-end) y una aplicación web HTML/JavaScript (front-end). El objetivo es implementar las funciones de negocio clave (alta y consulta de artículos, compra y gestión de carrito) integradas con una base de datos MySQL en PaaS, y desplegar el front-end mediante Azure Files montado en la Function App.

Durante el desarrollo se empleó inteligencia artificial de GitHub Copilot para acelerar la generación de código y la estructuración de las funciones, manteniendo buenas prácticas como el uso de transacciones en operaciones críticas (compra, eliminación y modificación del carrito), validaciones de acceso mediante tokens y manejo consistente de respuestas JSON. La IA apoyó en la producción de plantillas C#, en la organización del front-end y en la definición de pruebas, reduciendo el tiempo de implementación sin sacrificar calidad.

El prototipo se orienta a la ejecución en Azure (región Canada) con recursos nombrados según la nomenclatura exigida por la tarea, asegurando trazabilidad y cumplimiento. Se realizaron pruebas locales y móviles, y se dejó preparada la evidencia requerida para el reporte final.

NOTA: La plataforma de Moodle no permitió subir el reporte en buena calidad debido al límite de 2 MB por archivo. Sugiero revisar el siguiente documento de Google drive con el reporte en buena calidad:

<https://drive.google.com/file/d/1w9JvNSqul117kZ7yCsSpDczbWH0I-5Qa/view?usp=sharing>

2 Objetivos

2.1 Objetivo general

Desarrollar y desplegar un prototipo de sistema de comercio electrónico con arquitectura serverless en Azure (Functions + MySQL PaaS + Azure Files), que permita la captura, consulta y compra de artículos con gestión de carrito y verificación de acceso por token, cumpliendo la nomenclatura, región y lineamientos del curso; apoyándose en GitHub Copilot para acelerar el desarrollo manteniendo buenas prácticas.

2.2 Objetivos específicos

- Crear funciones para alta y consulta de artículos, compra, modificación y eliminación del carrito, con validaciones y respuestas JSON.
- Integrar MySQL PaaS (IP pública) y asegurar operaciones críticas mediante transacciones.
- Aplicar verificación de acceso con token por usuario en todas las funciones de negocio.
- Incorporar pantallas “Captura de artículo”, “Compra de artículos” y “Artículos en el carrito”.
- Mostrar resultados con foto, nombre, descripción, precio y controles de cantidad (+/-), compra y gestión del carrito.
- Consumir el back-end con WSClient.js (GET/POST/PUT/DELETE), usando id_usuario y token tras el login.
- Crear y nombrar recursos conforme a la boleta: t8vs2022630278, t8ap2022630278, t8mysql2022630278, t8af2022630278, t8rca2022630278, /t8pm2022630278.
- Montar Azure Files en la Function App y definir ROOT para servir el front-end con la función Get.

- Establecer variables de entorno (Server, UserID, Password, Database, ROOT) y un usuario remoto para MySQL.
- Ejecutar pruebas locales (curl) por cada función del back-end y documentar resultados.
- Realizar pruebas en dispositivo móvil del flujo completo del front-end y capturar evidencias conforme a lineamientos.
- Verificar disponibilidad y correcta lectura de archivos para la entrega en Moodle (PDF y texto/ZIP).

3 Nomenclatura y recursos creados en Azure

Para garantizar el cumplimiento de los requerimientos no funcionales y la trazabilidad por boleta, se aplicó la nomenclatura oficial basada en el número de boleta 2022630278. Todos los recursos se crearon en Azure for Students, región Canada, para homogeneidad operativa y disponibilidad en el entorno del curso.

Los recursos y nombres utilizados fueron:

- Proyecto de Visual Studio Code (carpeta local): t8vs2022630278
- Azure Functions App (back-end): t8ap2022630278
- Instancia MySQL en PaaS (IP pública): t8mysql2022630278
- Cuenta de almacenamiento (Azure Files): t8af2022630278
- Recurso compartido de archivos (File Share): t8rca2022630278
- Ruta de montaje del File Share en la Function App: /t8pm2022630278, asignada a la variable de entorno ROOT

La región seleccionada fue Canada. Esto se mantuvo de forma consistente al crear la Function App, la instancia de MySQL y la cuenta de almacenamiento, con el fin de minimizar latencias entre servicios y respetar el lineamiento de la práctica. Además, la variable de entorno ROOT apunta a la ruta montada del File Share, permitiendo que la función Get sirva los archivos del front-end (HTML, JS, CSS e imágenes) directamente desde Azure Files.

- Consideraciones de configuración:
 - Variables de entorno en la Function App: Server, UserID, Password, Database y ROOT.
 - MySQL PaaS configurado con usuario remoto (sin @localhost) y esquema “servicio_web”.
 - Montaje del File Share t8rca2022630278 en la ruta /t8pm2022630278 para servir el front-end.

4 Preparación del entorno local

Se realizó la instalación completa del entorno en Windows 11 para ejecutar el prototipo de manera local antes del despliegue a Azure. A continuación se detalla la instalación de MySQL, Visual Studio Code, .NET SDK y Azure Functions Core Tools, así como la preparación del proyecto y la configuración del front-end y variables.

4.1 Resumen de software requerido

Se identificaron las herramientas necesarias y sus versiones sugeridas para garantizar compatibilidad con C#, Azure Functions y MySQL.

Herramienta	Versión sugerida	URL oficial
MySQL Server	8.0.x	https://dev.mysql.com/downloads/
MySQL Workbench (opcional)	8.0.x	https://dev.mysql.com/downloads/workbench/
Visual Studio Code	Última estable	https://code.visualstudio.com/
.NET SDK (C#)	8.0.x (o 7.0.x)	https://dotnet.microsoft.com/download
Azure Functions Core Tools	v4 (para Functions v4)	https://aka.ms/azfunc-install
Node.js (para Azure Functions Tools)	18.x o 20.x (LTS)	https://nodejs.org/en

Tabla 1 Lista de herramientas y versiones recomendadas para el entorno local.

4.2 Instalación y configuración de MySQL en Windows 11

Se instaló MySQL Server utilizando el MySQL Installer. Durante la instalación se seleccionó el “MySQL Server” y opcionalmente “MySQL Workbench” para administrar la base de datos gráficamente. Se configuró el servicio para iniciar automáticamente y se estableció una contraseña segura para el usuario root. Se accedió a MySQL ya sea desde Workbench o desde la línea de comandos.

- Se descargó “MySQL Installer” y se ejecutó con privilegios de administrador.

The screenshot shows a web browser window with multiple tabs open. The active tab displays the MySQL Community Downloads page. On the left, there's a sidebar with links to MySQL Yum Repository, MySQL APT Repository, MySQL SUSE Repository, MySQL Community Server, MySQL NDB Cluster, MySQL Router, MySQL Shell, MySQL Operator, MySQL NDB Operator, MySQL Workbench, and MySQL Installer for Windows. To the right, there's a promotional box for 'MySQL Enterprise Edition for Developers' with a subtext 'Free for learning, developing, and prototyping.' and a 'Download Now' button. The bottom of the page includes copyright information for Oracle and links for Privacy, Terms of Use, Trademark Policy, and Cookie Preferences.

This screenshot shows the same MySQL Community Downloads page, but the main content area is focused on the 'MySQL Installer'. It features a 'General Availability (GA) Releases' tab selected, showing the 'MySQL Installer 8.0.44' section. Below this, there's a note stating: 'Note: MySQL 8.0 is the final series with MySQL Installer. As of MySQL 8.1, use a MySQL product's MSI or Zip archive for installation. MySQL Server 8.1 and higher also bundle MySQL Configurator, a tool that helps configure MySQL Server.' There are dropdown menus for 'Select Version' (set to 8.0.44) and 'Select Operating System' (set to Microsoft Windows). Two download options are listed: 'Windows (x86, 32-bit), MSI Installer' (8.0.44, 2.1M, MD5: f48ab9b8c2db55ee39dd5f534d4581676) and 'Windows (x86, 32-bit), MSI Installer' (8.0.44, 558.3M, MD5: 338dce4ac543dfc280664c857d265e3). A note at the bottom encourages users to verify package integrity using MD5 checksums and GnuPG signatures. The browser interface at the top shows other tabs like 'SISDIS-7CV4: Tarea 8. Prototipo de...', 'Índice para el reporte de la...', 'Tarea 7 - GitHub Copilot', 'YouTube', and 'MySQL - MySQL Community Down...'. The bottom of the screen shows a Windows taskbar with icons for File Explorer, Task View, Start, and others.

The screenshot shows a Microsoft Edge browser window with the following details:

- Address Bar:** https://dev.mysql.com/downloads/file/?id=546163
- Title Bar:** MySQL Community Downloads
- Content Area:**
 - Login/Sign Up Buttons:** "Login » using my Oracle Web account" and "Sign Up » for an Oracle Web account".
 - Note:** MySQL.com is using Oracle SSO for authentication.
 - Text:** "No thanks, just start my download."
 - Footer:** ORACLE © 2025 Oracle, Privacy / Do Not Sell My Info, Terms of Use, Trademark Policy, Cookie Preferences.
- Downloads Panel (Visible on the right):**
 - File: mysql-installer-web-community-8.0.44.0.msi
 - Folder: front-end.zip
 - Folder: back-end.zip
- Taskbar:** Shows the Windows Start button, Search bar, and pinned icons for File Explorer, Edge, and others. The date and time are 04:57 p.m. 05/12/2025.

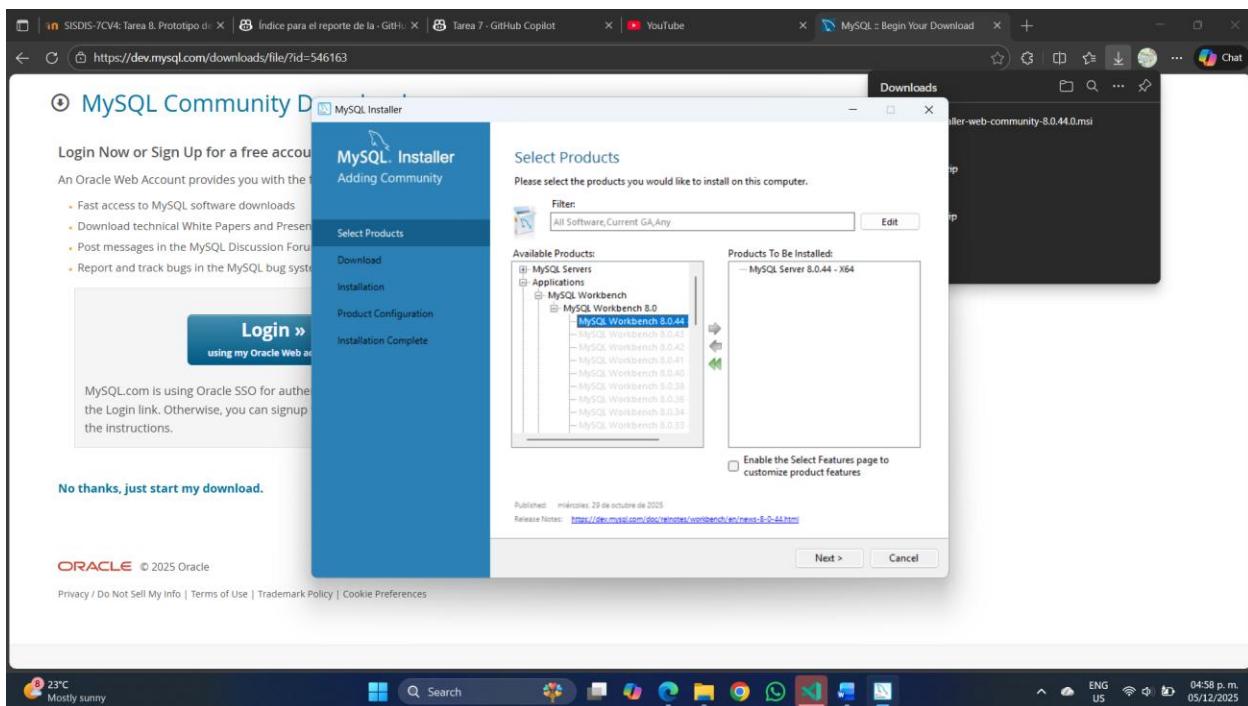
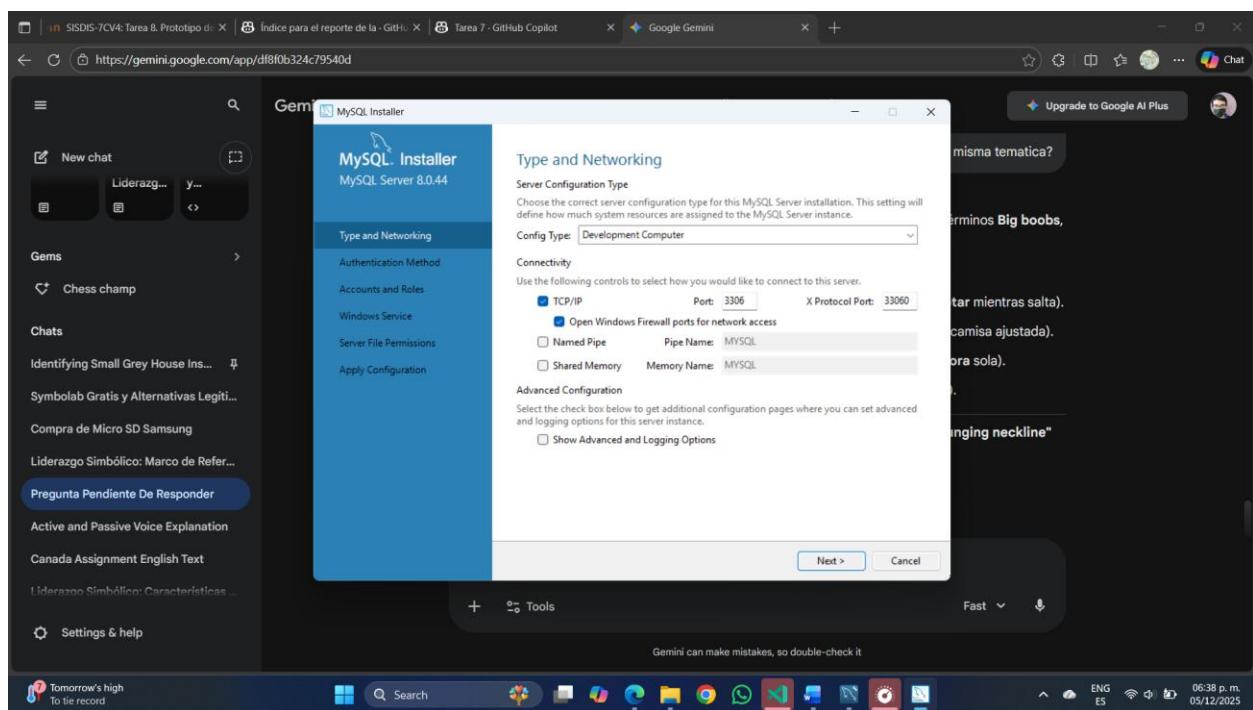
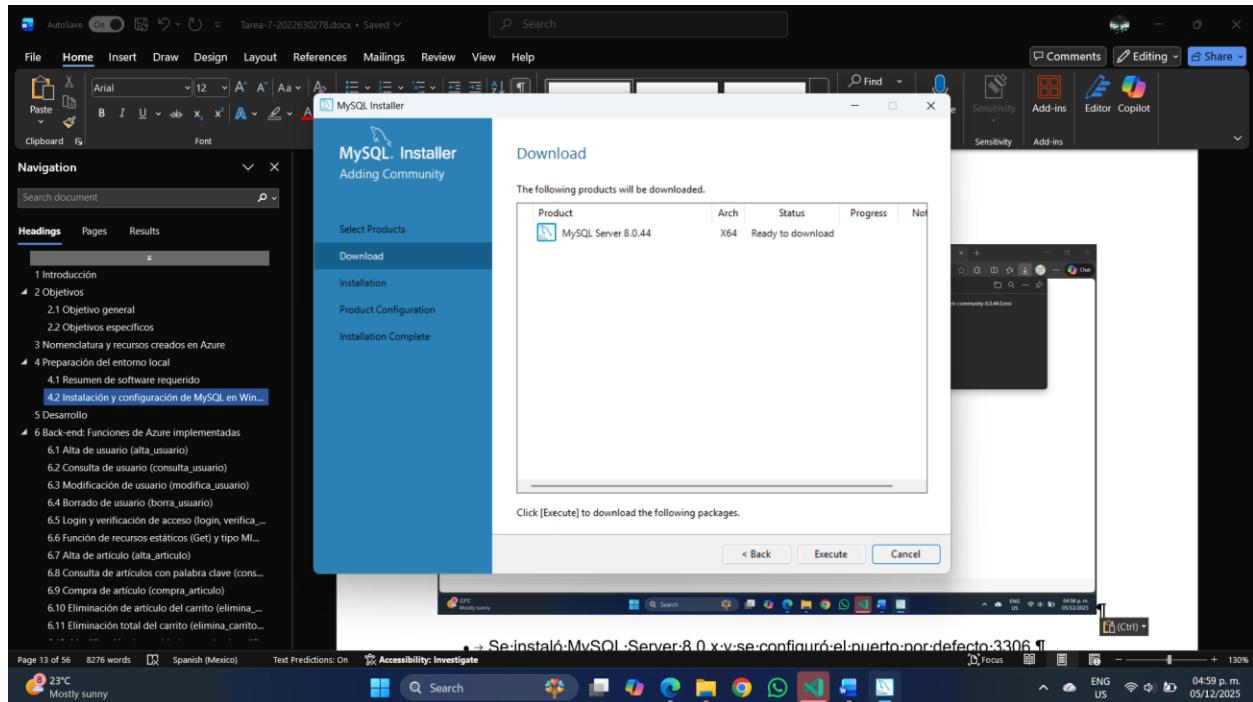
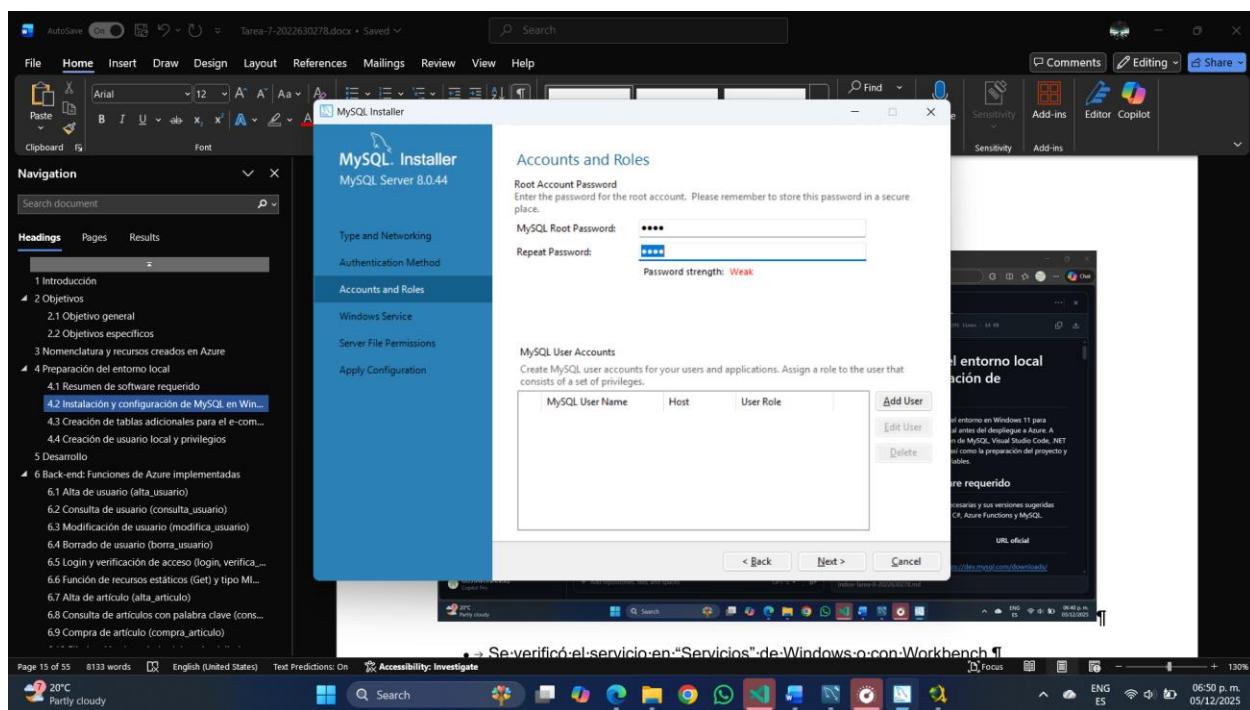
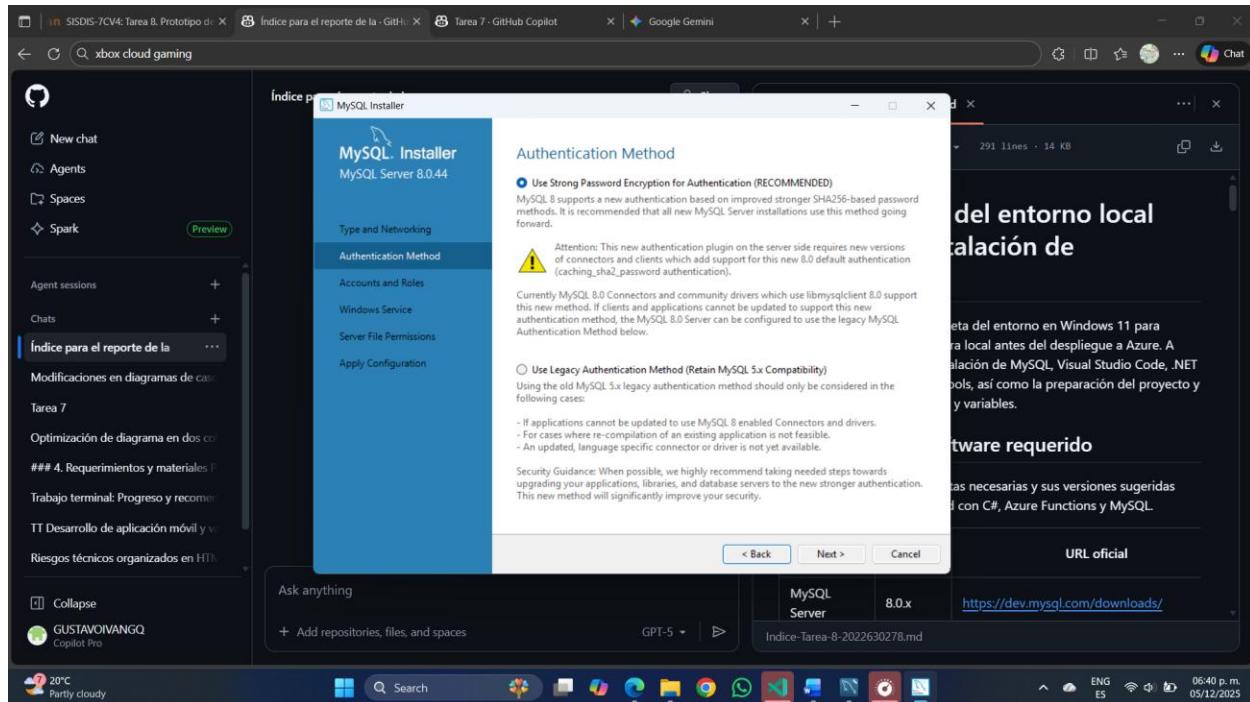
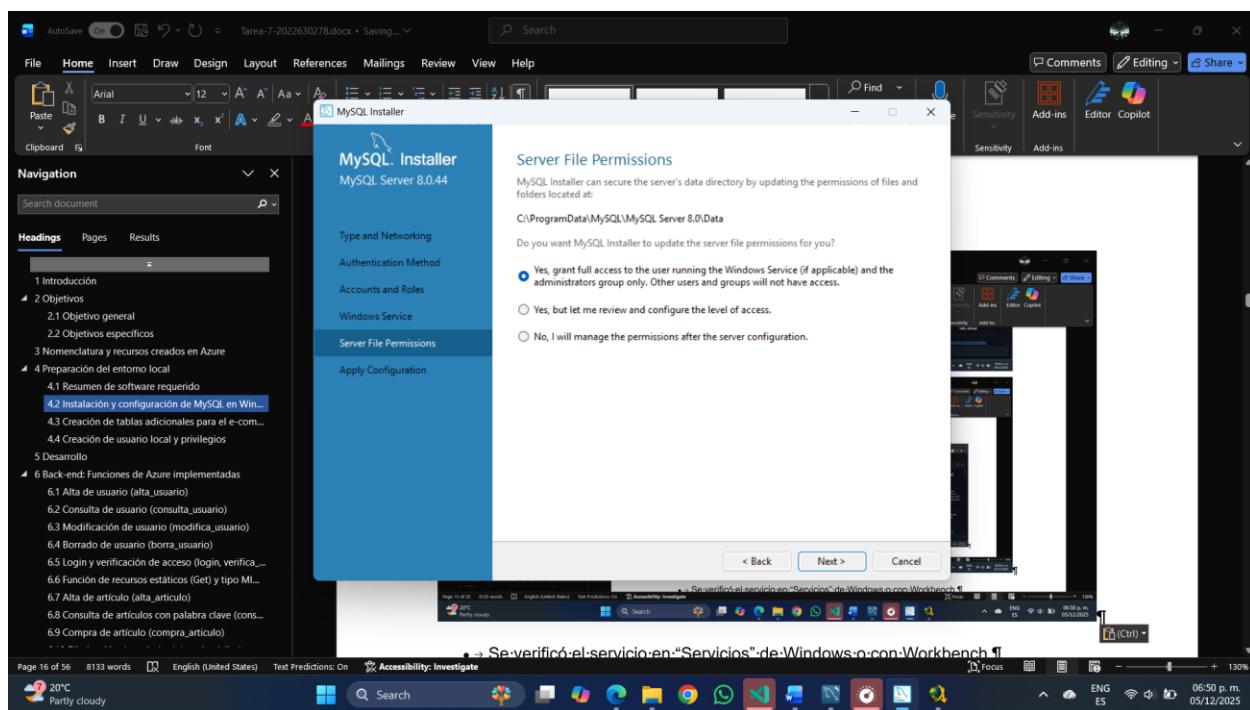
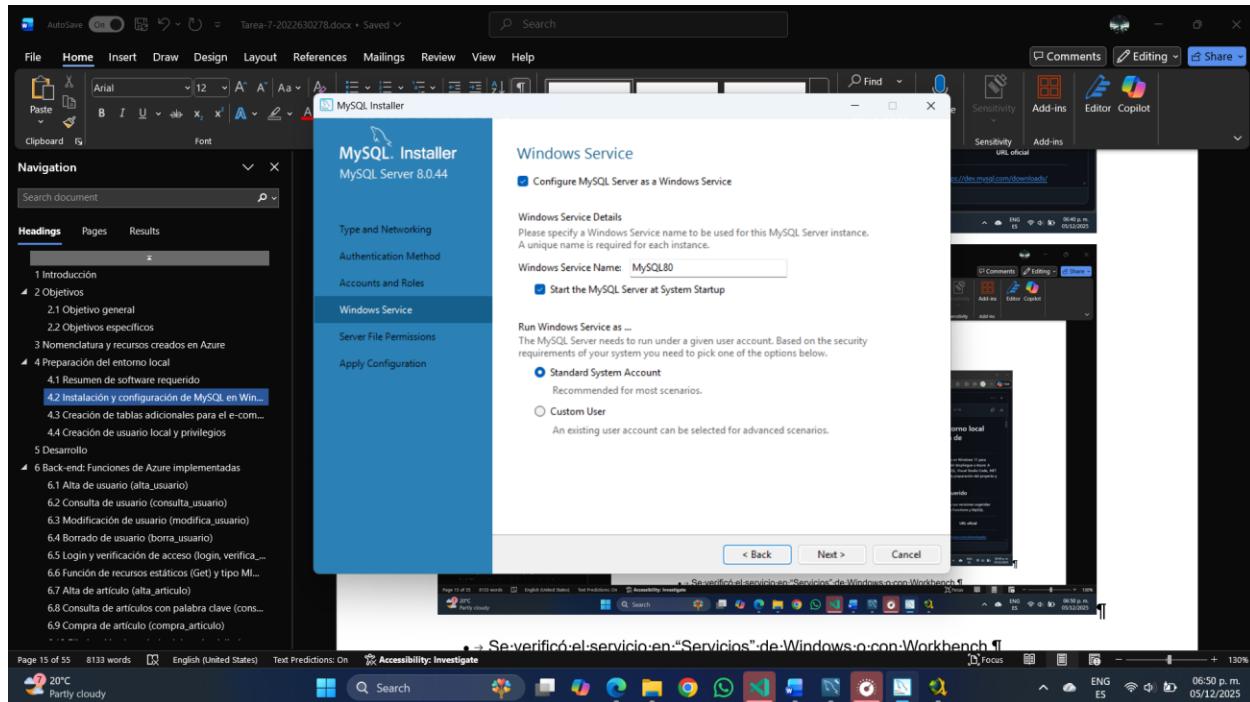


Figura 1.1 Instalación de MySQL Server.

- Se instaló MySQL Server 8.0.x y se configuró el puerto por defecto 3306.







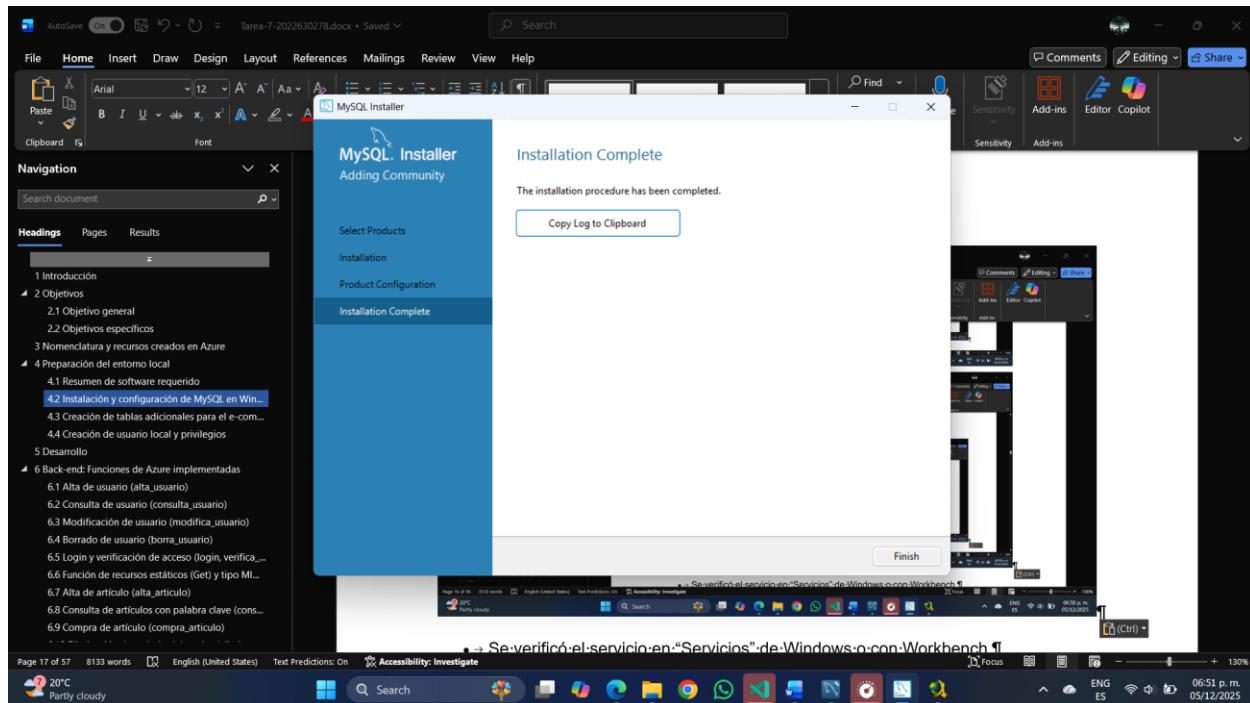
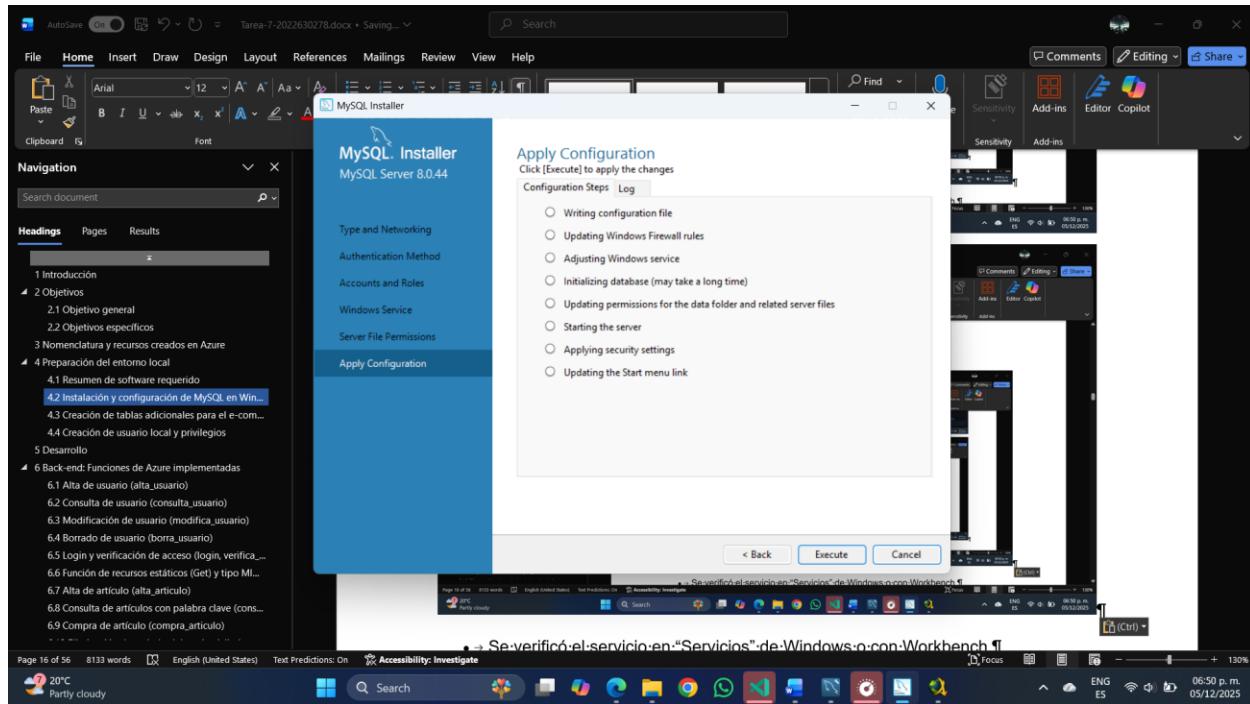


Figura 1.2 Instalación de MySQL Server.

- Se verificó el servicio en “Servicios” de Windows o con Workbench.

Parámetro	Valor (ejemplo local)

Host	localhost
Puerto	3306
Usuario admin	root
Autenticación	Contraseña definida en la instalación
Servicio	Iniciar automáticamente

Tabla 2 Parámetros

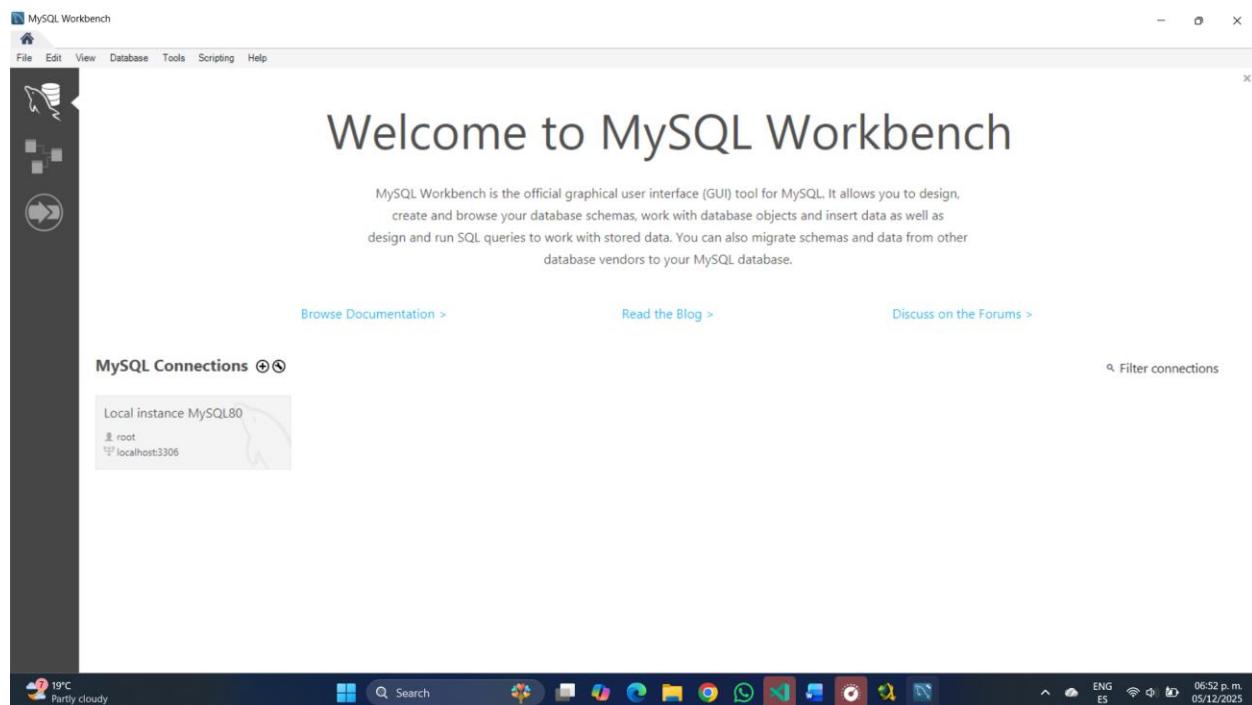


Figura 1. Instalación de MySQL Server y verificación del servicio activo en Windows 11.

4.3 Creación de la base de datos “servicio_web” y tablas iniciales

Se accedió a MySQL (Workbench o CLI) y se ejecutaron los scripts proporcionados para crear la base “servicio_web” y las tablas usuarios y fotos_usuarios con sus restricciones e índice único. Se validó que las columnas y llaves coinciden con el enunciado.

Tabla	Columnas principales
usuarios	id_usuario (PK, AI), email (UNIQUE), password, token, nombre, apellidos, fecha_nacimiento, teléfono, genero
fotos_usuarios	id_foto (PK, AI), foto (LONGBLOB), id_usuario (FK a usuarios.id_usuario)

Tabla 3 Tablas iniciales de la base de datos

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: MANAGEMENT INSTANCE PERFORMANCE

Query 1

```
1 * create database servicio_web;
2 use servicio_web;
3
4 * create table usuarios
5 (
6     id_usuario integer auto_increment primary key,
7     password varchar(64) not null,
8     token varchar(20),
9     email varchar(100) not null,
10    nombre varchar(100) not null,
11    apellido_paterno varchar(100) not null,
12    apellido_materno varchar(100),
13    fecha_nacimiento datetime not null,
14    telefono bigint,
15    genero char(1)
16 );
17
18 * create table fotos_usuarios
```

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Context Help Snippets

Administration Schemas Information

No object selected

Object Info Session

16°C Mostly clear

08:42 p.m. 05/12/2025

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: MANAGEMENT INSTANCE PERFORMANCE

Query 1

```
4 * create table usuarios
5 (
6     id_usuario integer auto_increment primary key,
7     password varchar(64) not null,
8     token varchar(20),
9     email varchar(100) not null,
10    nombre varchar(100) not null,
11    apellido_paterno varchar(100) not null,
12    apellido_materno varchar(100),
13    fecha_nacimiento datetime not null,
14    telefono bigint,
15    genero char(1)
16 );
17
18 * create table fotos_usuarios
19 (
20     id_foto integer auto_increment primary key,
21     foto longblob,
```

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Context Help Snippets

Administration Schemas Information

No object selected

Object Info Session

16°C Mostly clear

08:42 p.m. 05/12/2025

The screenshot shows the MySQL Workbench interface with a query editor containing the following SQL code:

```

8 token varchar(20),
9 email varchar(100) not null,
10 nombre varchar(100) not null,
11 apellido_paterno varchar(100) not null,
12 apellido_materno varchar(100),
13 fecha_nacimiento datetime not null,
14 telefono bigint,
15 genero char(1),
16 );
17
18 * create table fotos_usuarios
19 (
20     id_foto integer auto_increment primary key,
21     foto longblob,
22     id_usuario integer not null
23 );
24 * alter table fotos_usuarios add foreign key (id_usuario) references usuarios(id_usuario);
25 * create unique index usuarios_1 on usuarios(email);

```

The output pane shows the results of the executed statements:

Action	Time	Message	Duration / Fetch
1 create database servicio_web	20:43:10	1 row(s) affected	0.000 sec
2 use servicio_web		0 row(s) affected	0.016 sec
3 create table usuarios (id_usuario integer auto_increment primary key, password varchar(64) not null, token...		0 row(s) affected	0.016 sec
4 create table fotos_usuarios (id_foto integer auto_increment primary key, foto longblob, id_usuario integer n...		0 row(s) affected	0.016 sec
5 alter table fotos_usuarios add foreign key (id_usuario) references usuarios(id_usuario)		0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.032 sec
6 create unique index usuarios_1 on usuarios(email)		0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.016 sec

Figura 2. Ejecución del script SQL de creación de base y tablas iniciales en MySQL Workbench.

4.4 Creación de tablas adicionales para el e-commerce

Se extendió el esquema para soportar artículos, fotos de artículos y el carrito de compra. Se creó el índice único en la tabla carrito_compra sobre (id_usuario, id_articulo) para impedir duplicidad de artículos en un mismo carrito.

Tabla	Columnas principales
stock	id_articulo (PK, AI), nombre, descripcion, precio (DECIMAL), cantidad (INT)
fotos_articulos	id_foto (PK, AI), foto (LONGBLOB), id_articulo (FK a stock.id_articulo)
carrito_compra	id_usuario, id_articulo, cantidad, UNIQUE(id_usuario, id_articulo)

MySQL Workbench

Local instance MySQL80

servicio_web**

```

29 • create table stock
30   (
31     id_articulo integer auto_increment primary key,
32     nombre varchar(200) not null,
33     descripcion text,
34     precio decimal(10,2) not null,
35     cantidad integer not null
36   );
37
38 • create table fotos_articulos
39   (
40     id_foto integer auto_increment primary key,
41     foto longblob,
42     id_articulo integer not null,
43     foreign key (id_articulo) references stock(id_articulo)
44   );
45
46 • create table carrito_compra

```

Administration Schemas

Information

No object selected

Action Output

#	Time	Action	Message	Duration / Fetch
1	20:43:10	create database servicio_web	1 row(s) affected	0.000 sec
2	20:43:10	use servicio_web	0 row(s) affected	0.016 sec
3	20:43:10	create table usuarios (id_usuario integer auto_increment primary key, password varchar(54) not null, token...)	0 row(s) affected	0.016 sec
4	20:43:10	create table fotos_usuarios (id_foto integer auto_increment primary key, foto longblob, id_usuario integer n...)	0 row(s) affected	0.016 sec
5	20:43:10	alter table fotos_usuarios add foreign key (id_usuario) references usuarios(id_usuario)	0 rows affected Records: 0 Duplicates: 0 Warnings: 0	0.032 sec
6	20:43:10	create unique index usuarios_1 on usuarios(email)	0 rows affected Records: 0 Duplicates: 0 Warnings: 0	0.016 sec

Object Info Session

16°C Mostly clear

08:44 p.m. 05/12/2025

MySQL Workbench

Local instance MySQL80

servicio_web**

```

35
36   cantidad integer not null
37 );
38 • create table fotos_articulos
39   (
40     id_foto integer auto_increment primary key,
41     foto longblob,
42     id_articulo integer not null,
43     foreign key (id_articulo) references stock(id_articulo)
44   );
45
46 • create table carrito_compra
47   (
48     id_usuario integer not null,
49     id_articulo integer not null,
50     cantidad integer not null,
51     unique key carrito_unico (id_usuario, id_articulo)
52   );

```

Administration Schemas

Information

No object selected

Action Output

#	Time	Action	Message	Duration / Fetch
1	20:43:10	use servicio_web	0 row(s) affected	0.016 sec
2	20:43:10	create table usuarios (id_usuario integer auto_increment primary key, password varchar(54) not null, token...)	0 row(s) affected	0.016 sec
3	20:43:10	create table fotos_usuarios (id_foto integer auto_increment primary key, foto longblob, id_usuario integer n...)	0 row(s) affected	0.016 sec
4	20:43:10	alter table fotos_usuarios add foreign key (id_usuario) references usuarios(id_usuario)	0 rows affected Records: 0 Duplicates: 0 Warnings: 0	0.032 sec
5	20:43:10	create unique index usuarios_1 on usuarios(email)	0 rows affected Records: 0 Duplicates: 0 Warnings: 0	0.016 sec
6	20:44:53	create table carrito_compra (id_usuario integer not null, id_articulo integer not null, cantidad integer not ...)	0 row(s) affected	0.015 sec

Object Info Session

16°C Mostly clear

08:45 p.m. 05/12/2025

Figura 3. Creación de tablas stock, fotos_articulos y carrito_compra con sus llaves y restricciones.

4.5 Creación de usuario local y privilegios

Se creó el usuario local para desarrollo y se le otorgaron privilegios sobre la base “servicio_web”. Se accedió con este usuario para validar las operaciones del back-end.

```
create user x@localhost identified by ";
```

```
grant all on servicio_web.* to x@localhost;
```

Usuario	Host	Privilegios
x@localhost	localhost	ALL en servicio_web.*

Tabla 4 Usuarios

The screenshot shows the MySQL Workbench interface. In the central SQL editor window, the following SQL code is visible:

```
39  (
40     id_foto integer auto_increment primary key,
41     foto longblob,
42     id_articulo integer not null,
43     foreign key (id_articulo) references stock(id_articulo)
44 );
45
46 *  create table carrito_compra
47  (
48     id_usuario integer not null,
49     id_articulo integer not null,
50     cantidad integer not null,
51     unique key carrito_unico (id_usuario, id_articulo)
52 );
53
54 *  create user x@localhost identified by '';
55 *  grant all on servicio_web.* to x@localhost;
```

The lines starting with “*” indicate the creation of the user and granting of privileges. Below the SQL editor, the “Output” pane shows the execution log with the following entries:

Action	Time	Action	Message	Duration / Fetch
6 20:43:10	create unique index usuarios_1 on usuarios(email)		0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.016 sec
7 20:44:53	create table carrito_compra (id_usuario integer not null, id_articulo integer not null, cantidad integer not ...		0 row(s) affected	0.015 sec
8 20:45:59	grant all on servicio_web.* to x@localhost		Error Code: 1410. You are not allowed to create a user with GRANT	0.015 sec
9 20:46:28	grant all on servicio_web.* to x@localhost		Error Code: 1410. You are not allowed to create a user with GRANT	0.000 sec
10 20:47:18	create user x@localhost identified by ''		0 row(s) affected	0.000 sec
11 20:47:22	grant all on servicio_web.* to x@localhost		0 row(s) affected	0.031 sec

Figura 4. Confirmación de usuario local creado y concesión de privilegios en MySQL.

4.6 Instalación de Visual Studio Code

Se instaló Visual Studio Code desde el sitio oficial y se accedió al editor para instalar las extensiones necesarias: C# (Microsoft), Azure Functions y Azure Account. Se validó el inicio de sesión en Azure en caso de publicar posteriormente.

- Se descargó VS Code y se instaló con opciones por defecto.

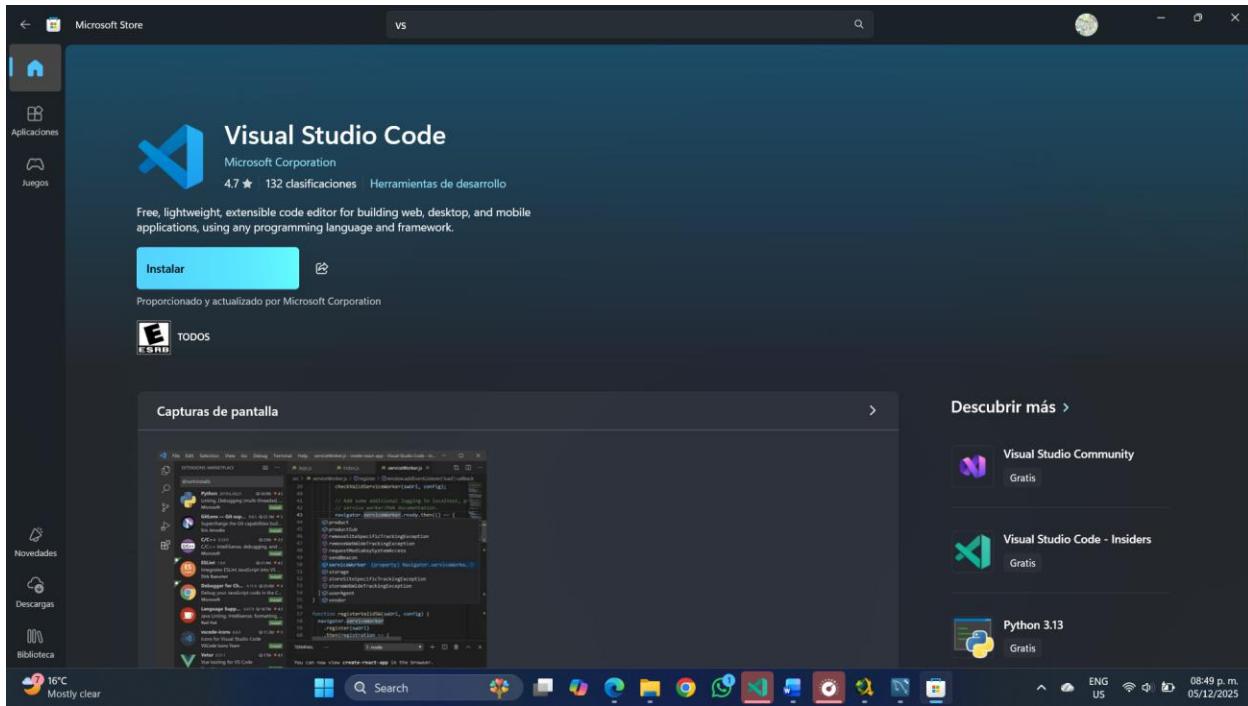
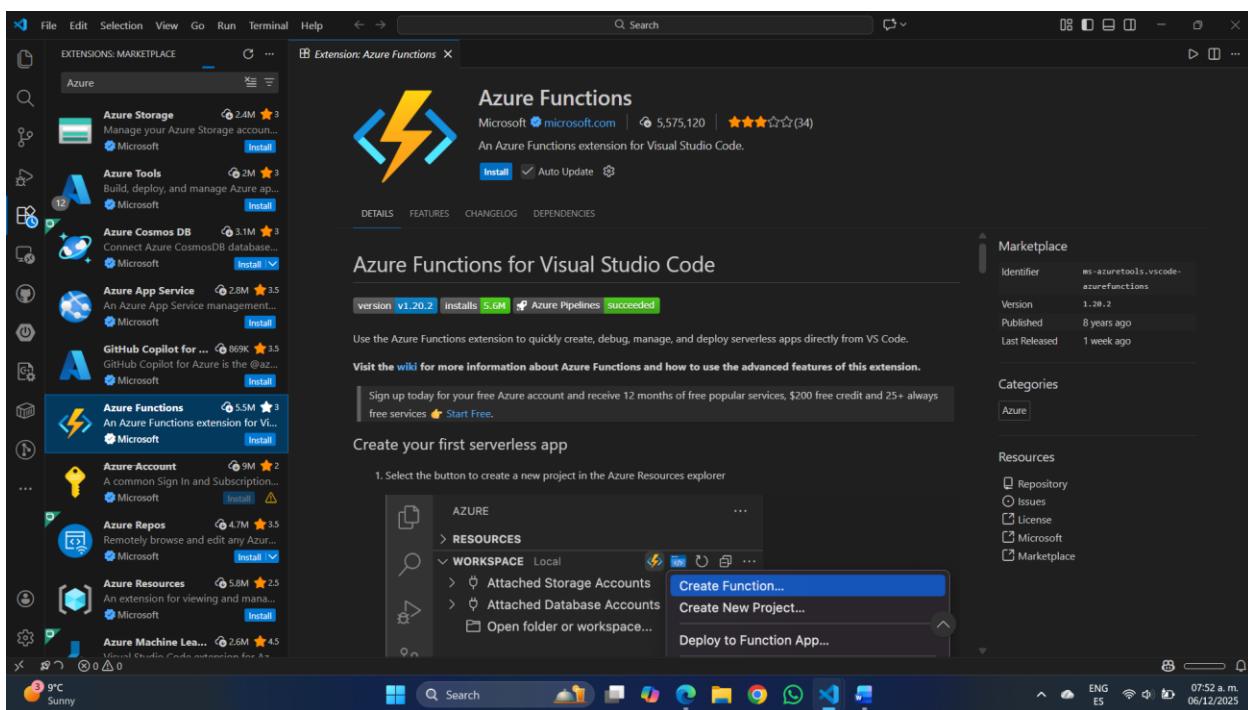
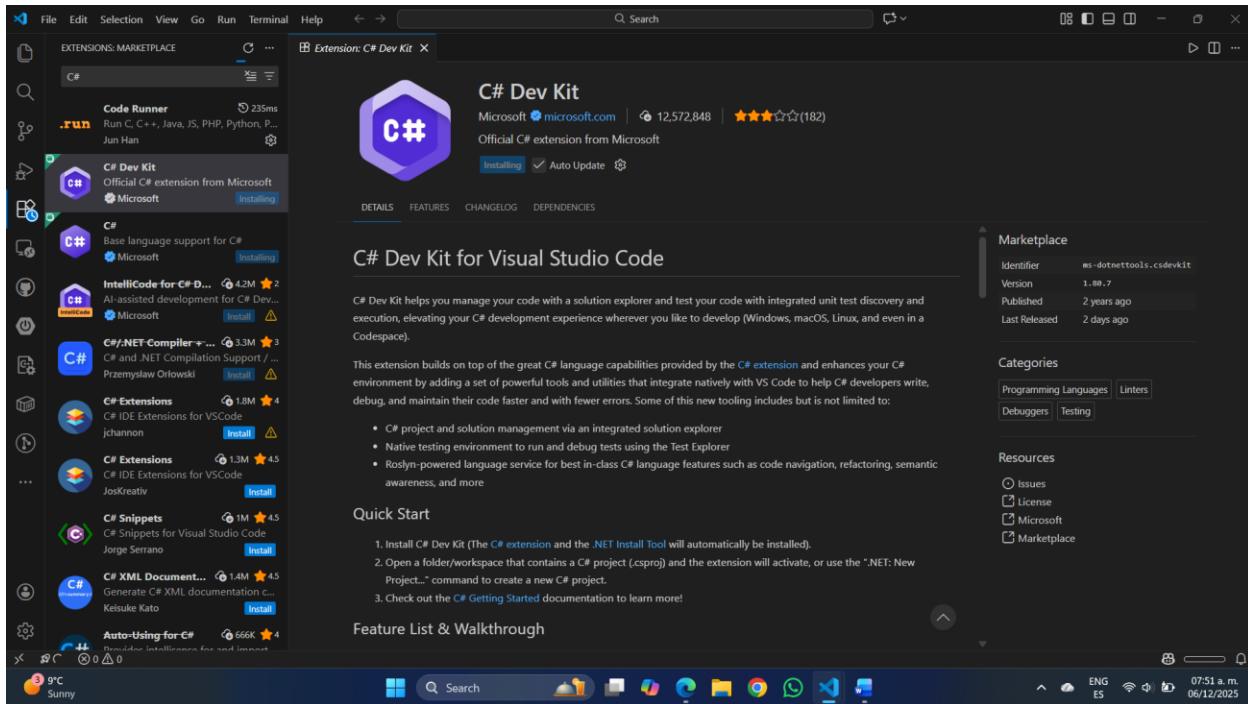


Figura 4.1 instalación de Visual Stdio Code desde Microsoft store.

- Se instaló la extensión “C#” para soporte de .NET.
- Se instaló la extensión “Azure Functions” para gestionar el proyecto serverless.
- Se instaló “Azure Account” para autenticación en Azure.

Extensión VS Code	Propósito
C# (ms-dotnettools)	Soporte lenguaje C# y .NET
Azure Functions	Creación y publicación Functions
Azure Account	Autenticación a Azure

Tabla 5 Extensiones de VS Code



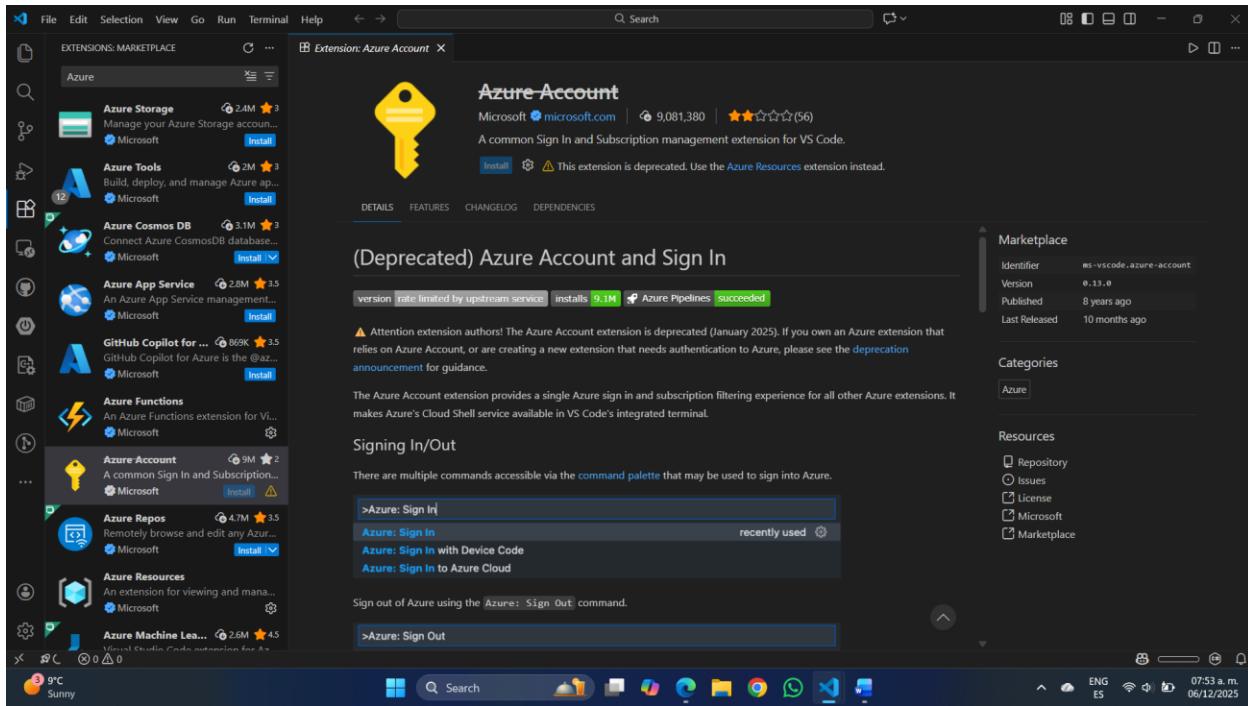


Figura 5. Visual Studio Code con extensiones C#, Azure Functions y Azure Account instaladas.

4.7 Instalación de .NET SDK

Se instaló .NET SDK 8.0.x (o 7.0.x si así se requiere). Se verificó la instalación ejecutando `dotnet --info` en una terminal. Se accedió a la terminal integrada de VS Code para futuras operaciones de paquetes.

- Se descargó el instalador .NET SDK y se ejecutó como administrador.
- Se verificó la versión instalada con `dotnet --version` y `dotnet --info`.

Comando	Resultado esperado
<code>dotnet --version</code>	8.0.x
<code>dotnet --info</code>	Información del runtime y SDK

Tabla 6 Tabla de comandos

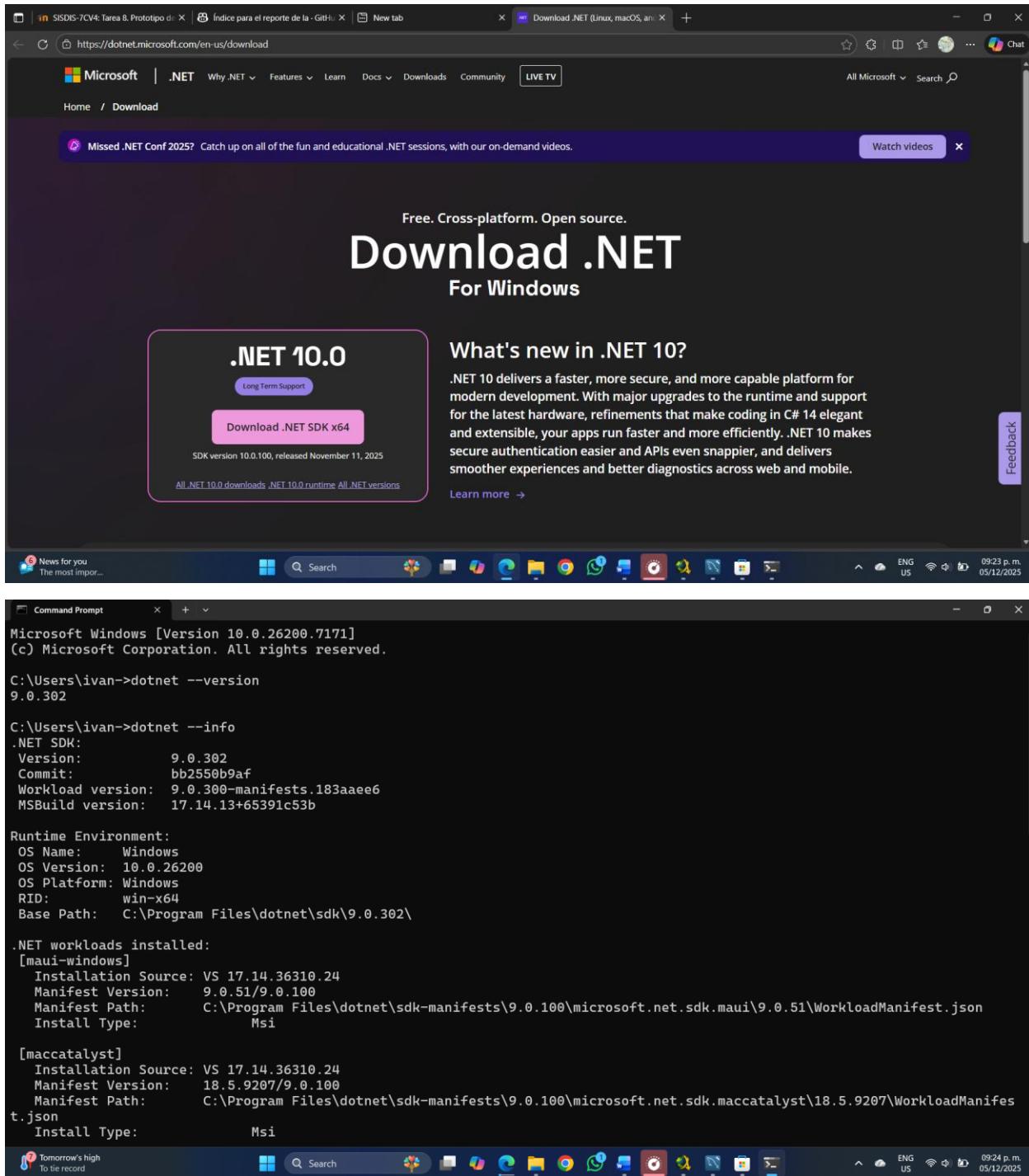


Figura 6. Verificación de .NET SDK instalado mediante la terminal.

4.8 Instalación de Azure Functions Core Tools

Se instaló Azure Functions Core Tools v4 para ejecutar Functions localmente. Se accedió a la guía oficial y se instaló usando npm o MSI (Windows). Se verificó con func --version.

- Opción MSI (Windows): descargar el instalador v4 desde la guía oficial.
- Opción npm: npm i -g azure-functions-core-tools@4 --unsafe-perm true (requiere Node.js).
- Se validó con func npm --version.

```
npm i azure-functions-core-tools
Microsoft Windows [Version 10.0.26200.7171]
(c) Microsoft Corporation. All rights reserved.

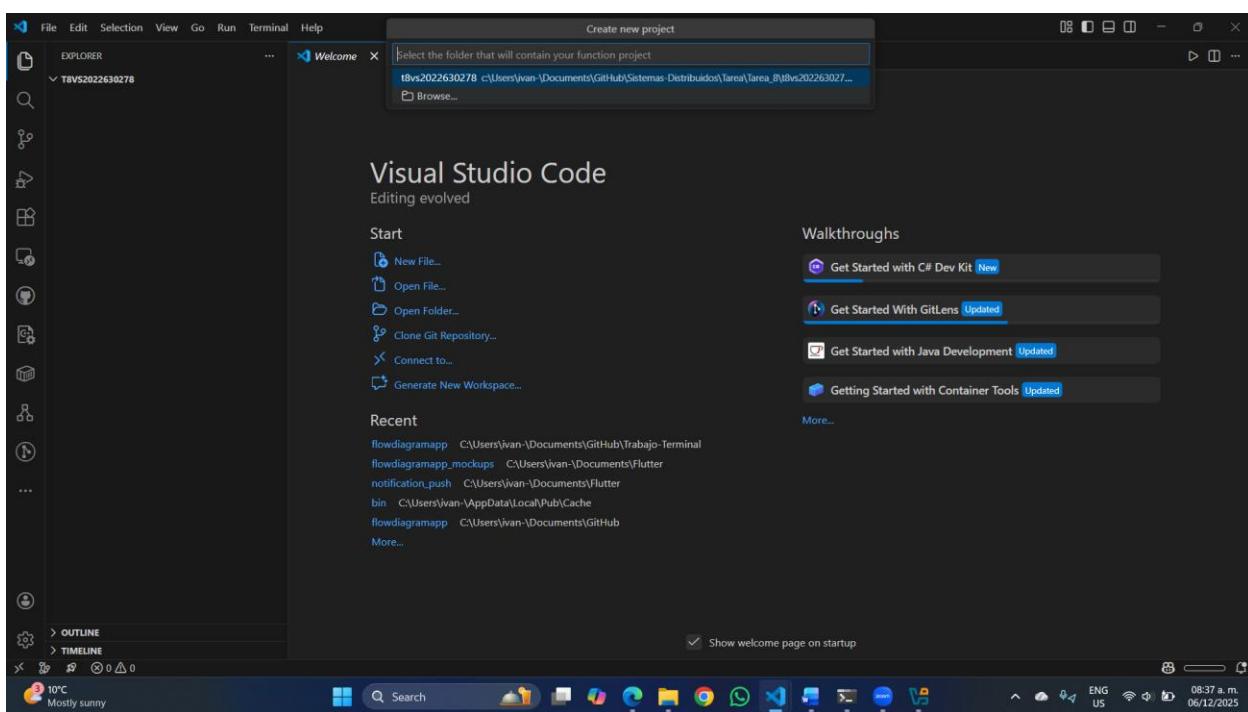
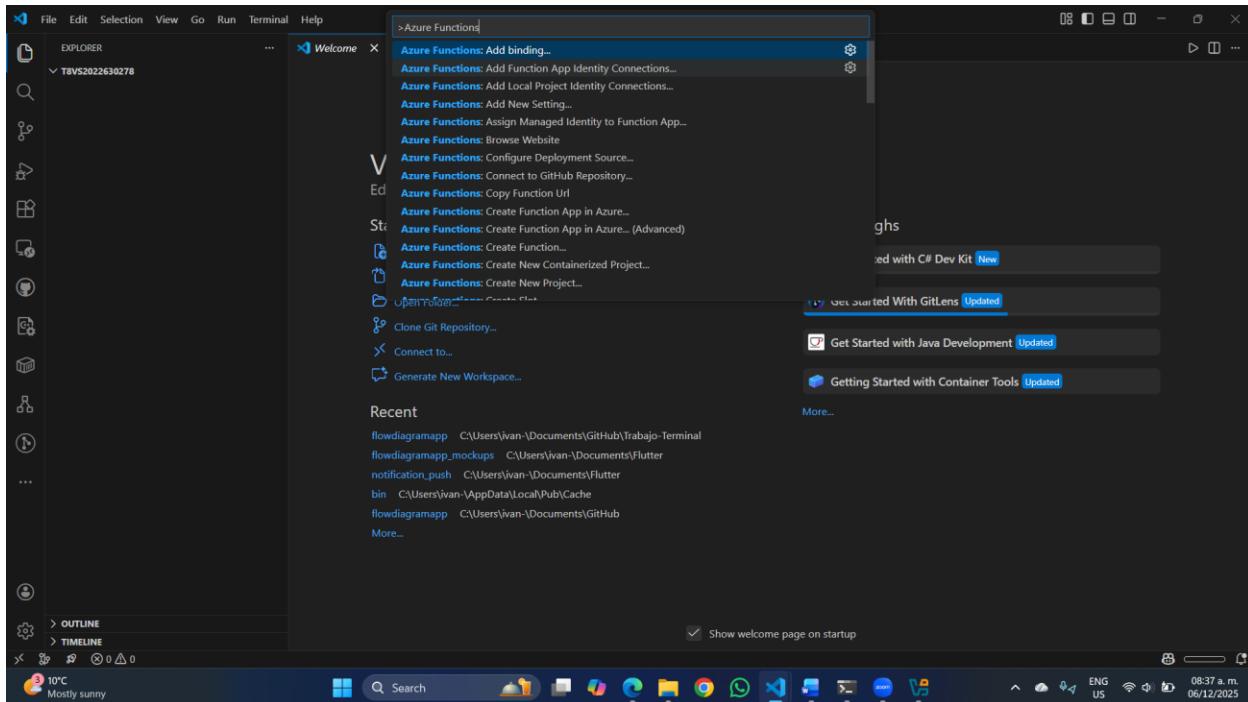
C:\Users\ivan->npm i -g azure-functions-core-tools@4 --unsafe-perm true
npm warn Unknown cli config "--unsafe-perm". This will stop working in the next major version of npm.
```

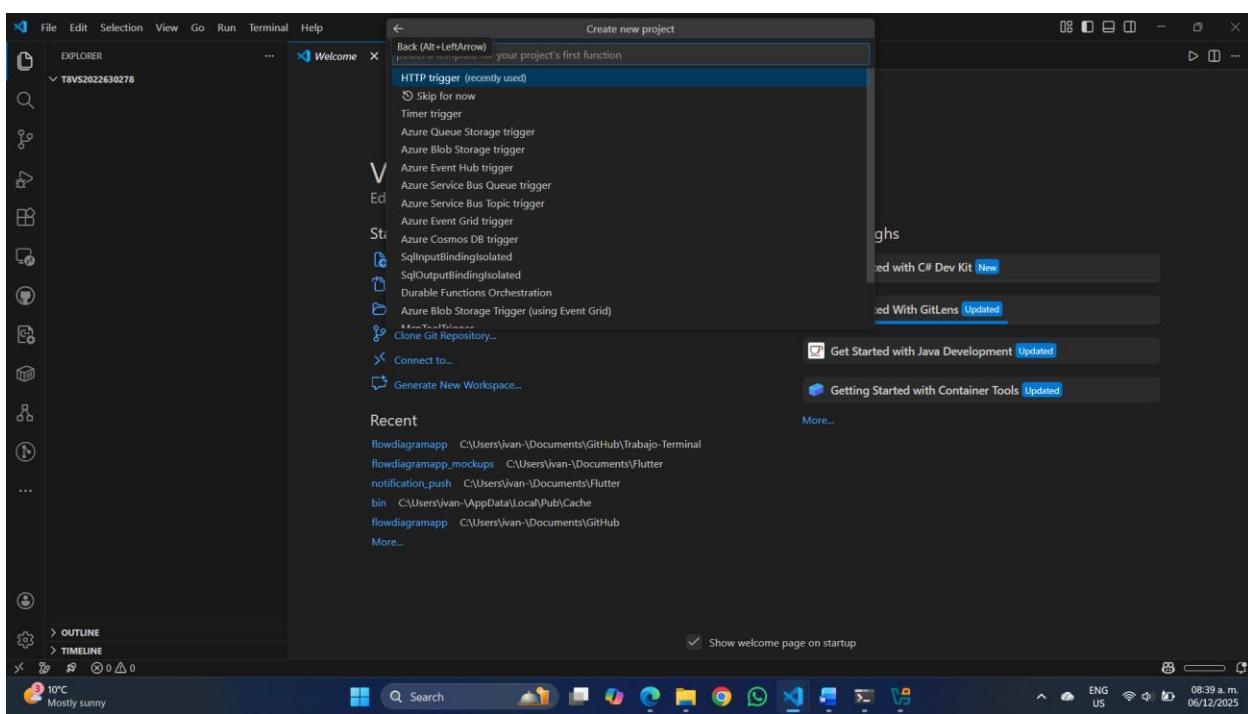
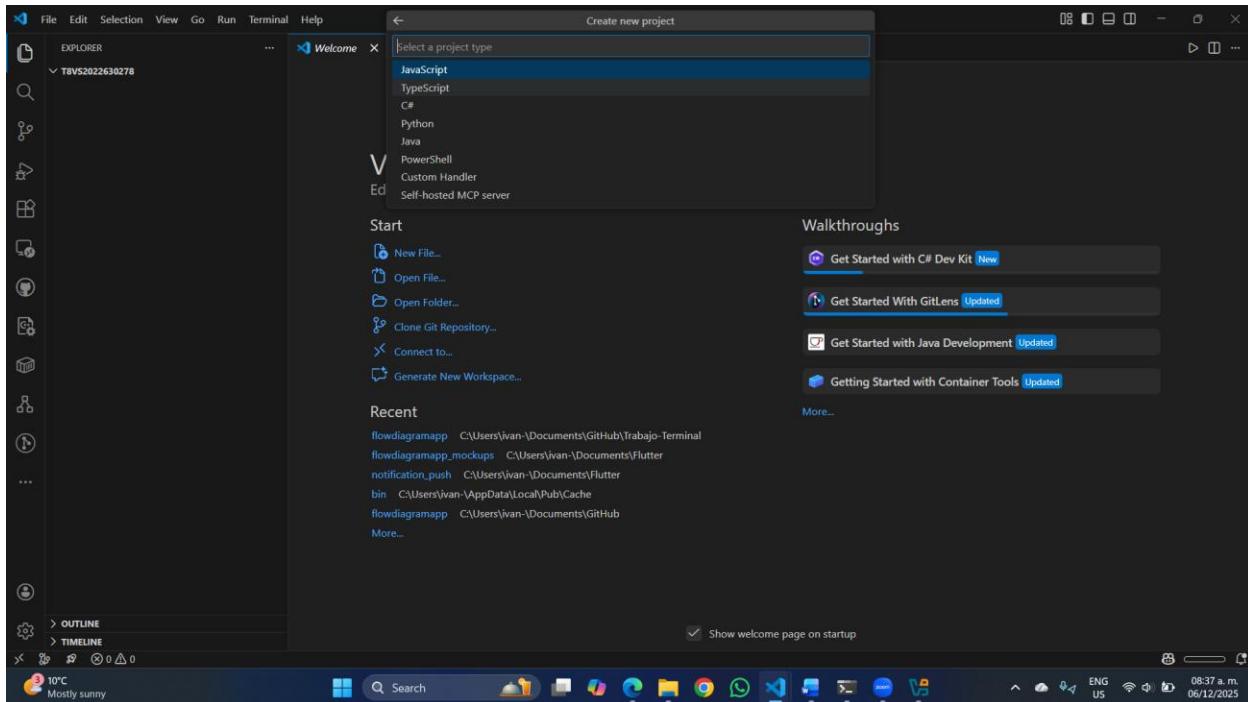
Figura 7. Instalación y verificación de Azure Functions Core Tools v4.

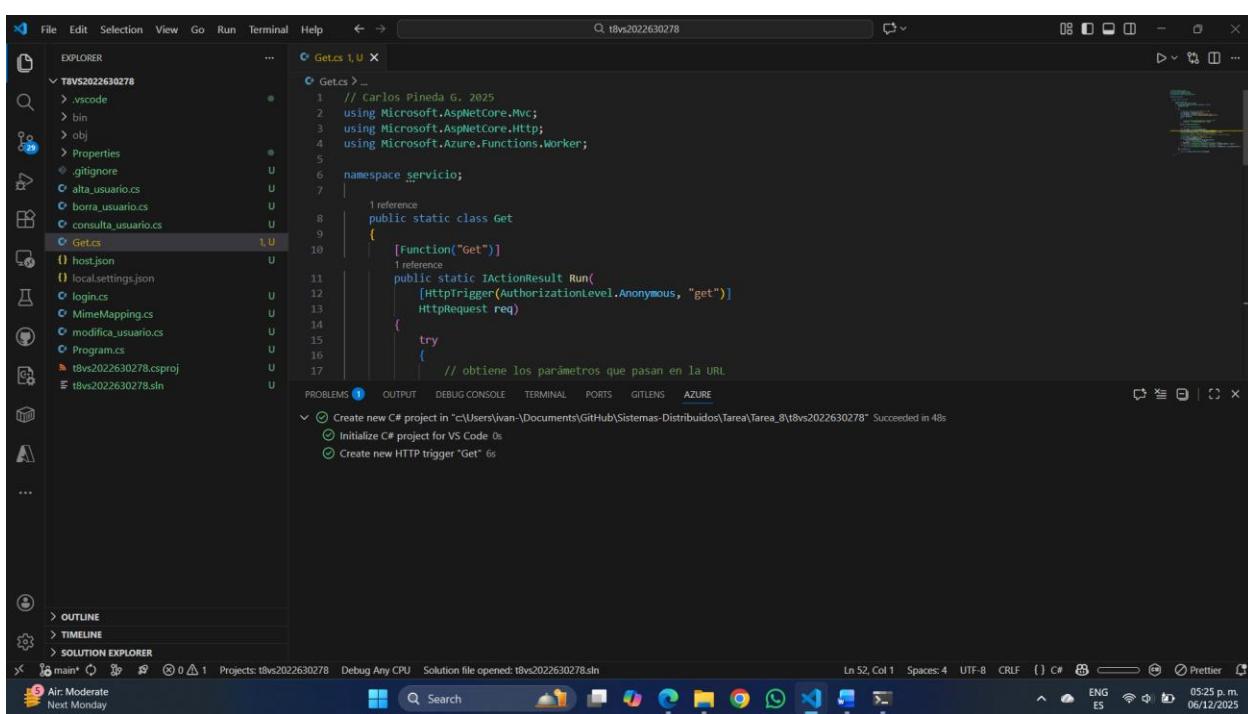
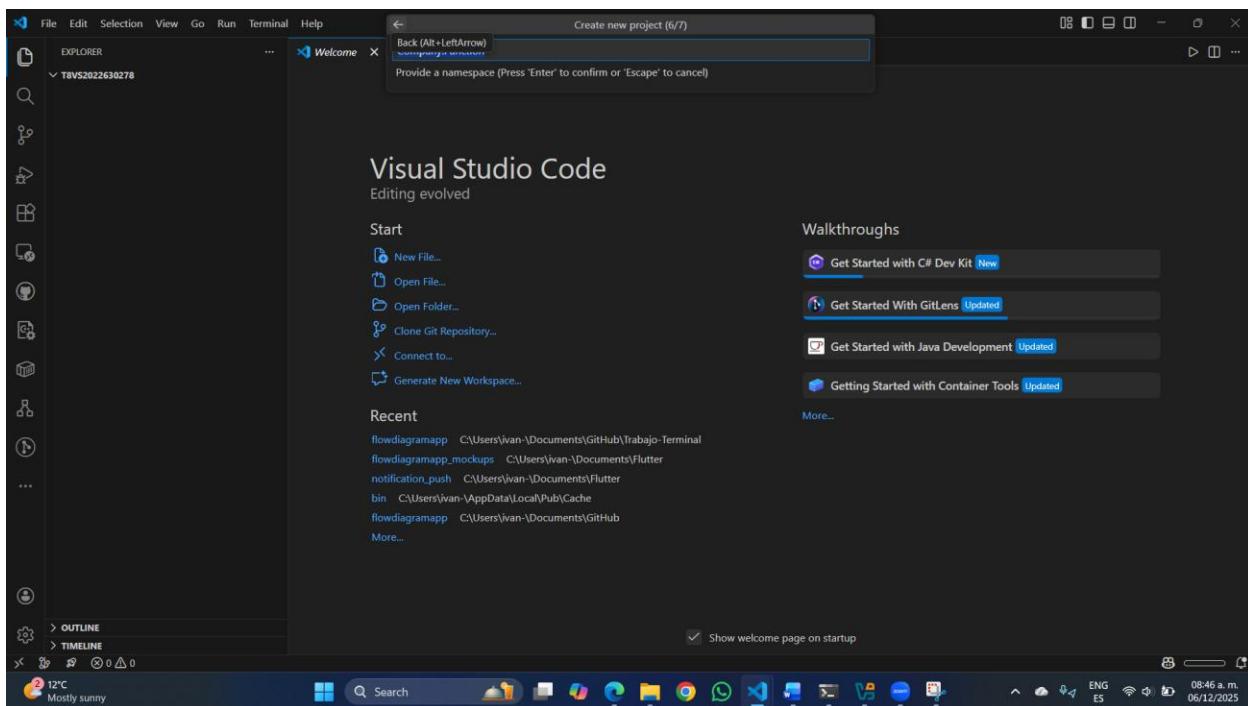
4.9 Creación del proyecto Azure Functions en Visual Studio Code

Se creó el proyecto de Azure Functions (C#, HTTP Trigger) en la carpeta t8vs2022630278. Se copió el contenido del back-end.zip al directorio del proyecto y se integraron las funciones base suministradas por el docente.

- Se accedió a VS Code y se creó el proyecto Functions (HTTP Trigger anónimo).
- Se copiaron los archivos: Get.cs, MimeMapping.cs, alta_usuario.cs, borra_usuario.cs, consulta_usuario.cs, login.cs, modifica_usuario.cs.
- Se compiló el proyecto para verificar que no existan errores.







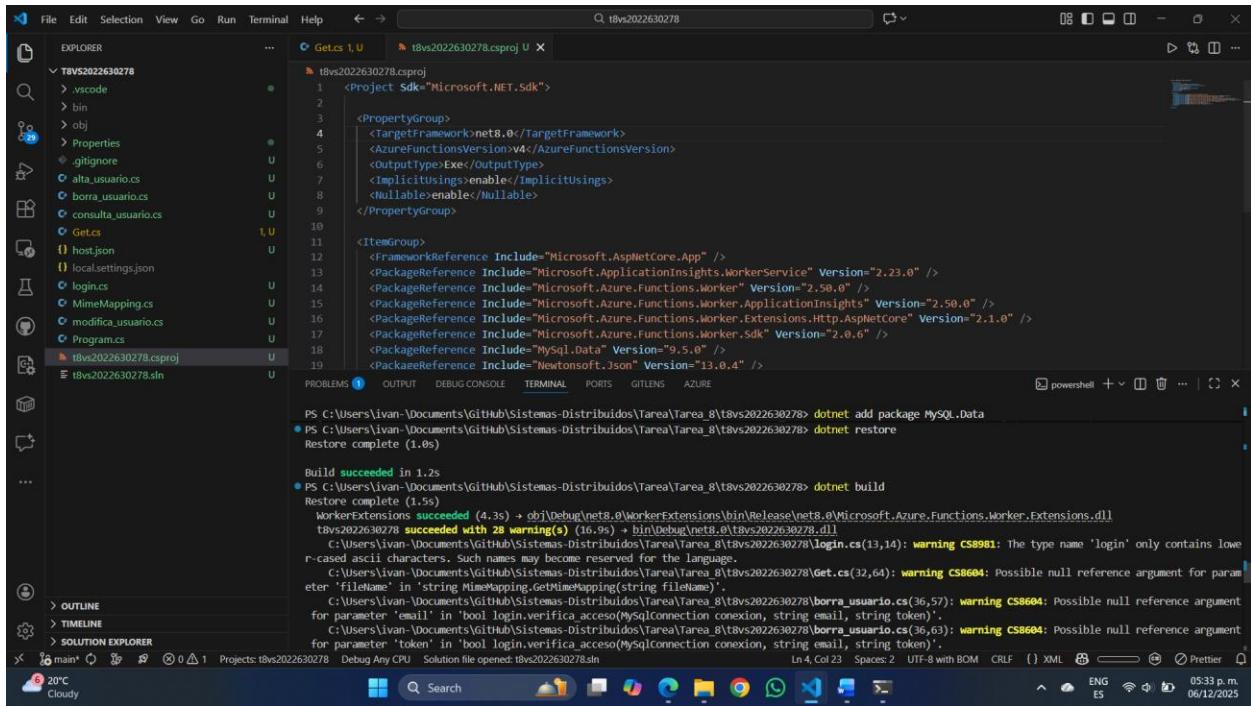


Figura 8. Estructura del proyecto t8vs2022630278 con funciones del back-end integradas.

4.10 Instalación de paquetes .NET requeridos

Se instalaron los paquetes Newtonsoft.Json y MySQL.Data desde la terminal de VS Code para habilitar la serialización JSON y la conectividad a MySQL. Se accedió al proyecto y se ejecutaron los comandos:

```
dotnet add package Newtonsoft.Json
```

```
dotnet add package MySQL.Data
```

```
dotnet restore
```

```
dotnet build
```

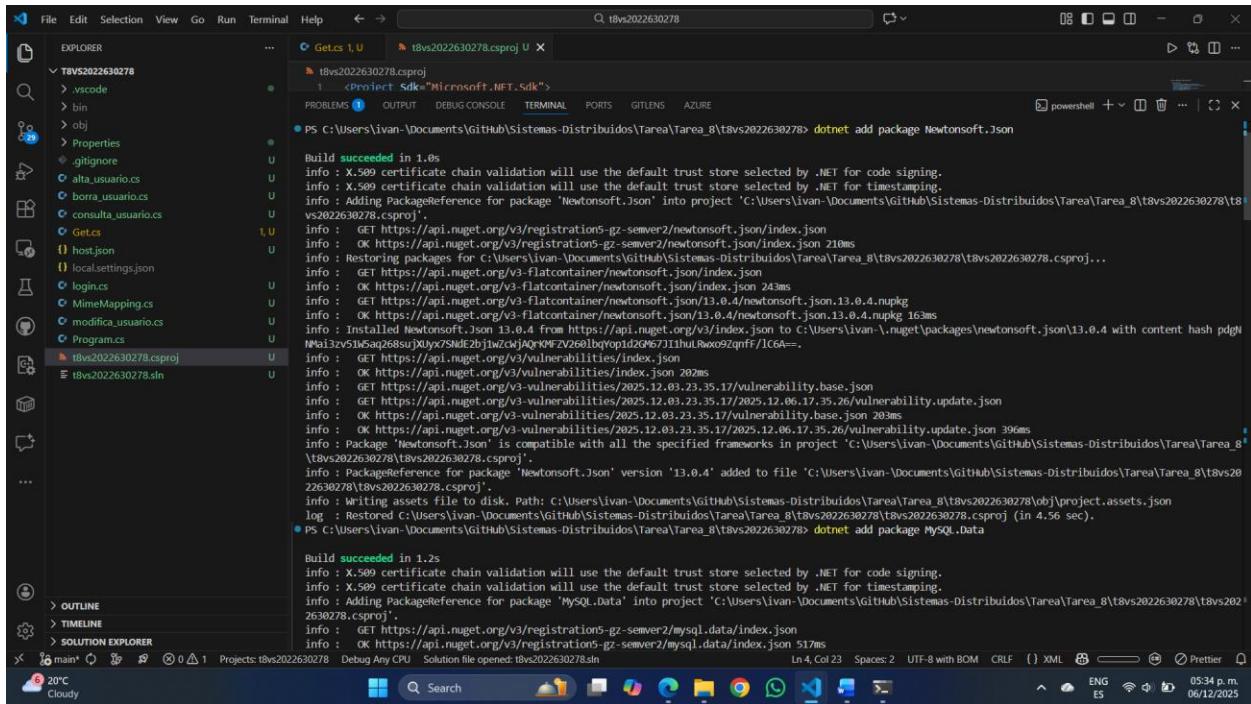


Figura 9. Instalación y restauración de paquetes .NET en el proyecto Functions.

4.11 Configuración de variables locales y front-end

Se creó y configuró local.settings.json con las variables de entorno para pruebas locales: Server, UserID, Password, Database y ROOT. Se colocó el front-end en una carpeta local y se verificó que la función Get sirva los archivos correctamente.

Ejemplo de local.settings.json (valores de ejemplo locales):

```
{
  "IsEncrypted": false,
  "Values": {
    "AzureWebJobsStorage": "UseDevelopmentStorage=true",
    "FUNCTIONS_WORKER_RUNTIME": "dotnet-isolated",
    "Server": "localhost",
    "UserID": "hugo",
```

```

    "Password": "AQUI-VA-LA-CONTRASEÑA-DEL-USUARIO-HUGO",
    "Database": "servicio_web",
    "ROOT": "C:\\t8vs2022630278\\front-end"
}

}

```

Variable	Valor (ejemplo local)	Descripción
Server	localhost	Host local de MySQL
UserID	hugo	Usuario creado para pruebas
Password	AQUI-VA-LA-CONTRASEÑA-DEL-USUARIO-HUGO	Contraseña del usuario local
Database	servicio_web	Nombre de la base de datos
ROOT	C:\\t8vs2022630278\\front-end	Ruta del front-end para servir con la función Get

Tabla 7 Tabla de variables

Se colocó en C:\\t8vs2022630278\\front-end:

- prueba.html
- WSClient.js
- usuario_sin_foto.png

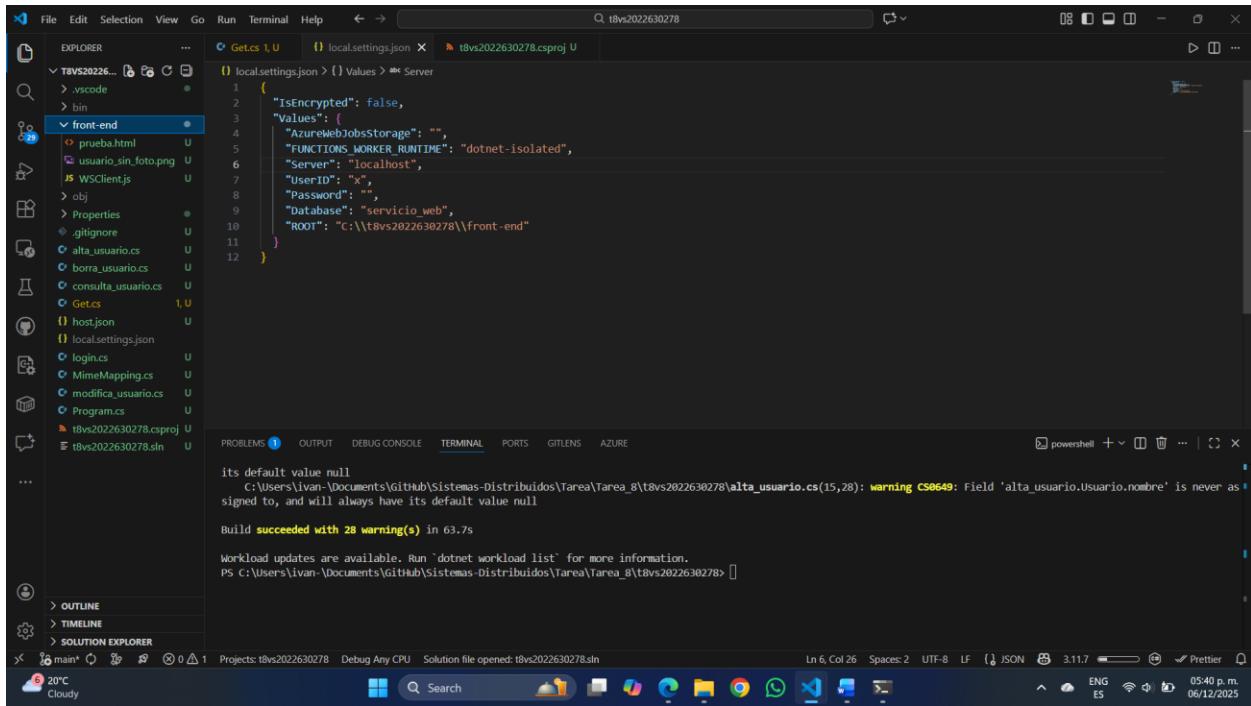
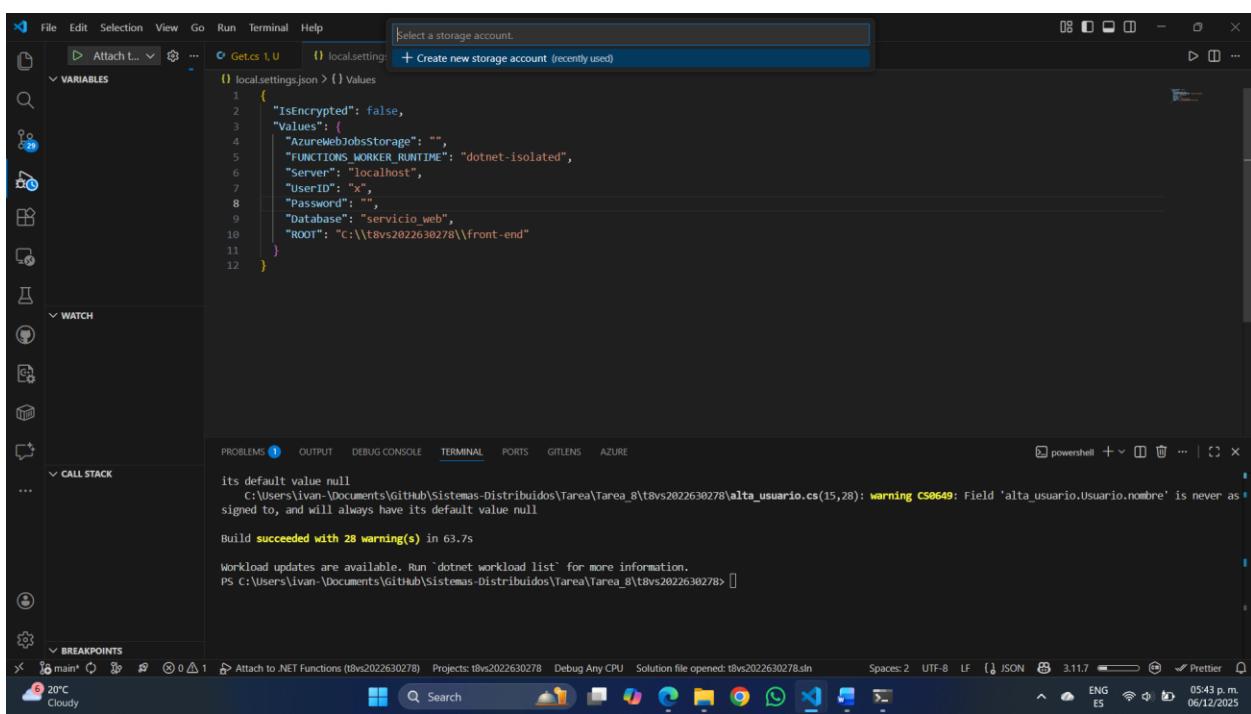
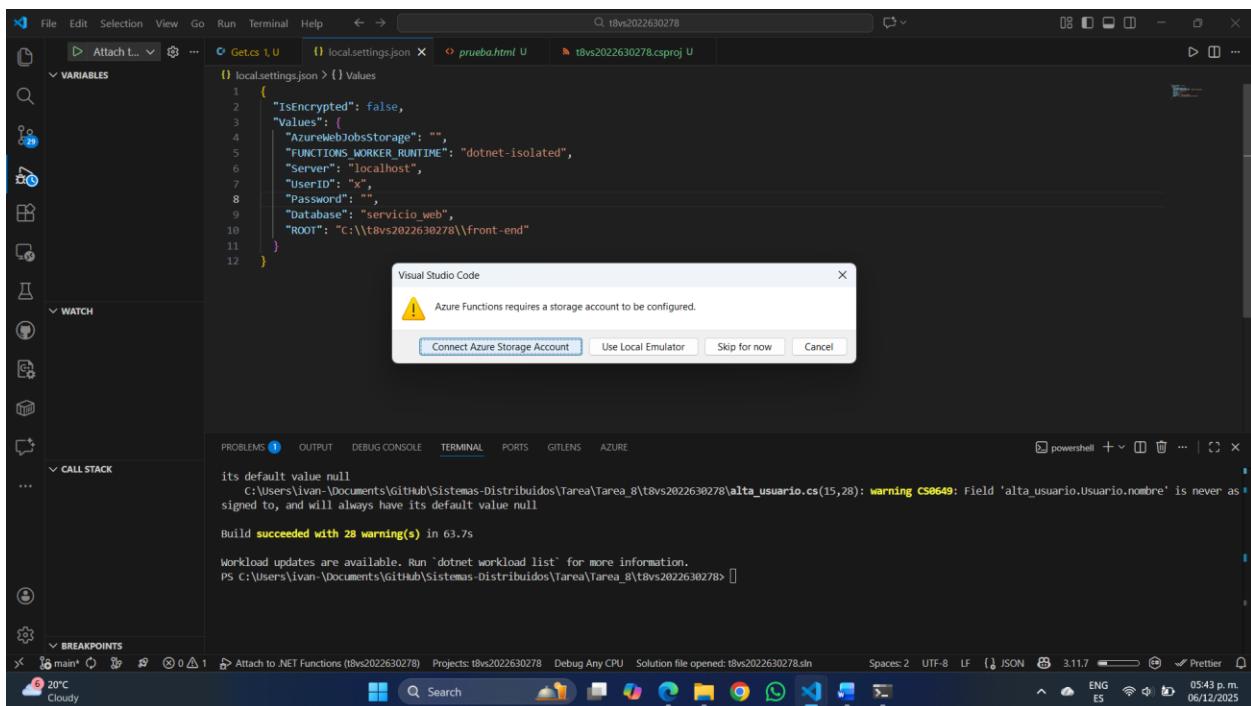


Figura 10. Configuración de local.settings.json y preparación de la carpeta del front-end.

4.12 Pruebas locales iniciales del front-end y back-end

Se accedió al proyecto Functions y se inició localmente con func start o F5 en VS Code.
Se abrió el navegador en la URL:

<http://localhost:7071/api/Get?nombre=/prueba.html>



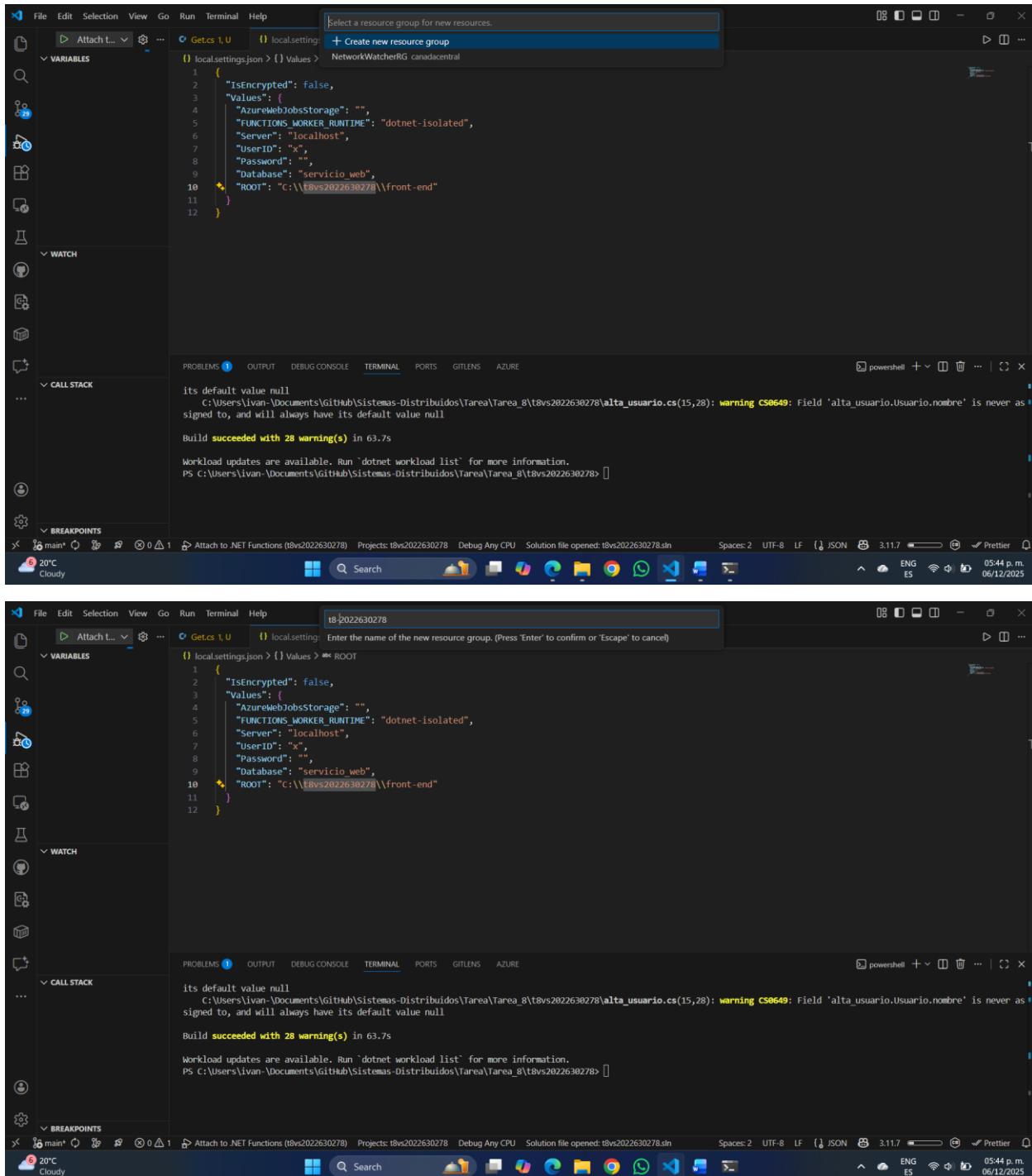


Figura 10. Compilación del proyecto.

Se realizó el flujo de registro y login de usuario y se validaron las respuestas JSON. Se prepararon los comandos curl para cada función a fin de incluirlos más adelante en el reporte de evidencias.

Prueba	URL / Operación	Resultado esperado
Cargar front-end	/api/Get?nombre=/prueba.html	Render de la app en navegador
Registro usuario	POST /api/alta_usuario	200 y mensaje de confirmación
Login	GET /api/login	200 y token
Consulta perfil	GET /api/consulta_usuario	JSON con datos del usuario
Modificar perfil	PUT /api/modifica_usuario	200 y mensaje de modificación
Borrar usuario	DELETE /api/borra_usuario	200 y mensaje de borrado

Tabla 8 Tabla de pruebas

```

# 1. Navega a la carpeta del proyecto
cd c:\Users\ivan-\Documents\GitHub\Sistemas-Distribuidos\Tarea\Tarea_8\t8vs2022630278

# 2. Restaura dependencias
dotnet restore

# 3. Construye (opcional, se hace al ejecutar dotnet build)

# 4. Ejecuta el host local
func start

Si todo está bien, verás:

Azure Functions Core Tools
Now listening on: http://localhost:7071

Luego pruebas con:
RestMethod "http://localhost:7071/api/Get"

¿Qué falta concretamente?
• ¿Tienes .NET 8 SDK instalado?
• ¿Tienes Azure Functions Core Tools?
• ¿Dónde debe buscar la función los archivos?
(necesito la ruta para Root)
```

For detailed output, run func with --verbose flag.

[2025-12-06T23:52:11.371Z] Host lock lease acquired by instance ID '000000000000000000000000A01F54'.
[2025-12-06T23:53:06.556Z] Executing 'Functions.Login' (Reason='This function was programmatically called via the host APIs.', Id=3607a1b1-7353-499e-9e8a-c0e67735c22e)
[2025-12-06T23:53:07.484Z] Executing 'Functions.Login' (Succeeded, Id=3607a1b1-7353-499e-9e8a-c0e67735c22e, Duration=102ms)
PS C:\Users\ivan-\Documents\GitHub\Sistemas-Distribuidos\Tarea\Tarea_8\t8vs2022630278>

Figura 11. Navegador mostrando la app y ejecución del flujo de usuario contra el backend local.

5 Desarrollo

6 Back-end: Funciones de Azure implementadas

Se realizó la implementación del back-end utilizando Azure Functions en C#, conectando con MySQL PaaS mediante el paquete MySQL.Data y manejando la serialización con Newtonsoft.Json. Se accedió a la base de datos “servicio_web” con variables de entorno y se respetó el patrón de respuestas JSON y códigos HTTP. En operaciones críticas del carrito y del stock se utilizaron transacciones para garantizar consistencia. A continuación se describen las funciones implementadas y su comportamiento.

6.1 Resumen de funciones y operaciones

La siguiente tabla resume cada función: método HTTP, endpoint, parámetros clave, operaciones en la base de datos, uso de transacción y códigos de respuesta.

Función	Método	Endpoint	Parámetros clave	Operaciones DB	Respuestas
alta_usuario	POST	/api/alta_usuario	email, password, nombre, apellidos, fecha, teléfono, genero, foto	INSERT en usuarios; INSERT opcional en fotos_usuarios	200 OK mensaje éxito; 400 Error con {"mensaje": "..."}
consulta_usuario	GET	/api/consulta_usuario	email, token	SELECT usuarios LEFT JOIN fotos_usuarios	200 OK con JSON usuario; 400 error
modifica_usuario	PUT	/api/modifica_usuario	email, token; body con datos y foto	UPDATE usuarios; opcional UPDATE password; DELETE/INSERT	200 OK mensaje; 400 error

				fotos_usuarios	
borra_usuario	DELETE	/api/borra_usuario	email, token	DELETE fotos_usuarios; DELETE usuarios	200 OK mensaje; 400 error
login	GET	/api/login	email, password (sha256)	SELECT credenciales; UPDATE token	200 OK {"token": "..."}; 400 "Acceso denegado"
alta_articulo	POST	/api/alta_articulo	nombre, descripcion, precio, cantidad, foto, id_usuario, token	INSERT stock; INSERT optional fotos_articulos	200 OK mensaje; 400 error acceso/validación
consulta_articulos	GET	/api/consulta_articulos	palabra_clave, id_usuario, token	SELECT con LIKE en nombre/descripcion; LEFT JOIN foto	200 OK arreglo de artículos; 400 error acceso
compra_articulo	POST	/api/compra_articulo	id_articulo, cantidad, id_usuario, token	SELECT stock; INSERT/UPDATE carrito_compra; UPDATE stock	200 OK; 400 "No hay suficientes artículos en stock"
elimina_articulo_carrito_compra	DELETE	/api/elimina_articulo_carrito_compra	id_usuario, id_articulo, token	SELECT cantidad del carrito; UPDATE stock (+); DELETE carrito	200 OK; 400 error acceso/consistencia
elimina_carrito_compra	DELETE	/api/elimina_carrito_compra	id_usuario, token	SELECT todos del carrito; UPDATE stock (+) por cada; DELETE	200 OK; 400 error

				carrito del usuario	
modifica_carrito_compra	PUT	/api/modifica_carrito_compra	id_articulo , incremento(+1/-1), id_usuario , token	SELECT stock y carrito; UPDATE carrito y stock	200 OK; 400 "No hay suficientes artículos en stock" / "No hay más artículos en el carrito"

Figura 9. Resumen de endpoints y operaciones del back-end.

6.2 alta_usuario (registro de usuarios)

Se realizó la función alta_usuario para registrar usuarios nuevos. Se validaron campos obligatorios, se insertó el registro en usuarios y, si el front-end envió foto en base64, se insertó en fotos_usuarios. Se utilizó una transacción para agrupar ambas operaciones y asegurar atomicidad.

- Se accedió a variables de entorno: Server, UserID, Password, Database.
- Se instaló MySQL.Data para la conexión y se empleó LastInsertedId para recuperar id_usuario.
- Se manejó OkObjectResult en éxito y BadRequestObjectResult con {"mensaje": "..."} en error.

```

ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$BASE/alta_usuario" -H "Content-Type: application/json" -d '{
  "email": "'$EMAIL'", "password": "'$PASSWORD_HASH'", "nombre": "Gustavo", "apellido_paterno": "Gonzalez", "apellido_materno": "Garcia", "fecha_nacimiento": "2000-01-15T12:00:00.0", "telefono": "5512345678", "genero": "M", "foto": null
}
{"mensaje": "Se dio de alta el usuario"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ BASE="http://localhost:7071/api"; EMAIL="alumno@example.com"; PASSWORD_HASH="d3c9bb2c3c7b09d0a0ad75f67a6e1dbaf7b6e6e0d3b3b4b1ccb3d9f2a"
ivan-@LAPTOP-X515JA MINGW64 ~
$ 
ivan-@LAPTOP-X515JA MINGW64 ~
$ 
ivan-@LAPTOP-X515JA MINGW64 ~
$ 
ivan-@LAPTOP-X515JA MINGW64 ~
$ 
ivan-@LAPTOP-X515JA MINGW64 ~

```

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: servicio_web

MANAGEMENT

- Server Status
- Client Connections
- Users and Privileges
- Status and System Variables
- Data Export
- Data Import/Restore

INSTANCE

- Startup / Shutdown
- Server Logs
- Options File

PERFORMANCE

- Dashboard
- Performance Reports
- Performance Schema Setup

Result Grid | Filter Rows | Edit | Export/Import | Wrap Cell Content |

sql

58 • SELECT user, host, account_locked, password_expired, plugin

59 • FROM mysql.user

60 • ORDER BY host, user;

61 •

62 •

63 • describe usuarios;

64 • select * from usuarios;

Result Grid | Filter Rows | Edit | Export/Import | Wrap Cell Content |

id_usuario	password	token	email	nombre	apellido_paterno	apellido_materno	fecha_nacimiento	telefono
1	573b70e3383ad9ea3ac7c4e7c62f69d47d762c4...	UpPfIVQJAUJTBeZDjwz0c...	x@x1.com	x1@x1.com	x1@x1.com	x1@x1.com	2025-12-08 00:20:00	1111
2	4e8b9*5d3c9bb2c3c7b09d0a0ad75f67a6e1dbaf7b6e6e0d3b3b4b1ccb3d9f2a...	U3L...	alumno@example.com	Gustavo	Gonzalez	Garcia	2000-01-15 12:00:00	5512
3	4e8b9*5d3c9bb2c3c7b09d0a0ad75f67a6e1dbaf7b6e6e0d3b3b4b1ccb3d9f2a...	U3L...	alumno@example.com	Gustavo	Gonzalez	Garcia	2000-01-15 12:00:00	5512

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Administration Schemas

usuarios 8

Output

Action Output

#	Time	Action	Message	Duration / Fetch
7	18:23:30	use servicio_web	0 row(s) affected	0.000 sec / 0.000 sec
8	18:23:35	describe usuarios	10 row(s) returned	0.000 sec / 0.000 sec
9	18:23:52	select * from usuarios LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
10	18:29:01	SELECT user, host, account_locked, password_expired, plugin FROM mysql.user ORDER BY host, user LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec
11	20:02:44	select * from usuarios LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
12	20:22:35	select * from usuarios LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec

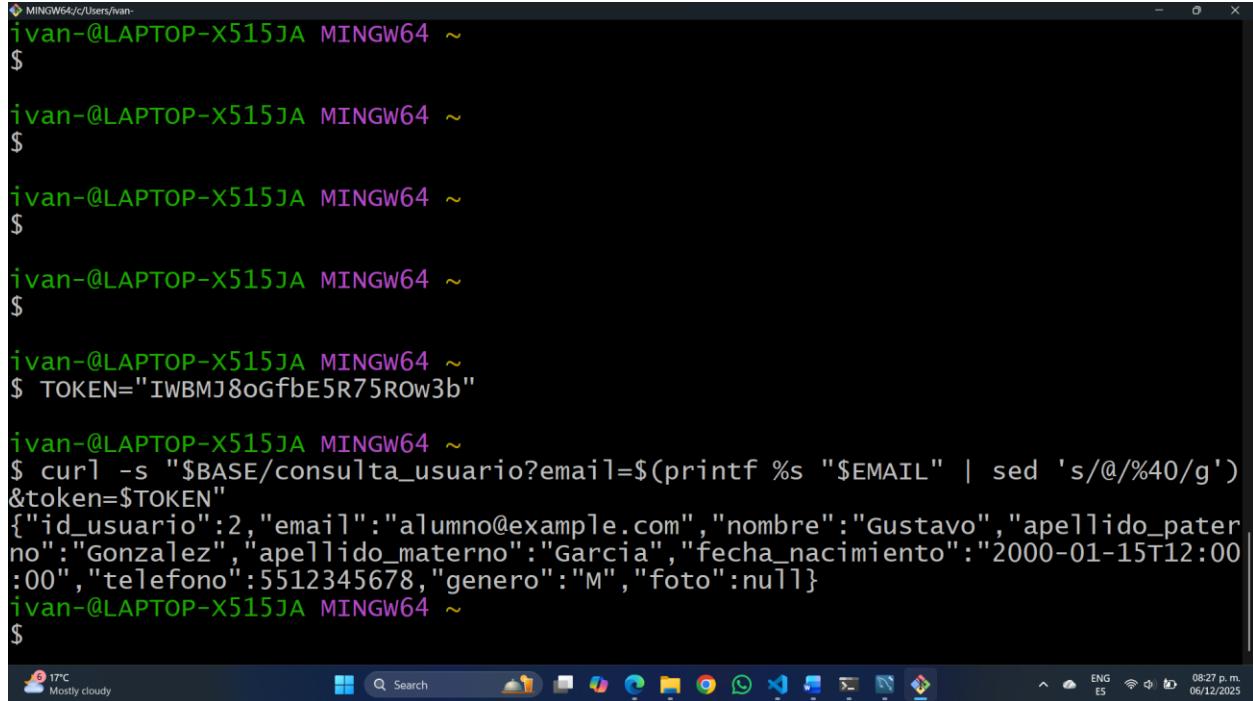
Object Info Session

Figura 10. Código de alta_usuario y consulta de inserción exitosa en la base de datos.

6.3 consulta_usuario (perfil del usuario)

Se realizó la función consulta_usuario que, tras verificar el acceso con verifica_acceso(email, token), recuperó los datos del usuario y la foto (si existe), devolviendo un JSON con todos los campos requeridos por el front-end.

- Se accedió a la BD con MySqlConnection.
- Se utilizó LEFT JOIN con fotos_usuarios para un resultado opcional de imagen.
- Se cerró el MySqlDataReader antes de ejecutar otros comandos.



```

MINGW64:/c/Users/wan-
ivan-@LAPTOP-X515JA MINGW64 ~
$ TOKEN="IWBMJ8oGfbE5R75Row3b"

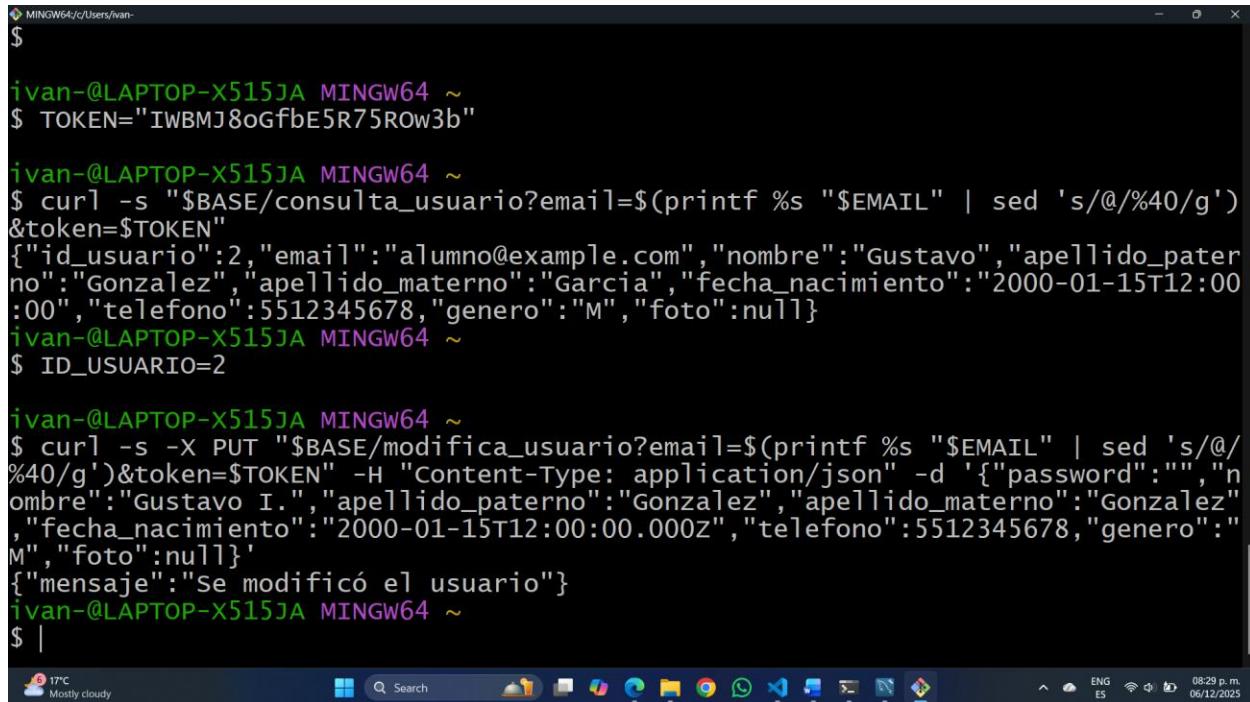
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s "$BASE/consulta_usuario?email=$(printf %s "$EMAIL" | sed 's/@/%40/g')&token=$TOKEN"
{"id_usuario":2,"email":"alumno@example.com","nombre":"Gustavo","apellido_paterno":"Gonzalez","apellido_materno":"Garcia","fecha_nacimiento":"2000-01-15T12:00:00","telefono":5512345678,"genero":"M","foto":null}
ivan-@LAPTOP-X515JA MINGW64 ~
$
```

Figura 11. Respuesta JSON completa del perfil del usuario en pruebas locales.

6.4 modifica_usuario (actualización de perfil y foto)

Se realizó la función modifica_usuario para actualizar el perfil. Tras verificar el acceso, se ejecutó un UPDATE de los datos básicos; si el cliente envió un password no vacío, se actualizó la contraseña. Se reemplazó la foto mediante DELETE y INSERT en fotos_usuarios. Todo se ejecutó dentro de una transacción.

- Se accedió a email y token por query string y al body JSON con nuevos valores.
- Se validaron campos obligatorios y se procesó la imagen en base64 si se incluyó.
- Se manejó commit/rollback ante excepciones.



```
MINGW64/c/Users/ivan-
$ 
ivan-@LAPTOP-X515JA MINGW64 ~
$ TOKEN="IWBMJ8oGfbE5R75Row3b"

ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s "$BASE/consulta_usuario?email=$(printf %s "$EMAIL" | sed 's/@/%40/g')&token=$TOKEN"
{"id_usuario":2,"email":"alumno@example.com","nombre":"Gustavo","apellido_paterno":"Gonzalez","apellido_materno":"Garcia","fecha_nacimiento":"2000-01-15T12:00:00","telefono":5512345678,"genero":"M","foto":null}
ivan-@LAPTOP-X515JA MINGW64 ~
$ ID_USUARIO=2

ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X PUT "$BASE/modifica_usuario?email=$(printf %s "$EMAIL" | sed 's/@/%40/g')&token=$TOKEN" -H "Content-Type: application/json" -d '{"password":"","nombre":"Gustavo I.", "apellido_paterno":"Gonzalez", "apellido_materno":"Gonzalez", "fecha_nacimiento":"2000-01-15T12:00:00.000Z", "telefono":5512345678, "genero":"M", "foto":null}'
{"mensaje":"Se modificó el usuario"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ |
```

Figura 12. Evidencia de actualización de perfil y cambio de foto con transacción.

6.5 borra_usuario (eliminación del perfil)

Se realizó la función `borra_usuario` que eliminó primero las fotos del usuario en `fotos_usuarios` y posteriormente su registro en `usuarios`, tras verificar el acceso. Se utilizó transacción para asegurar que ambas operaciones se aplicaran de forma consistente.

- Se accedió con email y token del usuario autenticado.
- Se ejecutó DELETE en tabla de fotos y luego en usuarios.
- Se confirmaron respuestas con código 200 y mensaje de éxito.

```

ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s "$BASE/consulta_usuario?email=$(printf %s \"$EMAIL\" | sed 's/@/%40/g')&token=$TOKEN"
{"id_usuario":2,"email":"alumno@example.com","nombre":"Gustavo","apellido_paterno":"Gonzalez","apellido_materno":"Garcia","fecha_nacimiento":"2000-01-15T12:00:00","telefono":5512345678,"genero":"M","foto":null}
ivan-@LAPTOP-X515JA MINGW64 ~
$ ID_USUARIO=2

ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X PUT "$BASE/modifica_usuario?email=$(printf %s \"$EMAIL\" | sed 's/@/%40/g')&token=$TOKEN" -H "Content-Type: application/json" -d '{"password":"","nombre":"Gustavo I.", "apellido_paterno":"Gonzalez", "apellido_materno":"Gonzalez", "fecha_nacimiento":"2000-01-15T12:00:00.000Z", "telefono":5512345678, "genero":"M", "foto":null}'
{"mensaje":"Se modificó el usuario"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X DELETE "$BASE/borra_usuario?email=$(printf %s \"$EMAIL\" | sed 's/@/%40/g')&token=$TOKEN"
{"mensaje":"se borró el usuario"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ |

17°C Mostly cloudy
MINGW64:~ ivan-@LAPTOP-X515JA ~ 08:30 p.m. 06/12/2025
ENG ES WiFi

MySQL Workbench Local instance MySQL80 X
File Edit View Query Database Server Tools Scripting Help
Navigator: servicio_web*
MANAGEMENT
- Server Status
- Client Connections
- Users and Privileges
- Status and System Variables
- Data Export
- Data Import/Restore
INSTANCE
- Startup / Shutdown
- Server Logs
- Options File
PERFORMANCE
- Dashboard
- Performance Reports
- Performance Schema Setup
Result Grid | Filter Rows | Edit | Wrap Cell Content | Result Grid
| id_usuario | password | token | email | nombre | apellido_paterno | apellido_materno | fecha_nacimiento | telefono |
| 1 | 573b70e3383ad9ea3ac7c4e7c62f69d47d762c4... | UpPhvQJAUUTBxZDjwz0c | x1@x1.com | x1@x1.com | x1@x1.com | x1@x1.com | 2025-12-08 00:20:00 | 1111111111 |
| 1 | 573b70e3383ad9ea3ac7c4e7c62f69d47d762c4... | UpPhvQJAUUTBxZDjwz0c | x1@x1.com | x1@x1.com | x1@x1.com | x1@x1.com | 2025-12-08 00:20:00 | 1111111111 |
SQLAdditions
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Administration Schemas usuarios 13 X
Information Output
No object selected
Object Info Session
Action Output
# Time Action Message Duration / Fetch
12 20:22:35 select * from usuarios LIMIT 0, 1000 2 row(s) returned 0.000 sec / 0.000 sec
13 20:24:28 select * from usuarios LIMIT 0, 1000 2 row(s) returned 0.000 sec / 0.016 sec
14 20:29:40 select * from usuarios LIMIT 0, 1000 2 row(s) returned 0.000 sec / 0.000 sec
15 20:30:15 select * from fotos_usuarios LIMIT 0, 1000 0 row(s) returned 0.000 sec / 0.000 sec
16 20:30:21 select * from usuarios LIMIT 0, 1000 2 row(s) returned 0.000 sec / 0.000 sec
17 20:31:04 select * from usuarios LIMIT 0, 1000 1 row(s) returned 0.000 sec / 0.000 sec

```

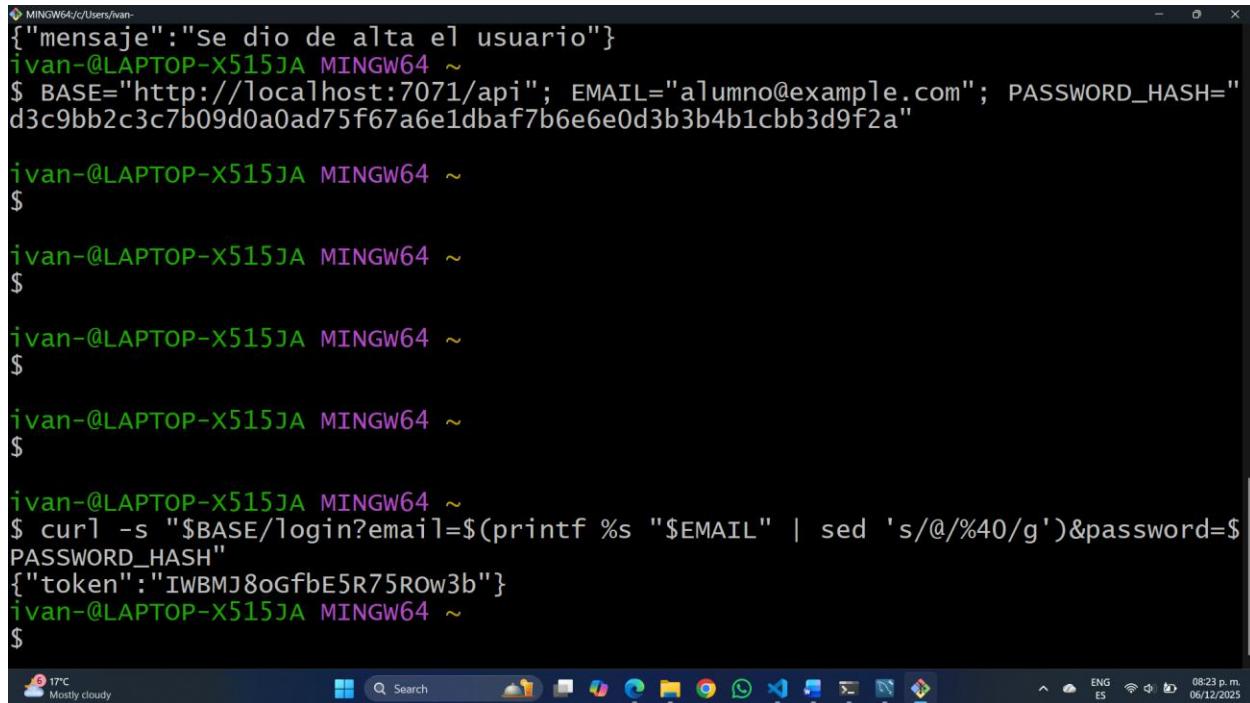
Figura 13. Ejecución de borra_usuario y validación del borrado en MySQL.

6.6 login y verifica_acceso (autenticación y seguridad)

Se realizó la función login que verificó credenciales (email y password SHA-256), generó un token alfanumérico y lo guardó en la tabla usuarios. Se implementaron dos

variantes de verifica_acceso (por email y por id_usuario) que se emplean en las demás funciones para autorizar operaciones.

- Se accedió a la BD y se actualizó el token mediante UPDATE.
- Se devolvió {"token": "..."} en éxito; se devolvió error con "Acceso denegado" en credenciales inválidas.
- Se instaló System.Security.Cryptography para generación de tokens.



```
MINGW64:/c/Users/ivan-
{"mensaje":"se dio de alta el usuario"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ BASE="http://localhost:7071/api"; EMAIL="alumno@example.com"; PASSWORD_HASH="d3c9bb2c3c7b09d0a0ad75f67a6e1dbaf7b6e6e0d3b3b4b1cbb3d9f2a"
ivan-@LAPTOP-X515JA MINGW64 ~
$ 
ivan-@LAPTOP-X515JA MINGW64 ~
$ 
ivan-@LAPTOP-X515JA MINGW64 ~
$ 
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s "$BASE/login?email=$(printf %s \"$EMAIL\" | sed 's/@/%40/g')&password=$PASSWORD_HASH"
>{"token":"IWBMJ8oGfbE5R75Row3b"}
ivan-@LAPTOP-X515JA MINGW64 ~
$
```

The screenshot shows a terminal window on a Windows operating system. The command `curl` is used to send a POST request to the endpoint `/login` of a local API. The request includes the user's email and password. The response is a JSON object containing a single key `token` with a long string value. The terminal window has a dark theme and is titled `MINGW64`. The taskbar at the bottom shows various application icons, and the system tray indicates the date and time as 08:23 p.m. on 06/12/2025.

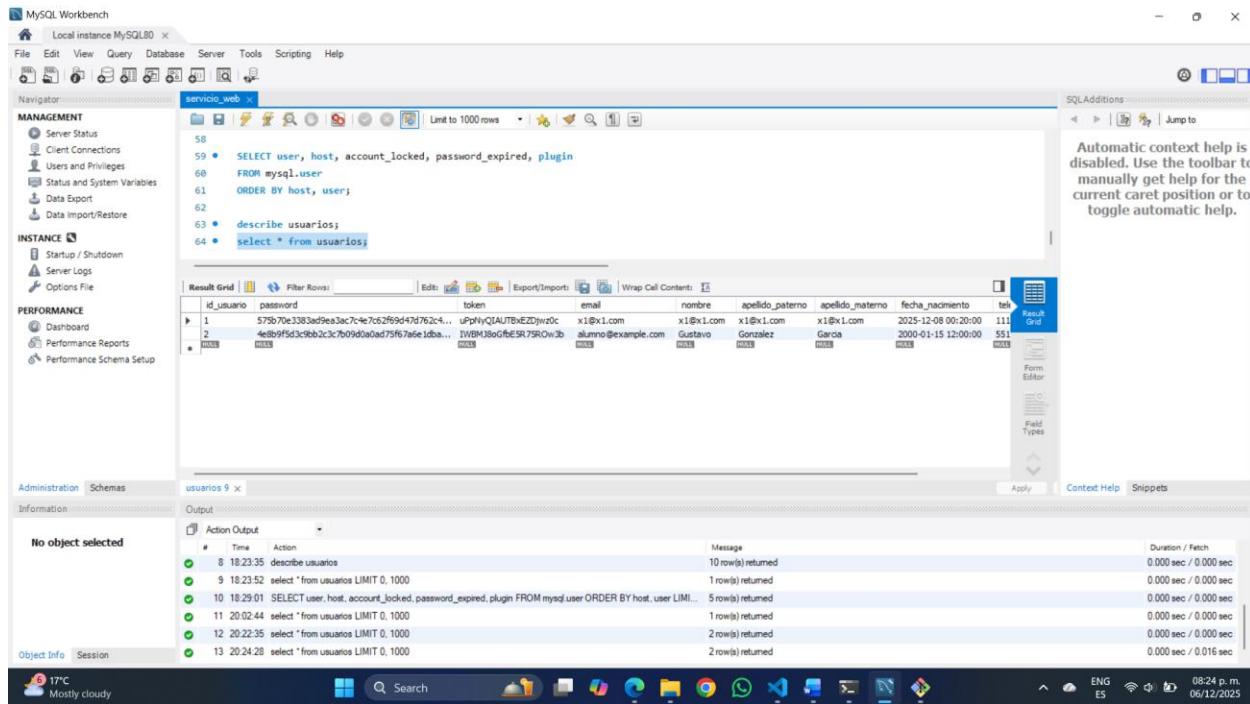
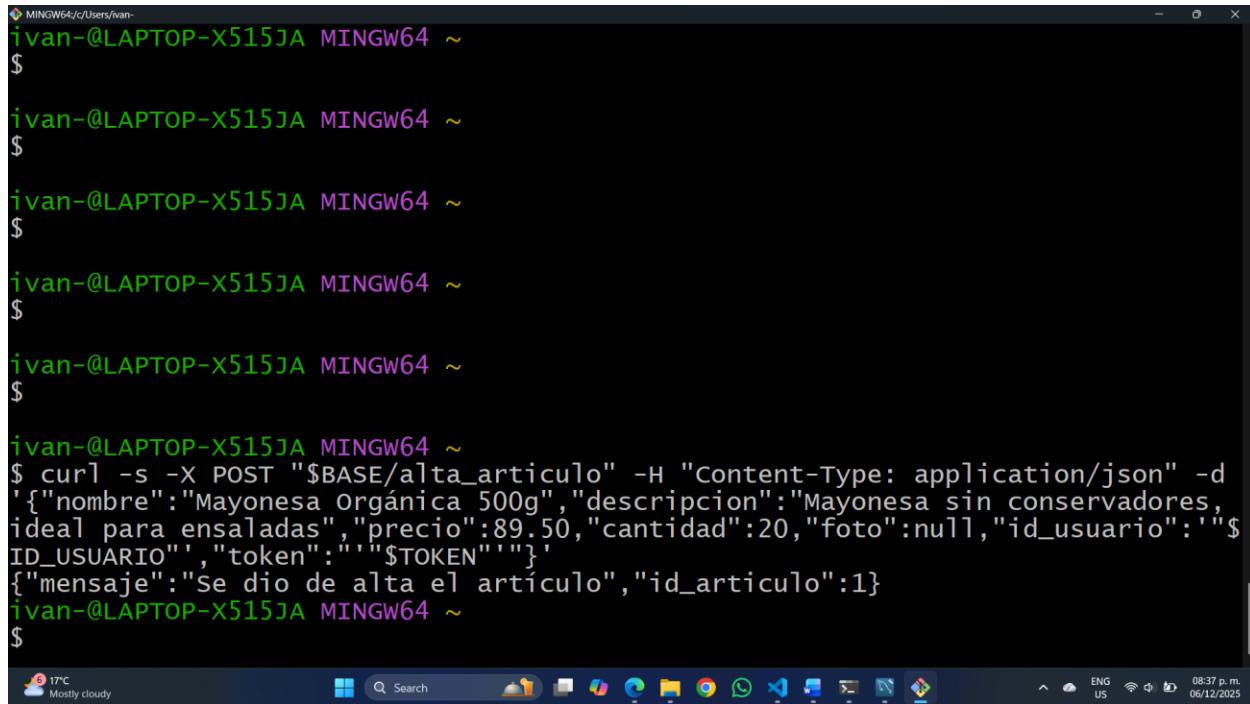


Figura 14. Prueba de login y actualización del token del usuario.

6.7 alta_articulo (captura de stock)

Se realizó la función `alta_articulo` para insertar artículos en la tabla `stock` con nombre, descripción, precio y cantidad; si se adjuntó foto, se insertó en `fotos_articulos` usando el `id_articulo` recién creado. Se verificó el acceso mediante `verifica_acceso(id_usuario, token)` y se utilizó transacción al incluir foto.

- Se accedió a parámetros enviados por el front-end (incluida foto en base64).
- Se manejó `INSERT` en `stock` y opcional `INSERT` en `fotos_articulos`.
- Se devolvió 200 en éxito y 400 en errores de validación o acceso.



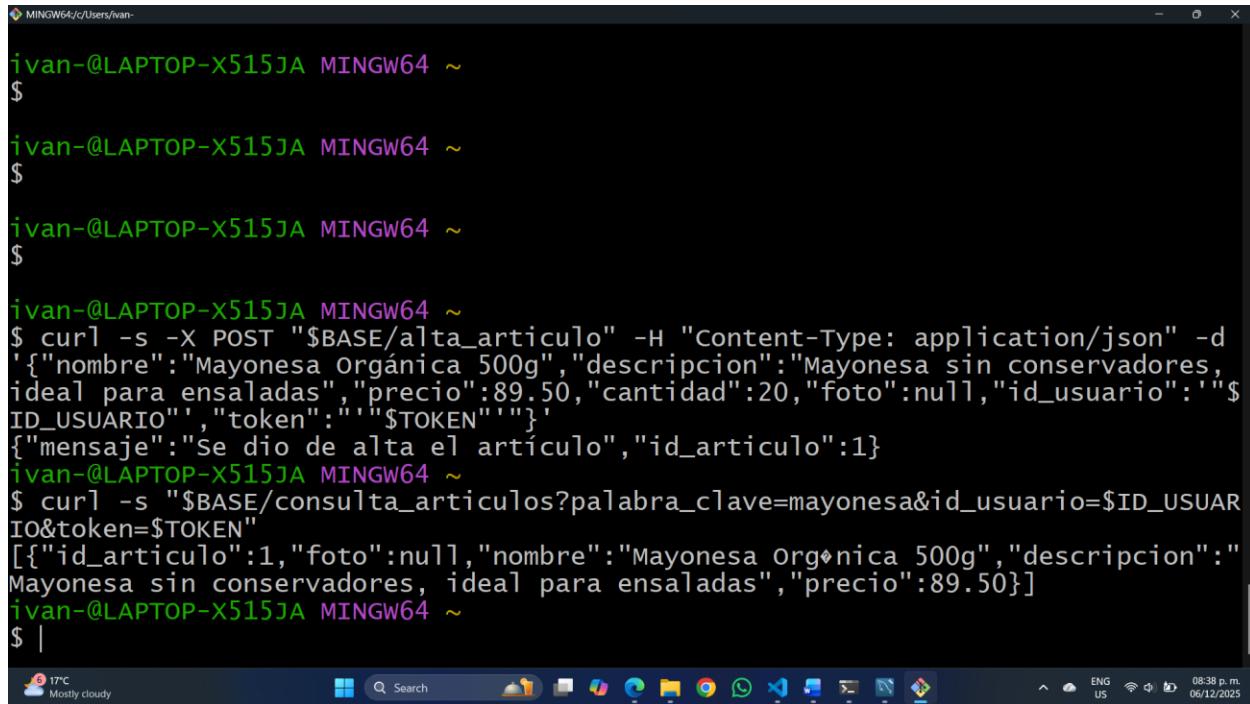
```
MINGW64/c/Users/ivan-
ivan-@LAPTOP-X515JA MINGW64 ~
$ 
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$BASE/alta_articulo" -H "Content-Type: application/json" -d '{"nombre":"Mayonesa Orgánica 500g","descripcion":"Mayonesa sin conservadores, ideal para ensaladas","precio":89.50,"cantidad":20,"foto":null,"id_usuario":"'${ID_USUARIO}'","token":"'${TOKEN}'"}' 
{"mensaje":"Se dio de alta el artículo","id_articulo":1}
ivan-@LAPTOP-X515JA MINGW64 ~
$
```

Figura 15. Inserción de artículo en stock y foto asociada con transacción.

6.8 consulta_articulos (búsqueda por palabra clave)

Se realizó la función consulta_articulos que, con acceso verificado, ejecutó una consulta SELECT con LIKE en nombre y descripcion para la palabra clave indicada. Se devolvió un arreglo de artículos con id_articulo, foto pequeña (si existe), nombre, descripción y precio.

- Se accedió con id_usuario y token para autorización.
- Se utilizó LEFT JOIN con fotos_articulos para obtener imagen opcional.
- Se devolvió un JSON arreglo listo para el renderizado en el front-end.



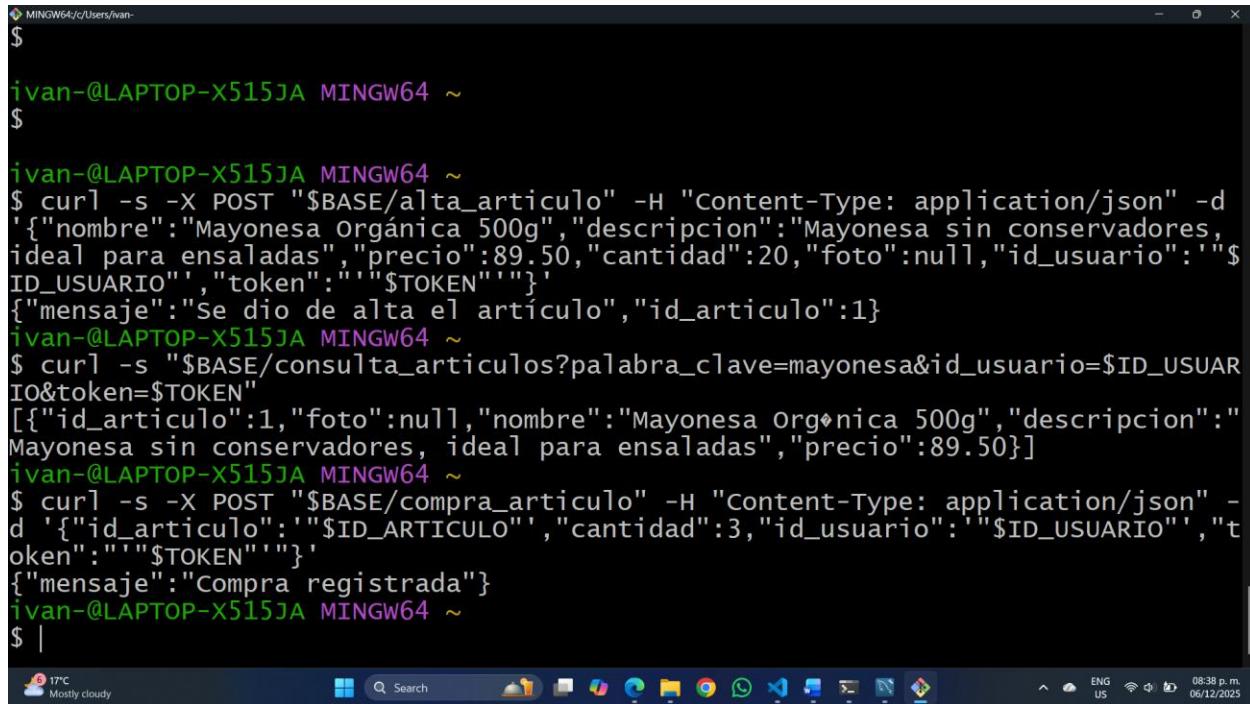
```
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$BASE/alta_articulo" -H "Content-Type: application/json" -d
'{"nombre":"Mayonesa Orgánica 500g","descripcion":"Mayonesa sin conservadores, ideal para ensaladas","precio":89.50,"cantidad":20,"foto":null,"id_usuario":"'${ID_USUARIO}'","token":"'${TOKEN}'"}'
{"mensaje":"Se dio de alta el artículo","id_articulo":1}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s "$BASE/consulta_articulos?palabra_clave=mayonesa&id_usuario=${ID_USUARIO}&token=${TOKEN}"
[{"id_articulo":1,"foto":null,"nombre":"Mayonesa Orgánica 500g","descripcion":"Mayonesa sin conservadores, ideal para ensaladas","precio":89.50}]
ivan-@LAPTOP-X515JA MINGW64 ~
$ |
```

Figura 16. Resultado de consulta_articulos con diferentes palabras clave.

6.9 compra_articulo (compra con transacción)

Se realizó la función compra_articulo que valida si la cantidad solicitada está disponible en stock. Si no hay suficientes artículos, se devolvió 400 con el mensaje “No hay suficientes artículos en stock”. Si alcanza, se ejecutó una transacción que inserta o actualiza el registro en carrito_compra (respetando el índice único (id_usuario, id_articulo)), y actualiza stock restando la cantidad.

- Se accedió con id_usuario, id_articulo, cantidad, token.
- Se realizó SELECT de stock.cantidad, INSERT/UPDATE en carrito_compra, UPDATE en stock.
- Se aseguró commit/rollback ante errores.



```
MINGW64/c/Users/ivan-  
$  
ivan-@LAPTOP-X515JA MINGW64 ~  
$  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ curl -s -X POST "$BASE/alta_articulo" -H "Content-Type: application/json" -d  
'{"nombre":"Mayonesa Orgánica 500g","descripcion":"Mayonesa sin conservadores,  
ideal para ensaladas","precio":89.50,"cantidad":20,"foto":null,"id_usuario":"'${ID_USUARIO}'","token":"'${TOKEN}'"}'  
{"mensaje":"se dio de alta el artículo","id_articulo":1}  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ curl -s "$BASE/consulta_articulos?palabra_clave=mayonesa&id_usuario=${ID_USUARIO}&token=${TOKEN}"  
[{"id_articulo":1,"foto":null,"nombre":"Mayonesa Orgánica 500g","descripcion":"Mayonesa sin conservadores, ideal para ensaladas","precio":89.50}]  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ curl -s -X POST "$BASE/compra_articulo" -H "Content-Type: application/json" -d  
'{"id_articulo":"'${ID_ARTICULO}'","cantidad":3,"id_usuario":"'${ID_USUARIO}'","token":"'${TOKEN}'"}'  
{"mensaje":"Compra registrada"}  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ |
```

Figura 17. Prueba de compra exitosa y manejo del error por stock insuficiente.

6.10 elimina_articulo_carrito_compra (devolver al stock y borrar del carrito)

Se realizó la función `elimina_articulo_carrito_compra` para remover un artículo del carrito de un usuario. Se ejecutó una transacción que primero lee la cantidad en el carrito, actualiza stock sumando esa cantidad y posteriormente borra el registro del carrito.

- Se accedió con `id_usuario`, `id_articulo`, `token`.
- Se realizó `SELECT` de cantidad en carrito; `UPDATE` en stock (`+cantidad`); `DELETE` en `carrito_compra`.
- Se devolvió 200 al completar la transacción.

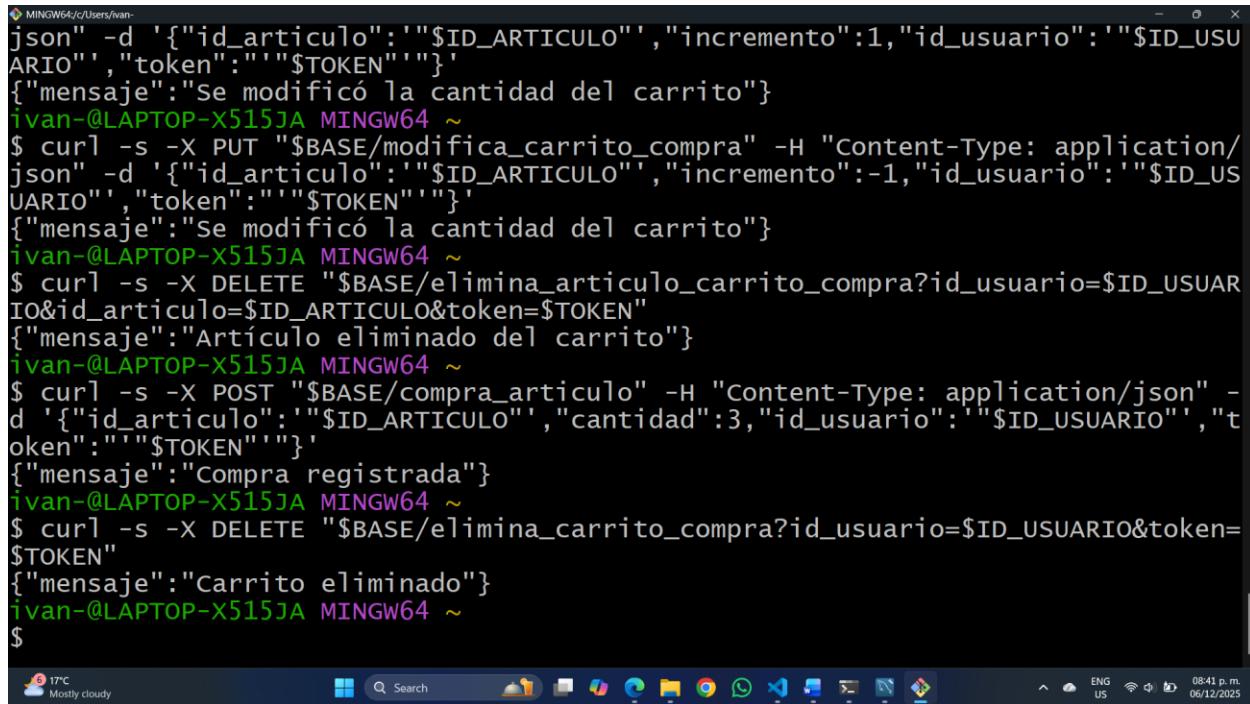
```
MINGW64/c/Users/ivan-[{"id_articulo":1,"foto":null,"nombre":"Mayonesa Orgánica 500g","descripcion":"Mayonesa sin conservadores, ideal para ensaladas","precio":89.50}]  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ curl -s -X POST "$BASE/compra_articulo" -H "Content-Type: application/json" -d '{"id_articulo":"'${ID_ARTICULO}'","cantidad":3,"id_usuario":"'${ID_USUARIO}'","token":"'${TOKEN}'"}'  
{"mensaje":"Compra registrada"}  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ curl -s -X PUT "$BASE/modifica_carrito_compra" -H "Content-Type: application/json" -d '{"id_articulo":"'${ID_ARTICULO}'","incremento":1,"id_usuario":"'${ID_USUARIO}'","token":"'${TOKEN}'"}'  
{"mensaje":"se modificó la cantidad del carrito"}  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ curl -s -X PUT "$BASE/modifica_carrito_compra" -H "Content-Type: application/json" -d '{"id_articulo":"'${ID_ARTICULO}'","incremento":-1,"id_usuario":"'${ID_USUARIO}'","token":"'${TOKEN}'"}'  
{"mensaje":"Se modificó la cantidad del carrito"}  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ curl -s -X DELETE "$BASE/elimina_articulo_carrito_compra?id_usuario=${ID_USUARIO}&id_articulo=${ID_ARTICULO}&token=${TOKEN}"  
{"mensaje":"Artículo eliminado del carrito"}  
ivan-@LAPTOP-X515JA MINGW64 ~  
$
```

Figura 18. Evidencia de eliminación de artículo del carrito y restauración del stock.

6.11 elimina_carrito_compra (borrado total del carrito)

Se realizó la función `elimina_carrito_compra` para vaciar el carrito del usuario. Dentro de una transacción, se sumaron las cantidades de cada artículo de vuelta al stock y se borraron todos los registros del carrito del usuario.

- Se accedió con `id_usuario` y `token`.
- Se ejecutó `SELECT` de todos los artículos del carrito, `UPDATE` acumulado del stock y `DELETE` general del carrito para el usuario.
- Se garantizó integridad de datos con `commit/rollback`.



```
MINGW64/c/Users/ivan-
json" -d '{"id_articulo":"'ID_ARTICULO','incremento":1,"id_usuario":"'ID_USUARIO','token':'$TOKEN"}'
{"mensaje":"Se modificó la cantidad del carrito"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X PUT "$BASE/modifica_carrito_compra" -H "Content-Type: application/json" -d '{"id_articulo":"'ID_ARTICULO','incremento":1,"id_usuario":"'ID_USUARIO','token':'$TOKEN"}'
{"mensaje":"Se modificó la cantidad del carrito"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X DELETE "$BASE/elimina_articulo_carrito_compra?id_usuario=$ID_USUARIO&id_articulo=$ID_ARTICULO&token=$TOKEN"
{"mensaje":"Artículo eliminado del carrito"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$BASE/compra_articulo" -H "Content-Type: application/json" -d '{"id_articulo":"'ID_ARTICULO','cantidad':3,"id_usuario":"'ID_USUARIO','token':'$TOKEN"}'
{"mensaje":"Compra registrada"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X DELETE "$BASE/elimina_carrito_compra?id_usuario=$ID_USUARIO&token=$TOKEN"
{"mensaje":"Carrito eliminado"}
ivan-@LAPTOP-X515JA MINGW64 ~
$
```

Figura 19. Eliminar carrito completo y verificación del stock recuperado.

6.12 modifica_carrito_compra (ajuste de cantidad +1/-1 con validaciones)

Se realizó la función modifica_carrito_compra que permite incrementar o decrementar la cantidad de compra. Si se incrementa y no hay stock suficiente, se devolvió 400 con “No hay suficientes artículos en stock”. Si se decrementa y ya no hay más artículos, se devolvió 400 con “No hay más artículos en el carrito”. Se actualizó el carrito y el stock en una transacción.

- Se accedió con id_articulo, incremento(+1/-1), id_usuario, token.
- Se validó existencia en carrito y disponibilidad en stock.
- Se ejecutaron UPDATE en carrito_compra y en stock coherentemente.

```
MINGW64/c/Users/ivan-  
{"mensaje":"se dio de alta el artículo","id_articulo":1}  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ curl -s "$BASE/consulta_articulos?palabra_clave=mayonesa&id_usuario=$ID_USUARIO&token=$TOKEN"  
[{"id_articulo":1,"foto":null,"nombre":"Mayonesa Orgánica 500g","descripcion":"Mayonesa sin conservadores, ideal para ensaladas","precio":89.50}]  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ curl -s -X POST "$BASE/compra_articulo" -H "Content-Type: application/json" -d '{"id_articulo":"$ID_ARTICULO","cantidad":3,"id_usuario":"$ID_USUARIO","token":"$TOKEN"}'  
{"mensaje":"Compra registrada"}  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ curl -s -X PUT "$BASE/modifica_carrito_compra" -H "Content-Type: application/json" -d '{"id_articulo":"$ID_ARTICULO","incremento":1,"id_usuario":"$ID_USUARIO","token":"$TOKEN"}'  
{"mensaje":"Se modificó la cantidad del carrito"}  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ curl -s -X PUT "$BASE/modifica_carrito_compra" -H "Content-Type: application/json" -d '{"id_articulo":"$ID_ARTICULO","incremento":-1,"id_usuario":"$ID_USUARIO","token":"$TOKEN"}'  
{"mensaje":"Se modificó la cantidad del carrito"}  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ |
```

Figura 20. Ajustes de cantidad en carrito (+/-) y respuestas de validación.

7 Front-end: Aplicación web

Se realizó la extensión del front-end para soportar la captura de artículos, la compra y la administración del carrito, manteniendo compatibilidad móvil y sirviendo los archivos estáticos mediante la función Get. Se accedió a las funciones del back-end con WSClient.js usando métodos GET/POST/PUT/DELETE y se aseguró que el flujo completo opere con token e id_usuario obtenidos tras el login.

7.1 Estructura de archivos y carga del front-end

Se organizó el front-end en la carpeta front-end con los archivos prueba.html, WSClient.js y usuario_sin_foto.png. Se instaló la referencia del script en prueba.html por medio de la función Get:

```
<script src='/api/Get?nombre=/WSClient.js'></script>.
```

Se accedió al front-end con la URL local

<http://localhost:7071/api/Get?nombre=/prueba.html> y, posteriormente, con la URL de la Function App tras el despliegue. Se validó la meta viewport para uso móvil y la disponibilidad de imágenes por defecto.

- Se realizó la verificación de carga de prueba.html en navegador de escritorio y móvil.
- Se accedió a WSClient.js vía Get y se comprobó que las rutas estén bajo /api.

Figura 22. Captura de pantalla de la estructura del front-end y carga de prueba.html utilizando la función Get.

7.2 Menú principal extendido

Se realizó la extensión del menú principal para incluir las opciones “Captura de artículo” y “Compra de artículos”. Se accedió a estas vistas tras el login, reutilizando las variables globales email, token y recuperando id_usuario desde el endpoint consulta_usuario después de iniciar sesión.

- Se realizó la integración de los nuevos botones en prueba.html.

- Se accedió correctamente a cada opción y se validó la visibilidad/ocultamiento de secciones.

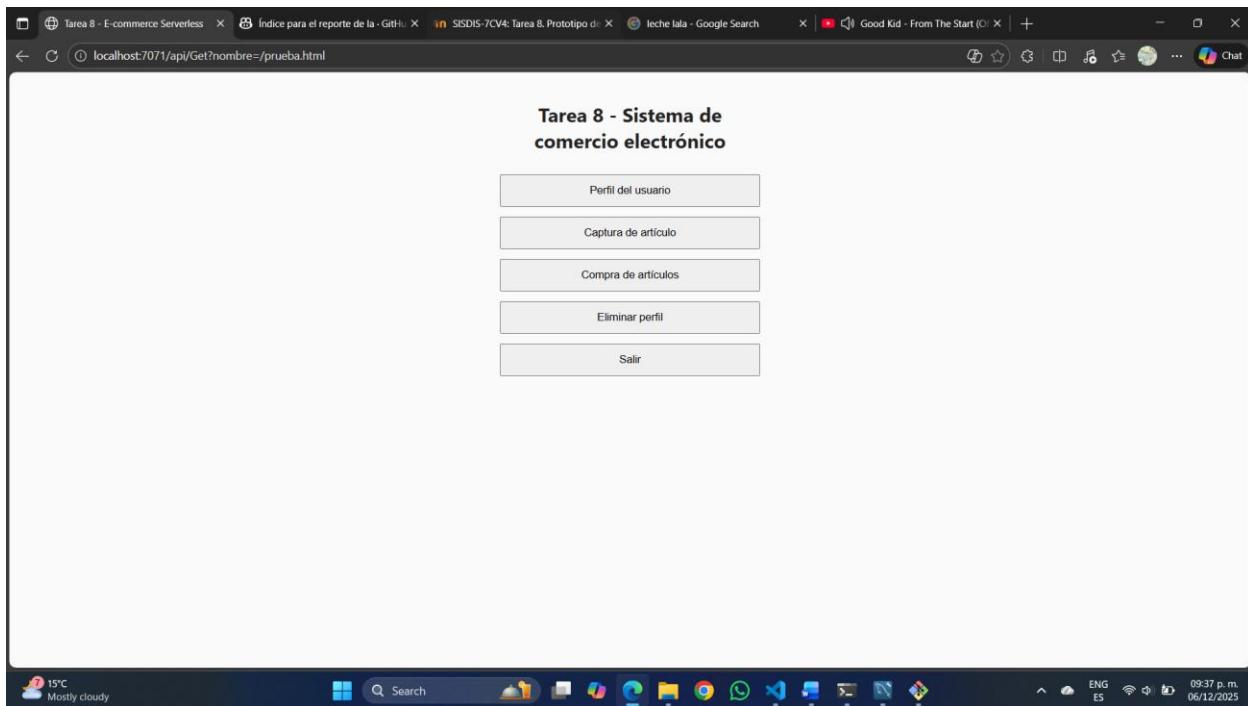


Figura 23. Captura del menú con las nuevas opciones visibles tras iniciar sesión.

7.3 Pantalla “Captura de artículo”

Se realizó la pantalla para alta de artículos que incluye campos: nombre, descripción, precio, cantidad y fotografía. Se reutilizó la función `readSingleFile` para convertir la imagen a base64 y se invocó el endpoint `alta_articulo` mediante `cliente.post("alta_articulo", {...})` con `id_usuario` y `token`.

- Se realizó la validación mínima: campos obligatorios y formato de precio/cantidad.
- Se accedió al servidor y se obtuvo respuesta de éxito o error con mensajes JSON.

Tabla de campos utilizados:

Campo	Tipo	Descripción
nombre	texto	Nombre del artículo

descripcion	texto	Descripción del artículo
precio	número/decimal	Precio unitario
cantidad	número entero	Existencia inicial
foto	base64 (opcional)	Imagen del artículo

Tabla 9 Tabla de campos

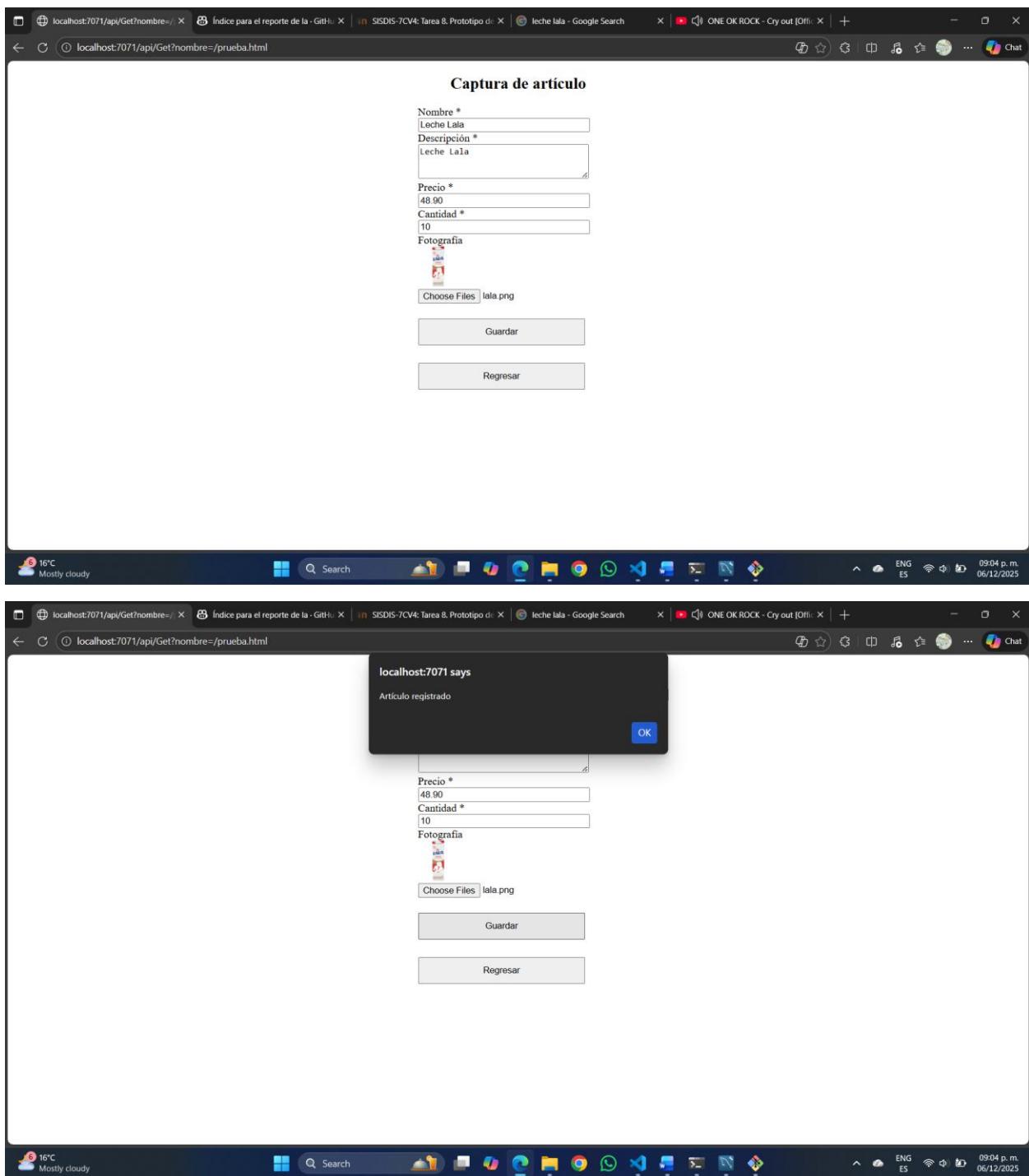


Figura 24. Pantalla de captura de artículo con imagen cargada y respuesta exitosa tras el envío.

7.4 Pantalla “Compra de artículos”

Se realizó la vista de compra en la que se accedió a consulta_articulos con una palabra_clave. Se mostraron resultados con mini-foto, nombre, descripción, precio, un campo de cantidad por cada artículo (default 1), y botones “+” y “-” para ajustar cantidad con el endpoint modifica_carrito_compra. Se instaló el botón “Compra” que invoca compra_articulo.

- Se accedió a la búsqueda con cliente.get("consulta_articulos", { palabra_clave, id_usuario, token }).
- Se realizó el control de cantidad por artículo, verificando las respuestas de validación del back-end.
- Se confirmó que al presionar “Compra” se registra la cantidad en el carrito y se descuenta del stock.

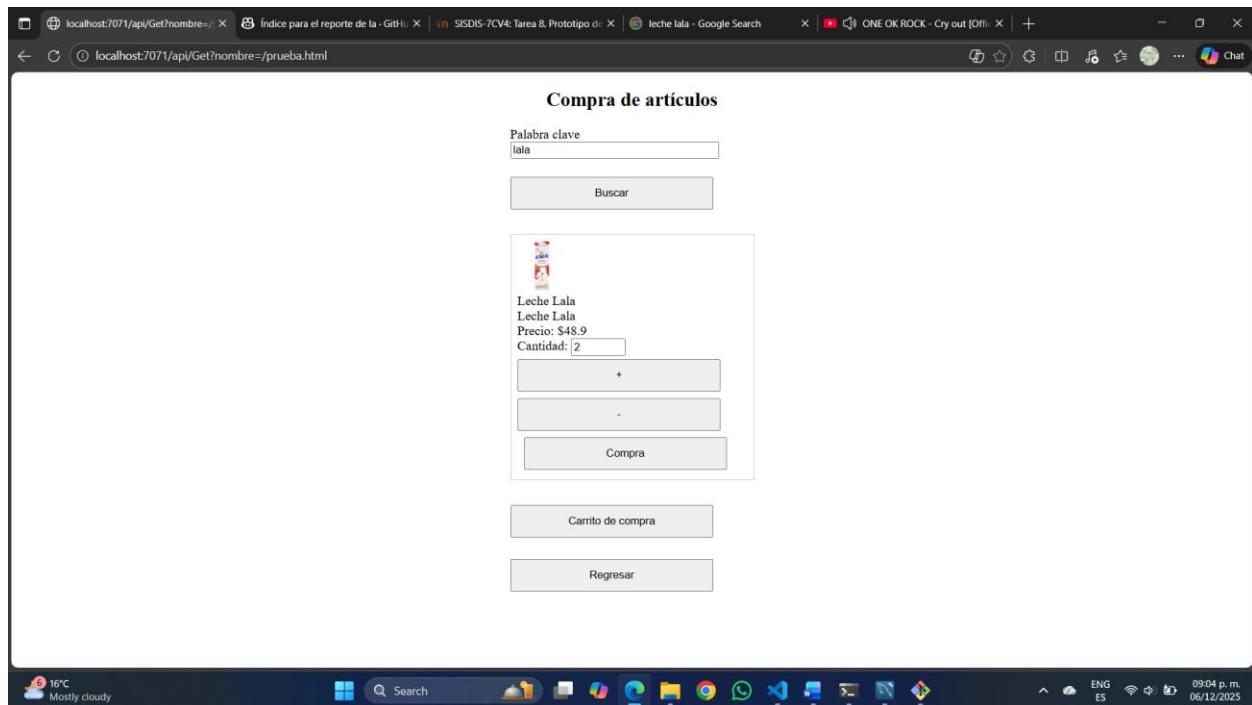


Figura 25. Listado de artículos mostrando imagen, precio, y controles de cantidad con el botón “Compra”.

7.5 Pantalla “Artículos en el carrito”

Se realizó la pantalla de carrito que muestra los artículos agregados: mini-foto, nombre, cantidad, precio y costo (cantidad x precio), además del total. Se instaló el botón “Eliminar artículo” que invoca `elimina_articulo_carrito_compra` y el botón “Eliminar carrito” que invoca `elimina_carrito_compra`. Se accedió al botón “Seguir comprando” para regresar a la vista de compra.

- Se realizó el cálculo de costos por artículo y total de la compra en el cliente.
- Se accedió a las operaciones de eliminación y se verificó la restauración de stock en las respuestas del servidor.

Tabla de acciones:

Acción	Endpoint	Método	Efecto
Ver carrito	(consulta local/refresh)	GET	Mostrar artículos, cantidades y total
Eliminar artículo	<code>elimina_articulo_carrito_compra</code>	DELETE	Sumar cantidad a stock y borrar del carrito
Eliminar carrito	<code>elimina_carrito_compra</code>	DELETE	Vaciar carrito y restaurar stock
Seguir comprando	Navegación	-	Regresar a búsqueda y lista de artículos

Tabla 10 Tabla de acciones

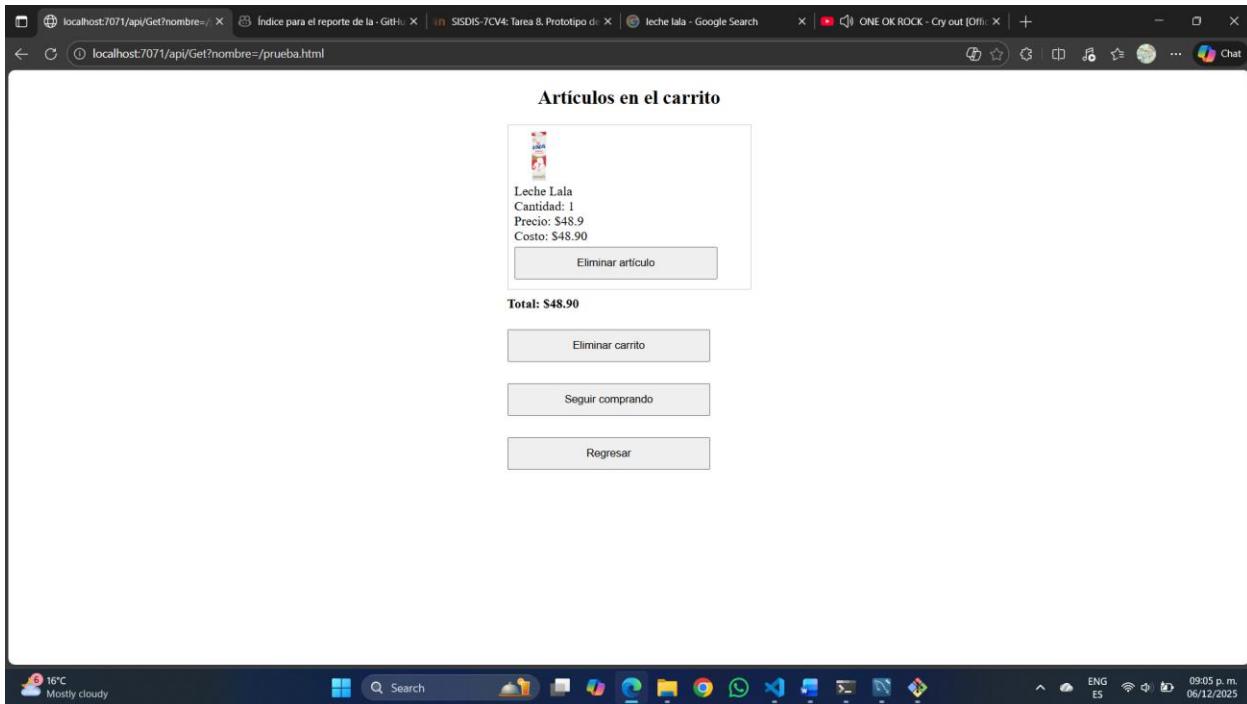


Figura 26. Vista del carrito mostrando costos por artículo, total, y botones de eliminación y navegación.

7.6 Integración con WSClient.js

Se realizó la integración de llamadas HTTP usando WSClient.js, aprovechando:

- `get(opcion, args, callback)` para consultas y login.
- `post(opcion, body, callback)` para altas, compras y registro.
- `put(opcion, args, body, callback)` para modificaciones.
- `delete(opcion, args, callback)` para eliminaciones.

Se accedió a los endpoints por medio de rutas `/api/...` y se validó la codificación de parámetros (uso de `encodeURI` y reemplazos de caracteres), además del manejo de Content-Type adecuado según método. Se realizaron pruebas de error y éxito, mostrando alertas con el contenido de los mensajes devueltos.

- Se realizó la verificación de callbacks y códigos HTTP.
- Se accedió a la consola del navegador para validar requests/responses.

8 Despliegue en Azure

Se realizó el despliegue del prototipo en Azure for Students, manteniendo la región Canada y la nomenclatura exigida por la tarea. Se instaló la base de datos en MySQL PaaS, se accedió a una cuenta de almacenamiento de Azure Files para hospedar el front-end y se publicó la Azure Function App configurando variables de entorno y el montaje de archivos. Se validó el acceso final mediante la URL pública del endpoint Get.

8.1 Creación de Azure Database for MySQL (t8mysql2022630278) e inicialización de esquema

Se realizó la creación de la instancia MySQL Flexible Server en Azure, con IP pública habilitada y reglas de firewall para acceso controlado desde la Function App y, temporalmente, desde la estación local. Se accedió al servidor con el usuario remoto (sin @localhost) y se ejecutó el esquema de la base de datos “servicio_web”, incluyendo las tablas iniciales y las tablas adicionales del e-commerce.

- Se instaló la instancia MySQL en la región Canada.
- Se accedió mediante MySQL Workbench/CLI con el usuario remoto y contraseña configurada.
- Se realizó la ejecución del script de creación de tablas: usuarios, fotos_usuarios, stock, fotos_articulos, carrito_compra e índice único en (id_usuario, id_articulo).

Tabla de verificación del esquema:

Recurso	Estado	Observaciones
Base de datos servicio_web	Creada	Acceso con usuario remoto
Tabla usuarios	OK	Índice único en email
Tabla fotos_usuarios	OK	FK hacia usuarios(id_usuario)
Tabla stock	OK	PK id_articulo AUTO_INCREMENT
Tabla fotos_articulos	OK	FK hacia stock(id_articulo)
Tabla carrito_compra	OK	UNIQUE (id_usuario, id_articulo)

Tabla 11 Tabla de verificación del esquema

Basics Networking Additional configuration Tags Review + create

Create an Azure Database for MySQL flexible server. [Learn more](#)

Did you know? Did you know that new users in Azure can use MySQL - Flexible Server free for up to 750 hours using Azure free account? [Learn more](#)

Project details
Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription: Azure for Students
Resource group: t8-2022630278
Create new

Server details
Enter required settings for this server, including picking a location and configuring the compute and storage resources.

Estimated costs

	USD 275.94/month
Compute	Standard_D4ads_v5 (4 vCores, 68.98 per vCore) 68.98
Storage	64 GiB (USD 0.13 per GiB) 64 x 0.13
Auto scale IOPS	Auto scale IOPS is billed on usage in per million request increments. Learn more
High availability	Same zone or Zone redundant high availability
Backup Retention	Backup retention is billed based on additional storage used for retaining backups. Learn more

[Review + create](#) [Next : Networking >](#)

Basics Networking Additional configuration Tags Review + create

Create an Azure Database for MySQL flexible server. [Learn more](#)

Did you know? Did you know that new users in Azure can use MySQL - Flexible Server free for up to 750 hours using Azure free account? [Learn more](#)

Flexible server Microsoft

Server name: t8mysql[2022630278]

Region: Canada Central

MySQL version: 8.0

Workload type: Dev/Test

Compute + storage: Burstable, B1ms
1 vCores, 2 GiB RAM, 20 GiB storage, Auto scale IOPS
Geo-redundancy: Disabled
[Configure server](#)

Availability zone: No preference

High availability
High availability provides additional server resilience in the event of a failure. You can also specify high availability options in 'Compute + storage'.

Estimated costs

	USD 13.50/month
Compute	Free upto 750 hours Standard_B1ms (1 vCore) 13.50
Storage	Free upto 12 GB 20 GiB (USD 0.13 per GiB) 20 x 0.13
Auto scale IOPS	Auto scale IOPS is billed on usage in per million request increments. Learn more
Backup Retention	Backup retention is billed based on additional storage used for retaining backups. Learn more

[Review + create](#) [Next : Networking >](#)

Flexible server

Server names, networking connectivity method, zone redundant HA and backup redundancy cannot be changed after server is created. Review these options carefully before provisioning.

Authentication

Select the authentication method you would like to support for accessing this MySQL server. Enabling MySQL password authentication allows you to authenticate with user names and passwords that are stored inside MySQL.

Enabling Microsoft Entra authentication allows you to create user names in MySQL, which are mapped to accounts stored in Microsoft Entra ID. Users or applications authenticated against Microsoft Entra ID, can retrieve tokens that are presented to MySQL as their corresponding time-limited password. [Learn more](#)

Authentication method

- MySQL authentication only
- Microsoft Entra authentication only
- MySQL and Microsoft Entra authentication

Administrator login *

Password *

Confirm password *

Auto scale IOPS

Auto scale IOPS is billed on usage in per million request increments. [Learn more](#)

Backup Retention

Backup retention is billed based on additional storage used for retaining backups. [Learn more](#)

Bandwidth

Outbound data transfer across services in different regions will incur additional charges. Any inbound data transfer is free. [Learn more](#)

Review + create **Next : Networking >**

Networking

Configure networking access and security for your server.

Network connectivity

You can connect to your server by specifying a public IP address, creating private endpoints or from within a selected virtual network.

Connectivity method

- Public access (allowed IP addresses) and Private endpoint
- Private access (VNet Integration)

Database port *

Public access

Allow public access to this resource through the internet using a public IP address

Estimated costs

	USD 13.50/month
Compute	USD 13.50/month
Free quota 750 hours	
Standard_B1ms (1 vCore)	13.50
Storage	USD 2.54/month
Free quota 12 GB	
20 GiB (USD 0.13 per GiB)	20 x 0.13
Auto scale IOPS	
Auto scale IOPS is billed on usage in per million request increments. Learn more	
Backup Retention	
Backup retention is billed based on additional storage used for retaining backups. Learn more	

Review + create **< Previous** **Next : Additional configuration >**

Estimated costs

Compute	USD 13.50/month	
Standard_B1ms (1 vCore)	13.50	
Storage	USD 2.54/month	
Free upto 32 GB		
20 GB (USD 0.13 per GiB)	20 x 0.13	
Auto scale IOPS		
Auto scale IOPS is billed on usage in per million request increments. Learn more		
Backup Retention		
Backup retention is billed based on additional storage used for retaining backups. Learn more		
Bandwidth		

Estimated total

Standard_B1ms (1 vCore)	13.50	
Storage	USD 2.54/month	
Free upto 32 GB		
20 GB (USD 0.13 per GiB)	20 x 0.13	
Auto scale IOPS		
Auto scale IOPS is billed on usage in per million request increments. Learn more		
Backup Retention		
Backup retention is billed based on additional storage used for retaining backups. Learn more		
Bandwidth		
Outbound data transfer across services in different regions will incur additional charges. Any inbound data transfer is free. Learn more		
Estimated total	USD 16.04/month	
Charges will apply if you use above the free monthly limits. Please check your usage		

The screenshot shows the Microsoft Azure portal interface. The main page displays the overview of an Azure Database for MySQL flexible server named **t8mysql[2022630278]**. The left sidebar contains navigation links such as Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Learning center, Resource visualizer, Settings, Compute + storage, Networking, Databases, Connect, Server parameters, Replication, Maintenance, High availability, and Backup and restore. The central pane shows the server's configuration details under the **Essentials** section, including Subscription (move) to **Azure for Students**, Subscription ID **fd58a3da-fcef-47d1-ac0e-5b891faa4251**, Resource group **t8-2022630278**, Status **Ready**, Location **Canada Central**, Endpoint **t8mysql[2022630278].mysql.database.azure.com**, Administrator login, Configuration **Burstable, 81ms, 1 vCores, 2 GiB RAM, 20 storage, 360...**, MySQL version **8.0 Upgrade**, Availability zone **2**, and Created on **2025-12-07 04:05:07.4144543 UTC**. Below this, there are tabs for Getting started, Properties, Recommendations, Monitoring, and Tutorials. A message at the bottom says "We've prepared a checklist to get you started". On the right side, a **Notifications** panel is open, showing a single event: **Deployment succeeded** with the message "Deployment 'MySQLflexibleServer_e9d20ba206624881914b-aa49c594994b' to resource group 't8-2022630278' was successful." with a timestamp of "a few seconds ago". The bottom right corner shows the system status bar with the date "06/12/2025", time "10:08 p.m.", and language "ENG ES".

Pre-requisites check

The most common connection methods have one or more of the requirements listed below.

- Firewall rules are enabled on this server.
- Any resources that are part of the same virtual network as the private endpoint can access the server.
- SSL is enforced and TLS version is 1.2.
- Server is in Ready state.

Connection details

hostname=t8mysql2022630278.mysql.database.azure.com
port=3306
username=x
password=[your-password]
ssl-mode=require

MySQL Workbench

```
C:\Users\ivan->winget install Oracle.MySQLShell
Encontrado MySQL Shell [Oracle.MySQLShell] Versión 8.4.5
El propietario de esta aplicación le concede una licencia.
Microsoft no es responsable, ni tampoco concede ninguna licencia de paquetes de terceros.
Este paquete requiere las siguientes dependencias:
- Paquetes
  Microsoft.VCRedist.2015+.x64
Descargando https://cdn.mysql.com/Downloads/MySQL-Shell/mysql-shell-8.4.5-windows-x86-64bit.msi
[Progress Bar] 125 MB / 125 MB
El hash del instalador se verificó correctamente
Iniciando instalación de paquete...
Instalado correctamente

C:\Users\ivan->mysql -h t8mysql2022630278.mysql.database.azure.com -P 3306 -u x -p
'mysql' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\ivan->mysqlsh --sql -h t8mysql2022630278.mysql.database.azure.com -P 3306 -u TU_USUARIO -p
Please provide the password for 'TU_USUARIO@t8mysql2022630278.mysql.database.azure.com:3306':
MySQL Shell 8.4.5

Copyright (c) 2016, 2025, Oracle and/or its affiliates.
Oracle is a registered trademark of Oracle Corporation and/or its affiliates.
Other names may be trademarks of their respective owners.

Type '\help' or '\?' for help; '\quit' to exit.
Creating a session to 'TU_USUARIO@t8mysql2022630278.mysql.database.azure.com:3306'
MySQL Error 1045 (28000): Access denied for user 'TU_USUARIO'@'201.162.226.149' (using password: NO)
```

```
Command Prompt - mysqlsh x + v
Creating a session to 'TU_USUARIO@t8mysql2022630278.mysql.database.azure.com:3306'
MySQL Error 1045 (28000): Access denied for user 'TU_USUARIO'@'201.162.226.149' (using pa
ssword: NO)

C:\Users\ivan->mysqlsh --sql -h t8mysql2022630278.mysql.database.azure.com -P 3306 -u x -
p
Please provide the password for 'x@t8mysql2022630278.mysql.database.azure.com:3306': ****
*****
Save password for 'x@t8mysql2022630278.mysql.database.azure.com:3306'? [Y]es/[N]o/Ne[v]er
(default No): y
MySQL Shell 8.4.5

Copyright (c) 2016, 2025, Oracle and/or its affiliates.
Oracle is a registered trademark of Oracle Corporation and/or its affiliates.
Other names may be trademarks of their respective owners.

Type '\help' or '\?' for help; '\quit' to exit.
Creating a session to 'x@t8mysql2022630278.mysql.database.azure.com:3306'
Fetching global names for auto-completion... Press ^C to stop.
Your MySQL connection id is 15
Server version: 8.0.42-azure Source distribution
No default schema selected; type \use <schema> to set one.
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl SQL > |
```

```
Command Prompt - mysqlsh x + v
Type '\help' or '\?' for help; '\quit' to exit.
Creating a session to 'x@t8mysql2022630278.mysql.database.azure.com:3306'
Fetching global names for auto-completion... Press ^C to stop.
Your MySQL connection id is 15
Server version: 8.0.42-azure Source distribution
No default schema selected; type \use <schema> to set one.
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl SQL > create database servicio_web;
create database servicio_web;
Query OK, 1 row affected (0.1204 sec)
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl SQL > use servicio_web;
Default schema set to `servicio_web`.
Fetching global names, object names from `servicio_web` for auto-completion... Press ^C to stop.
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl SQL > show databases;
+-----+
| Database      |
+-----+
| information_schema |
| mysql          |
| performance_schema |
| servicio_web   |
| sys            |
+-----+
5 rows in set (0.1057 sec)
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servici... SQL > |
```

```
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio... SQL > create table usuarios
-> (
->     id_usuario integer auto_increment primary key,
->     password varchar(64) not null,
->     token varchar(20),
->     email varchar(100) not null,
->     nombre varchar(100) not null,
->     apellido_paterno varchar(100) not null,
->     apellido_materno varchar(100),
->     fecha_nacimiento datetime not null,
->     telefono bigint,
->     genero char(1)
-> );
Query OK, 0 rows affected (0.2504 sec)
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio... SQL > create table fotos_usuarios
-> (
->     id_foto integer auto_increment primary key,
->     foto longblob,
->     id_usuario integer not null
-> );
Query OK, 0 rows affected (0.1810 sec)
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio... SQL > alter table fotos_usuarios add foreign key (id_usuario) references usuarios(id_usuario);
Query OK, 0 rows affected (0.2854 sec)

Records: 0 Duplicates: 0 Warnings: 0

```

```
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio... SQL > create unique index usuarios_1 on usuarios(email);
Query OK, 0 rows affected (0.1923 sec)

Records: 0 Duplicates: 0 Warnings: 0
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio... SQL > create table stock
-> (
->     id_articulo integer auto_increment primary key,
->     nombre varchar(200) not null,
->     descripcion text,
->     precio decimal(10,2) not null,
->     cantidad integer not null
-> );
Query OK, 0 rows affected (0.1732 sec)
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio_web SQL >
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio_web SQL > create table fotos_articulos
-> (
->     id_foto integer auto_increment primary key,
->     foto longblob,
->     id_articulo integer not null,
->     foreign key (id_articulo) references stock(id_articulo)
-> );
Query OK, 0 rows affected (0.1754 sec)
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio_web SQL >
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio_web SQL > create table carrito_compra
-> (
->     id_usuario integer not null,
->     id_articulo integer not null,
->     cantidad integer not null
-> );

```

```

Command Prompt - mysqlsh
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio_web SQL > create table carrito_compra
-> (
->     id_usuario integer not null,
->     id_articulo integer not null,
->     cantidad integer not null,
->     unique key carrito_unico (id_usu
ario, id_articulo)
-> );
Query OK, 0 rows affected (0.1851 sec)
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio_web SQL > create user 'a' identified by '';
Query OK, 0 rows affected (0.1275 sec)
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio_web SQL > grant all on servicio_web.* to 'a';
Query OK, 0 rows affected (0.1256 sec)
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio_web SQL > SELECT user, host, account_locked, p
assword_expired, plugin
-> FROM mysql.user
-> ORDER BY host, user;
+-----+-----+-----+-----+-----+
| user | host | account_locked | password_expired | plugin |
+-----+-----+-----+-----+-----+
| a    | %   | N      | N      | mysql_native_password |
| x    | %   | N      | N      | mysql_native_password |
| azure_superuser | 127.0.0.1 | N      | N      | mysql_native_password |
| azure_superuser | localhost | N      | N      | mysql_native_password |
| mysql.infoschema | localhost | Y      | N      | caching_sha2_password |
| mysql.session    | localhost | Y      | N      | caching_sha2_password |
| mysql.sys        | localhost | Y      | N      | caching_sha2_password |
+-----+-----+-----+-----+-----+
7 rows in set (0.1240 sec)
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio_web SQL > |

```

```

Command Prompt - mysqlsh
+-----+-----+-----+-----+-----+
| a    | %   | N      | N      | mysql_native_password |
| x    | %   | N      | N      | mysql_native_password |
| azure_superuser | 127.0.0.1 | N      | N      | mysql_native_password |
| azure_superuser | localhost | N      | N      | mysql_native_password |
| mysql.infoschema | localhost | Y      | N      | caching_sha2_password |
| mysql.session    | localhost | Y      | N      | caching_sha2_password |
| mysql.sys        | localhost | Y      | N      | caching_sha2_password |
+-----+-----+-----+-----+-----+
7 rows in set (0.1240 sec)
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio_web SQL > CREATE TABLE IF NOT EXISTS ordenes (
Y KEY,
->     id_orden INT AUTO_INCREMENT PRIMAR
->     id_usuario INT NOT NULL,
->     fecha DATETIME NOT NULL,
->     total DECIMAL(10,2) NOT NULL
-> );
Query OK, 0 rows affected (0.2099 sec)
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio_web SQL > CREATE TABLE IF NOT EXISTS orden_det
alle (
->     id_orden INT NOT NULL,
->     id_articulo INT NOT NULL,
->     cantidad INT NOT NULL,
->     precio_unitario DECIMAL(10,2) NOT
NULL,
->     subtotal DECIMAL(10,2) NOT NULL,
->     INDEX idx_orden (id_orden),
->     INDEX idx_articulo (id_articulo)
-> );
Query OK, 0 rows affected (0.1855 sec)
MySQL t8mysql2022630278.mysql.database.azure.com:3306 ssl servicio_web SQL > |

```

Figura 29. Creación de MySQL Flexible Server en Azure (Canada) y ejecución del esquema “servicio_web”.

8.2 Creación de Storage Account (t8af2022630278) y File Share (t8rca2022630278)

Se realizó la creación de la cuenta de almacenamiento de Azure Files en la región Canada bajo el nombre t8af2022630278. Se accedió al recurso compartido de archivos t8rca2022630278 donde se hospedó el front-end (HTML, JS, imágenes).

- Se accedió al panel de t8af2022630278 y se creó el File Share t8rca2022630278.

The screenshot shows the 'Create a storage account' wizard in the Microsoft Azure portal. The 'Basics' tab is selected. In the 'Project details' section, the subscription is set to 'Azure for Students' and the resource group is 't8-2022630278'. Under 'Instance details', the storage account name is 't8af2022630278', the region is '(Canada) Canada Central', and the preferred storage type is 'Azure Files'. A note indicates that this helps provide relevant guidance but doesn't restrict storage to this type. At the bottom, there are 'Previous', 'Next', and 'Review + create' buttons.

This screenshot shows the 'Create a storage account' wizard in the Microsoft Azure portal, specifically the 'Instance details' step. The storage account name is 't8af2022630278', the region is '(Canada) Canada Central', and the preferred storage type is 'Azure Files'. A note states that this helps provide relevant guidance. Below these, the 'Performance' section is expanded, showing options for 'Standard' (selected) and 'Premium' storage types. The 'File share billing' section shows 'Pay-as-you-go file shares' (selected) and 'Provisioned v2'. The 'Redundancy' section shows 'Locally-redundant storage (LRS)' selected. At the bottom, there are 'Previous', 'Next', and 'Review + create' buttons.

The screenshot shows the 'Create a storage account' review step in the Azure portal. The 'Review + create' tab is selected. The page displays basic and advanced configuration details:

Basics

Subscription	Azure for Students
Resource group	t8-2022630278
Location	Canada Central
Storage account name	t8af2022630278
Preferred storage type	Azure Files
Performance	Standard
File share billing	Pay-as-you-go file shares
Replication	Locally-redundant storage (LRS)

Advanced

Enable hierarchical namespace	Disabled
Enable SFTP	Disabled
Enable network file system v3	Disabled
Allow cross-tenant replication	Disabled
Access tier	Hot

At the bottom, there are 'Previous', 'Next', and 'Create' buttons. The 'Create' button is highlighted in blue. The status bar at the bottom right shows the date and time: 10:44 p.m. 06/12/2025.

The screenshot shows the 'Create a storage account' review step in the Azure portal, displaying security, networking, and data protection settings:

Security

Secure transfer	Enabled
Blob anonymous access	Disabled
Allow storage account key access	Enabled
Default to Microsoft Entra authorization in the Azure portal	Disabled
Minimum TLS version	Version 1.2
Permitted scope for copy operations (preview)	From any storage account

Networking

Public network access	Enabled
Public network access scope	Enabled from all networks
Default routing tier	Microsoft network routing

Data protection

Point-in-time restore	Disabled
Blob soft delete	Enabled
Blob retention period in days	7
Container soft delete	Enabled

At the bottom, there are 'Previous', 'Next', and 'Create' buttons. The 'Create' button is highlighted in blue. The status bar at the bottom right shows the date and time: 10:44 p.m. 06/12/2025.

The screenshot shows the Microsoft Azure Storage account overview for the resource group 't8-2022630278'. Key details include:

- Resource group (move):** t8-2022630278
- Location:** canadacentral
- Subscription (move):** Azure for Students
- Subscription ID:** fd58a3da-fcef-47d1-ac0e-5b891faa4251
- Disk state:** Available
- Tags:** None

The 'Properties' tab is selected, showing configuration for the Blob service and Networking. The Blob service has 'Hierarchical namespace' disabled, 'Default access tier' set to Hot, and 'Blob soft delete' enabled. Networking settings include 'Public network access' and 'Public network access scope' both set to 'Enabled from all networks'.

The screenshot shows the Microsoft Azure Storage center File shares page for the storage account 't8af2022630278'. The left sidebar shows the navigation path: Home > Storage center | Blob Storage > t8af2022630278 | File shares.

The main area displays the 'File share settings' section, which includes:

- Identity-based access: Not configured
- Default share-level permissions: Disabled
- Soft delete: 7 days
- Maximum capacity: 100 TiB
- Security: Maximum compatibility

A search bar at the top allows searching by prefix (case-sensitive). Below the settings, a table lists file shares, showing one entry: 'You don't have any file shares yet. Click '+ File share' to get started.'

New file share

Basics **Backup** **Review + create**

Name * t8ca2022630278

Access tier * Transaction optimized

Performance

Maximum IO/s 20000

Maximum capacity 100 TiB

To use the SMB protocol with this share, check if you can communicate over port 445. These scripts for Windows client and Linux client can help. Learn how to circumvent port 445 issues.

Review + create < Previous Next : Backup > Give feedback

New file share

Basics **Backup** **Review + create**

Azure Backup protects your file shares from accidental deletion or modification with granular restore and at-scale management capabilities. [Learn more](#)

Enable backup

Recovery Services Vault * Create new Select existing

Vault name * vault-minv8z2lo

Resource group * t8-2022630278

Backup policy * (new) DailyPolicy-minv8z2lf [Edit this policy](#)

Policy details

Backup frequency Daily at 7:30 PM UTC

Retention of daily backup point Retain backup taken every day at 7:30 PM for 30 Day(s)

Review + create < Previous Next : Review > Give feedback

Validation passed

Basics Backup Review + create

Basics

File share name: t8rca2022630278
Access Tier: TransactionOptimized
Protocol: SMB

Backup

Vault name: (new) vault-minv8s2lo
Backup policy: (new) DailyPolicy-minv8s2tf
Policy details: Backup frequency: Daily at 7:30 PM UTC
Retention of daily backup point: Retain backup taken every day at 7:30 PM for 30 Day(s)

Create < Previous Next > Download a template for automation Give feedback

- Se realizó la subida de prueba.html, WSClient.js, usuario_sin_foto.png y recursos adicionales del front-end.
- Se verificó la cuota y permisos del File Share, asegurando disponibilidad.

t8rca2022630278

Overview Connect Upload Refresh Add directory Delete share Change tier Edit quota Give feedback

Storage account: t8rca2022630278 Share URL: https://t8rca2022630278.file.core.windows.net/t8rca2022630278
Resource group: t8-2022630278 Redundancy: Locally-redundant storage (LRS)
Location: Canada Central Configuration modified: 12/6/2025, 10:48:56 PM
Subscription: Azure for Students
Subscription ID: fd58a3da-fcef-47d1-ac0e-5b891faa4251

Properties Capabilities (2) Tutorials

Size
Maximum storage (GiB): 102400
Used storage capacity (GiB): 0
Access tier: Transaction optimized

Performance
IOPS: Varies by region, Learn more
Throughput (MiB/sec): Varies by region, Learn more

Feature status
Soft delete: 14 days
Large file shares: Enabled

Identity-based access
Directory service: Not configured
Domain:

SMB protocol settings
Security profile: Maximum compatibility

Add or remove favorites by pressing Ctrl+Shift+F Give feedback

Figura 30. Azure Storage Account con el File Share y evidencia de archivos del front-end cargados.

8.3 Montaje en Function App (ruta /t8pm2022630278) y variable ROOT

Se realizó el montaje del File Share en la Function App t8ap2022630278. Se accedió a la configuración de “Path mounts” (o “App Service Mounted Storage”) para vincular el File Share t8rca2022630278 en la ruta /t8pm2022630278. Se instaló la variable de entorno ROOT con el valor /t8pm2022630278 para que la función Get pudiera servir los archivos estáticos.

- Se realizó la creación del montaje con credenciales del Storage Account (t8af2022630278).

Screenshot of Microsoft Azure portal showing the Function App blade. The search bar shows "Search resources, services, and docs (G+)" and the user "ggarciaq1800@alumno... INSTITUTO POLITÉCNICO NACIONAL".

The blade title is "Function App" and the sub-blade title is "Instituto Politécnico Nacional (correo.ipn.mx)".

Actions available include Create, Manage view, Refresh, Export to CSV, Open query, Assign tags, Start, Stop, Delete, Add to service group, and Group by none.

A message at the top says: "You are viewing a new version of Browse experience. Click here to access the old experience."

Filtering options include Filter for any field..., Subscription equals all, Resource Group equals all, Location equals all, and Add filter.

The main content area displays a lightning bolt icon and the message "No function apps to display". Below it, a brief description reads: "Create, build, deploy, and manage powerful web, mobile, and API apps for employees or customers using a single back-end. Build standards-based web apps and APIs using .NET, Java, Node.js, PHP, and Python." A "Create" button is present.

At the bottom, there is a "Give feedback" link and a "Showing 1 - 0 of 0. Display count: 20" dropdown.

Create Function App

Select a hosting option: Flex Consumption (selected), Functions Premium, App Service, Container Apps environment, and Consumption.

These options determine how your app scales, resources available per instance, and pricing. [Learn more about Functions hosting options](#).

Hosting plans	Flex Consumption	Functions Premium	App Service	Container Apps environment	Consumption
Get high scalability with compute choices, virtual networking, and pay-as-you-go billing.	✓	-	-	✓	✓
Scale to zero	✓	-	-	✓	✓
Scale behavior	Fast event-driven	Event-driven	Metrics based	Event-driven with KEDA	Event-driven
Virtual networking	✓	✓	✓	✓	-
Dedicated compute and prevent cold start	Optional with Always Ready	Minimum of 1 instance required	Minimum of 1 instance required	Optional with minimum replicas	-
Max scale out (instances)	1000	100	30	300	200

Select

At the bottom, the system status shows 14°C, Mostly cloudy, and the date/time is 10:57 p.m. 06/12/2025.

Create a function app, which lets you group functions as a logical unit for easier management, deployment and sharing of resources. Functions lets you execute your code in a serverless environment without having to first create a VM or publish a web application.

Project Details

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *

Resource Group *

Instance Details

Function App name *

Secure unique default hostname on. [More about this update](#)

Region *

Runtime stack *

[Review + create](#) [< Previous](#) [Next : Storage >](#)

Instances of your app are distributed across availability zones for increased reliability. [More about zone redundancy](#)

Zone redundancy

Enabled: Your Flex Consumption app will be zone redundant. This changes your app's required instance count per function or function group.

Disabled: Your Flex Consumption app will not be zone redundant.

[Review + create](#) [< Previous](#) [Next : Storage >](#)

Create Function App (Flex Consumption)

Storage account: t8af2022630278 (v2)

Your storage account has no network access restrictions. If you'd like to configure network security, either opt for a new storage account or select an existing storage account with network security.

Diagnostic Settings

Configure diagnostic settings to enable data monitoring for your storage account. Learn more ↗

Blob service diagnostic settings:

- Configure later (Recommended for custom controls) Select this option for control over log destinations, retention policies, and specific logs and metrics.
- Configure now (Recommended for basic controls) Select to configure Azure Log Analytics with storage, write logs, and transaction metrics for the blob service. You can still modify these settings later.

Review + create < Previous Next : Azure OpenAI >

Create Function App (Flex Consumption)

Storage account: t8af2022630278

Plan (New)

Hosting options and plans	Flex Consumption
Name	ASP-t8af2022630278-a505
Operating System	Linux
Region	Canada Central

Monitoring (New)

Application Insights	Enabled
Name	t8ap2022630278
Region	Canada Central

Azure OpenAI

Enable Azure OpenAI	Not enabled
---------------------	-------------

Deployment

Create < Previous Next > Download a template for automation

Create Function App (Flex Consumption)

Subscription: fd58a3da-fcef-47d1-ac0e-5b891faa4251
 Resource Group: t8-2022630278
 Name: t8ap2022630278
 Secure unique default hostname: Enabled
 Runtime stack: .NET 8 (LTS), isolated worker model
 Instance size: 2048 MB

Hosting

Create < Previous Next > Download a template for automation

14°C Mostly cloudy 11:02 p.m. 06/12/2025

t8ap2022630278

Resource group (move) : t8-2022630278 Status : Running Default domain : t8ap2022630278-f9hahsd6graeaf.canadacentral-01.azurewebsites.net
 Location (move) : Canada Central Operating System : Linux Plan type : Flex Consumption Subscription (move) : Azure for Students Instance Memory : 2048 MB
 Tags (edit) : Add tags

Functions Properties Notifications (0)

Create functions in your preferred environment

VS Code Desktop Other editors or CLI
 Best optimized for:
 • Local development within VS Code
 • Custom development tool requirements
 Create with VS Code Desktop Set up your editor

14°C Mostly cloudy 11:04 p.m. 06/12/2025

- Se accedió a “Configuration” y se agregó ROOT=/t8pm2022630278.

The image displays two screenshots of the Microsoft Azure portal interface, showing the configuration of environment variables for a Function App named "t8ap2022630278".

Screenshot 1: Environment variables

This screenshot shows the "Environment variables" section under "App settings". The sidebar on the left lists various settings like Overview, Activity log, and Functions. The main pane shows three environment variables:

Name	Value	Deployment slot setting	Source	Delete
APPLICATIONINSIGHTS_CONNECTION_STRING	Show value		App Service	
AzureWebJobsStorage	Show value		App Service	
DEPLOYMENT_STORAGE_CONNECTION_STRING	Show value		App Service	

Screenshot 2: Add/Edit application setting

This screenshot shows the "Add/Edit application setting" dialog. The sidebar on the left is identical to the first screenshot. The main pane shows a single application setting:

Name *	Value	Deployment slot setting
ROOT	/home/site/wwwroot/front-end	<input type="checkbox"/>

The "Name" field is set to "ROOT" and the "Value" field is set to "/home/site/wwwroot/front-end". The "Deployment slot setting" checkbox is unchecked.

Add/Edit application setting

Name: Server
Value: t8mysql2022630278.mysql.database.azure.com

Deployment slot setting:

App settings

APPLICATION

AzureWebJob

DEPLOYMENT

ROOT

Environment variables

Configuration (preview)

Add or remove favorites by pressing Ctrl+Shift+F

Add/Edit application setting

Name: UserID
Value: x

Deployment slot setting:

App settings

APPLICATION

AzureWebJob

DEPLOYMENT

ROOT

Server

Environment variables

Configuration (preview)

Add or remove favorites by pressing Ctrl+Shift+F

- Se verificó que la Function App pueda leer archivos via File.ReadAllText(ROOT + path).

Tabla de configuración principal:

Variable	Valor	Uso
----------	-------	-----

ROOT	/home/site/wwwroot/front-end	Raíz del front-end para Get
Server	t8mysql2022630278.mysql.database.azure.com	Conexión a MySQL PaaS
UserID	x	Usuario para MySQL
Password	Aaaaaaaaaaa0	Secreto para MySQL
Database	servicio_web	Base de datos del sistema

Tabla 12 Configuración de montaje /t8pm2022630278 y variables de entorno en la Function App.

8.4 Publicación de Azure Functions (t8ap2022630278)

Se realizó la publicación del proyecto de Azure Functions desde Visual Studio Code a la Function App t8ap2022630278 en la región Canada. Se accedió al panel de despliegue, se autenticó la suscripción de Azure for Students y se seleccionó la Function App correcta. Se validó la presencia de los endpoints (/api/login, /api/alta_usuario, /api/Get, etc.) y el correcto enlazado con MySQL PaaS.

- Se instaló la extensión de Azure Functions para VS Code (si no estaba instalada).
- Se accedió a “Deploy to Function App” y se seleccionó t8ap2022630278.
- Se verificó que las funciones respondieran con 200/400 según casos.

```

11:22:28 PM t8ap2022630278: [Kudu-RemoveWorkersStep] completed.
11:22:28 PM t8ap2022630278: [Kudu-SyncTriggerStep] starting.
11:23:28 PM t8ap2022630278: [Kudu-CleanUpStep] starting.
11:23:28 PM t8ap2022630278: [Kudu-CleanUpStep] completed.
11:23:28 PM t8ap2022630278: Finished deployment pipeline.
11:23:31 PM t8ap2022630278: Querying triggers...
11:23:34 PM t8ap2022630278: HTTP Trigger Urls:
alta_articulo: https://t8ap2022630278-f9ahsdsdgraeanef.canadacentral-01.azurewebsites.net/api/alta_articulo
alta_usuario: https://t8ap2022630278-f9ahsdsdgraeanef.canadacentral-01.azurewebsites.net/api/alta_usuario
borra_usuario: https://t8ap2022630278-f9ahsdsdgraeanef.canadacentral-01.azurewebsites.net/api/borra_usuario
compra_articulo: https://t8ap2022630278-f9ahsdsdgraeanef.canadacentral-01.azurewebsites.net/api/compra_articulo
consulta_articulos: https://t8ap2022630278-f9ahsdsdgraeanef.canadacentral-01.azurewebsites.net/api/consulta_articulos
consulta_carrito: https://t8ap2022630278-f9ahsdsdgraeanef.canadacentral-01.azurewebsites.net/api/consulta_carrito
consulta_usuario: https://t8ap2022630278-f9ahsdsdgraeanef.canadacentral-01.azurewebsites.net/api/consulta_usuario
eliminar_articulo_compra: https://t8ap2022630278-f9ahsdsdgraeanef.canadacentral-01.azurewebsites.net/api/eliminar_articulo_compra
eliminar_carrito_compra: https://t8ap2022630278-f9ahsdsdgraeanef.canadacentral-01.azurewebsites.net/api/eliminar_carrito_compra
finaliza_compra: https://t8ap2022630278-f9ahsdsdgraeanef.canadacentral-01.azurewebsites.net/api/finaliza_compra
get: https://t8ap2022630278-f9ahsdsdgraeanef.canadacentral-01.azurewebsites.net/api/get
login: https://t8ap2022630278-f9ahsdsdgraeanef.canadacentral-01.azurewebsites.net/api/login
modifica_articulo_compra: https://t8ap2022630278-f9ahsdsdgraeanef.canadacentral-01.azurewebsites.net/api/modifica_articulo_compra
modifica_usuario: https://t8ap2022630278-f9ahsdsdgraeanef.canadacentral-01.azurewebsites.net/api/modifica_usuario

```

Figura 32. Panel de publicación en VS Code y confirmación de endpoints en la Function App.

8.5 Pruebas de acceso a la aplicación: URL [/api/Get?nombre=/prueba.html](https://t8ap2022630278.azurewebsites.net/api/Get?nombre=/prueba.html)

Se accedió al front-end mediante la URL pública de la Function App:

<https://t8ap2022630278.azurewebsites.net/api/Get?nombre=/prueba.html>

Se realizó la navegación por las distintas vistas (“Captura de artículo”, “Compra de artículos”, “Artículos en el carrito”) y se efectuaron llamadas al back-end desplegado.

Se verificó que las operaciones de carrito y stock funcionaran como en local, incluyendo validaciones y transacciones.

- Se accedió desde un dispositivo móvil y desde escritorio para validar disponibilidad.
- Se realizó la prueba de login para obtener token, y la recuperación de id_usuario.
- Se confirmó el flujo completo: alta, búsqueda, compra, modificación y eliminación del carrito.

Tabla de validaciones de producción:

Prueba	Resultado	Observaciones
Carga de prueba.html	OK	Servido vía Get desde Azure Files
Login (token)	OK	Actualiza usuarios.token
Alta de artículo	OK	Inserta en stock y opcional fotos_articulos
Consulta por palabra	OK	LIKE en nombre y descripcion
Compra y descuentos	OK	Transacción en carrito_compra y stock
Eliminar artículo/carrito	OK	Restaura stock y borra entradas
Modificar cantidad	OK	Valida stock y límites de carrito

Tabla 13 Tabla de validaciones de producción

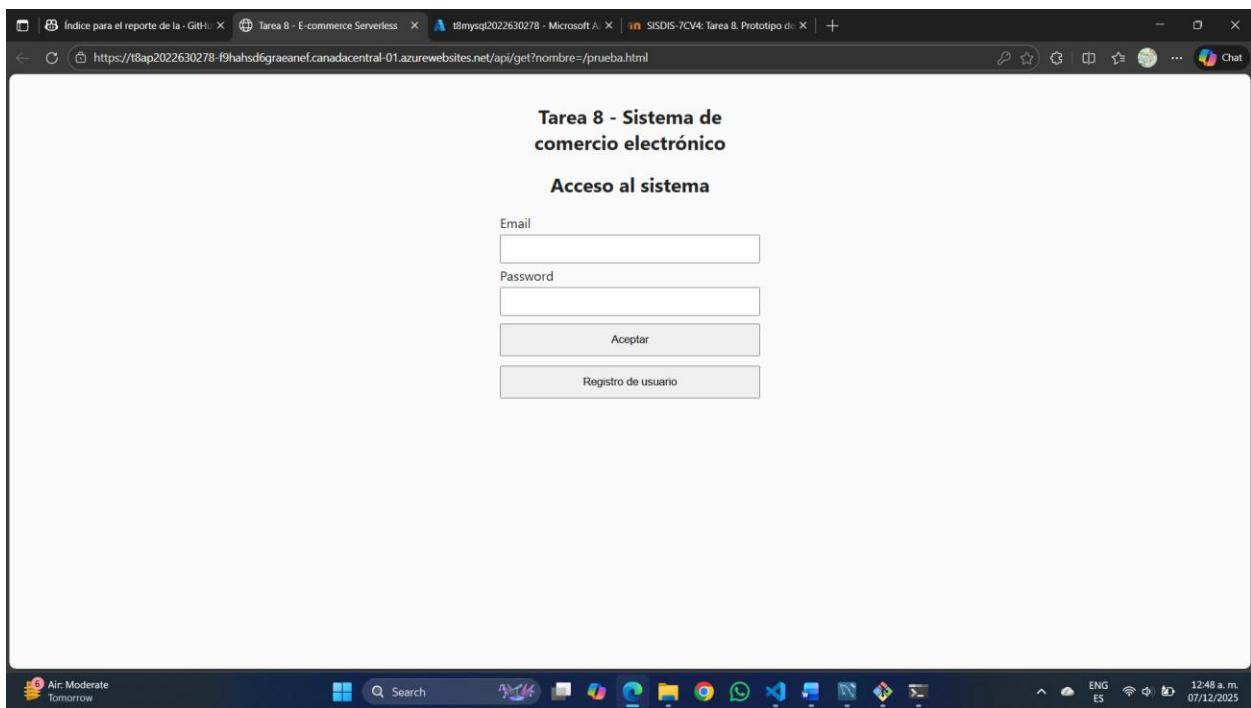


Figura 33. Acceso a la URL pública del front-end y ejecución de flujos en producción (Azure – Canada).

9 Evidencias requeridas (capturas de pantalla completas con fecha y hora)

Se realizaron y documentaron las evidencias solicitadas para el reporte, asegurando que todas las capturas sean pantallas completas, legibles y con barra de tareas mostrando fecha y hora. Las evidencias se organizaron por tipo: configuración en Azure, pruebas en dispositivo móvil del front-end y pruebas unitarias locales del back-end con curl. A continuación se indica dónde colocar cada imagen y el contenido mínimo que debe mostrar.

9.1 Variables de entorno en Azure (Function App: Configuration)

Se accedió a la Function App t8ap2022630278 y se realizó la captura de la sección “Configuration”, mostrando las variables Server, UserID, Password, Database y ROOT. Se aseguró que ROOT apunte a /t8pm2022630278 y que la región sea Canada, acorde a la nomenclatura indicada.

- Se realizó la verificación de valores y su activación.
- Se accedió al apartado de “Path mounts” para corroborar el montaje del File Share.

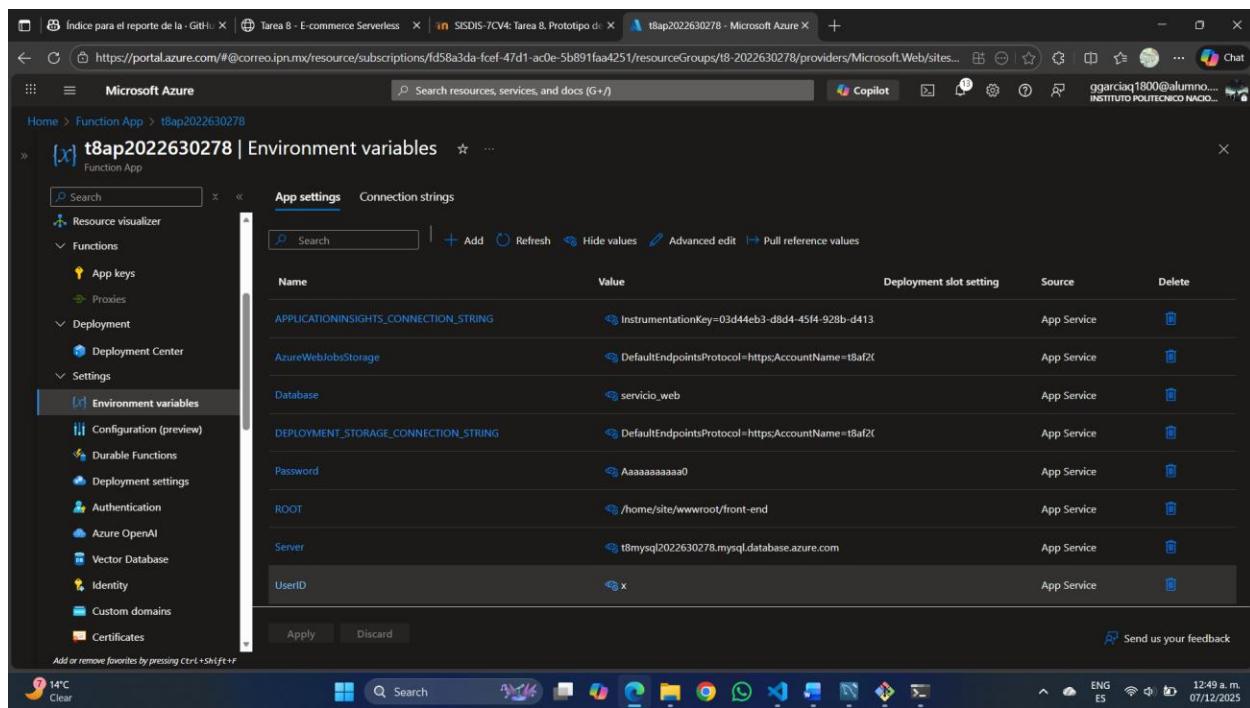


Figura 34. Configuración de variables de entorno y montaje de ruta en t8ap2022630278 (pantalla completa con fecha/hora).

9.2 Pruebas en dispositivo móvil del front-end

Se accedió al front-end desde un teléfono o tableta, y se realizaron las pruebas de cada requerimiento funcional. Cada captura debe mostrar la vista correspondiente, los controles y los resultados visibles, además de la fecha y hora del dispositivo.

Tabla de evidencias por vista:

Vista / Acción	Qué debe mostrar la captura	Endpoint implicado
Captura de artículo	Formulario con nombre, descripción, precio, cantidad y foto; confirmación de alta	alta_articulo
Búsqueda y listado	Campo de palabra clave; lista con mini-foto, nombre, descripción, precio; botones +/- y “Compra”	consulta_articulos, modifica_carrito_compra, compra_articulo
Compra de artículo	Confirmación visible de la compra y ajuste de stock	compra_articulo

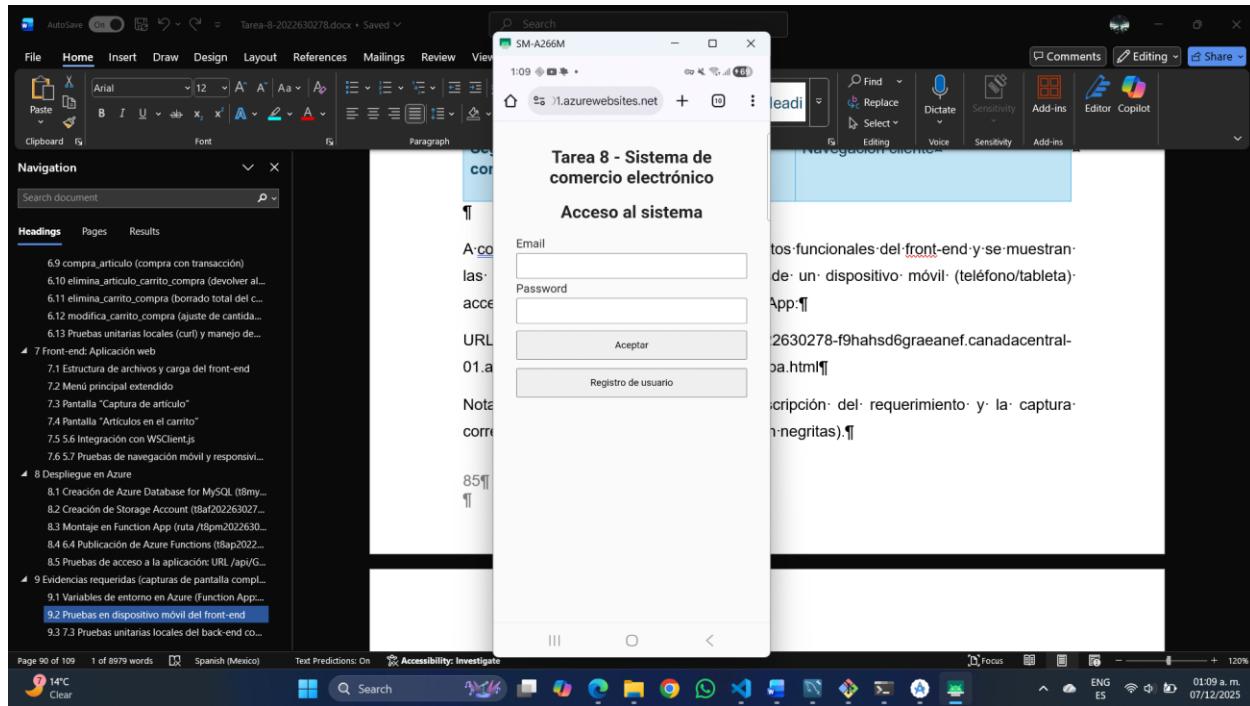
Carrito de compra	Lista de artículos, cantidad, precio, costo por artículo y total	(consulta/refresh del carrito)
Eliminar artículo	Botón y confirmación de eliminación; stock restaurado	elimina_articulo_carrito_compra
Eliminar carrito	Vaciar el carrito y restauración del stock	elimina_carrito_compra
Seguir comprando	Navegación de regreso a la vista de “Compra de artículos”	Navegación cliente

Tabla 14 Tabla de evidencias por vista

A continuación se describen los requerimientos funcionales del front-end y se muestran las evidencias de prueba ejecutadas desde un dispositivo móvil (teléfono/tableta) accediendo a la URL pública de la Function App:

URL de acceso:

<https://t8ap2022630278-f9hahsd6graeanef.canadacentral-01.azurewebsites.net/api/get?nombre=/prueba.html>



9.2.1 Requerimiento 1: Pantalla “Captura de artículo”

- Descripción: Al seleccionar “Captura de artículo”, se muestra una pantalla para capturar nombre, descripción, precio, cantidad en existencia y la fotografía del artículo. El botón “Guardar” envía la alta al back-end (función alta_articulo), incluyendo id_usuario y token.

The figure consists of two side-by-side screenshots of a mobile application interface. Both screenshots show a header with the text "Tarea 8 - Sistema de comercio electrónico".
The left screenshot is titled "Acceso al sistema" and contains fields for "Email" (with value "x1@x1.com") and "Password" (with value "*****"). It includes a "Aceptar" button and a "Registro de usuario" button.
The right screenshot is titled "Captura de artículo" and contains fields for "Nombre *" (with value "Leche Alpura 1L"), "Descripción *" (with value "Leche"), "Precio *" (with value "47.99"), and "Cantidad *" (with value "10"). It includes a "Fotografía" field showing a thumbnail of a milk carton and a "Choose Files" button with the file name "6347.jpg". It also includes "Guardar" and "Regresar" buttons.



Figura 1. Pantalla “Captura de artículo” en móvil (campos visibles y carga de imagen)



Figura 2. Confirmación de artículo registrado (toast o alerta de éxito en móvil)

9.2.2 Requerimiento 2: Pantalla “Compra de artículos” con búsqueda por palabra clave

- Descripción: La pantalla permite ingresar una palabra clave, que se busca con LIKE en las columnas nombre y descripción de la tabla stock. Se llama a la función consulta_articulos del back-end.
- Evidencia:



Figura 3. Búsqueda por palabra clave y resultados con imagen, nombre, descripción y precio en móvil

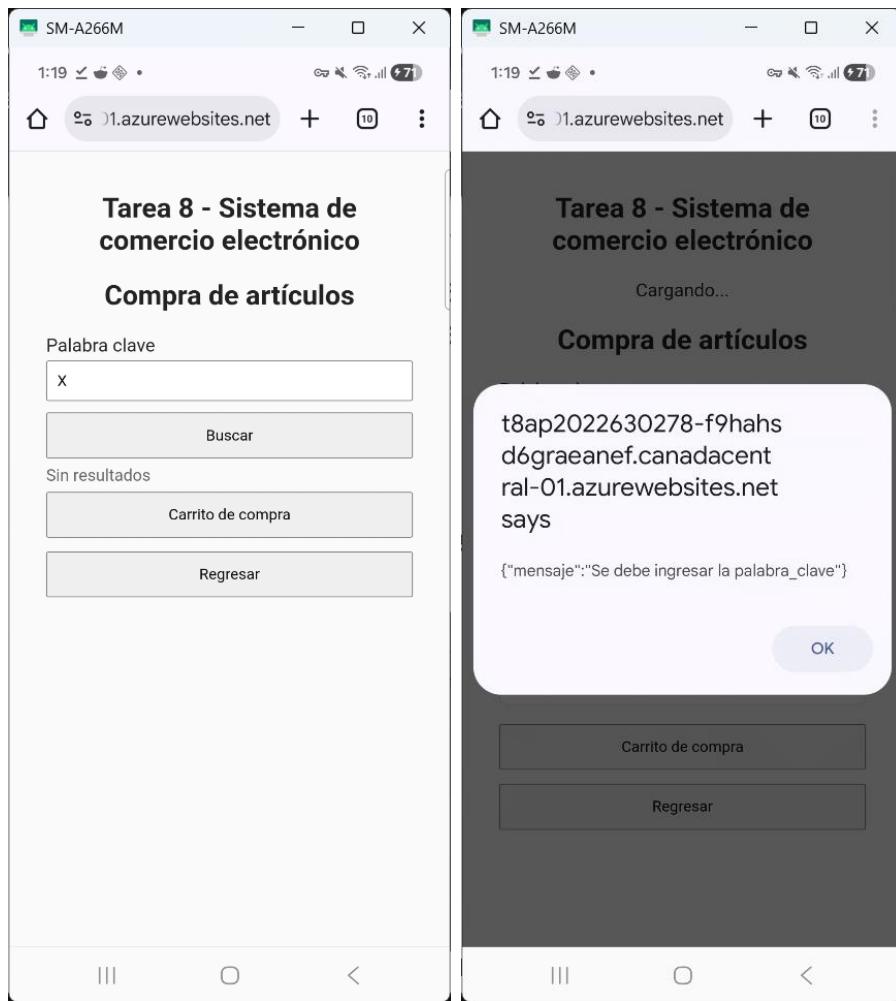


Figura 4. Indicador de “Sin resultados” cuando la búsqueda no devuelve artículos

9.2.3 Requerimiento 3: Agregar artículo al carrito desde “Compra de artículos”

- Descripción: Cada resultado muestra un campo “Cantidad” y el botón “Agregar al carrito”. Al presionar, se invoca compra_articulo (back-end), que inserta/actualiza en carrito_compra y descuenta del stock bajo transacción.
- Evidencia:



Figura 5. Botón “Agregar al carrito” con cantidad especificada y mensaje de agregado



Figura 6. Confirmación de agregado al carrito (toast o alerta de éxito)

9.2.4 Requerimiento 5: Pantalla “Artículos en el carrito” con total

- Descripción: El botón “Carrito de compra” despliega los artículos en carrito_compra con mini-imagen, nombre, cantidad, precio y costo (cantidad x precio), así como el total. Se invoca consulta_carrito (back-end).
- Evidencia:



Figura 7. Pantalla “Artículos en el carrito” mostrando líneas con imagen, nombre, cantidad, precio y costo

9.2.5 Requerimiento 6: Modificación de cantidad en el carrito (+/-)

- Descripción: En “Artículos en el carrito” existen botones “+” y “-” por línea para modificar la cantidad (función `modifica_carrito_compra`). La operación ajusta carrito y stock dentro de una transacción. Maneja casos de error (stock insuficiente o cantidad ya en 0) mostrando mensajes.
- Evidencia:



Figura 9. Incremento de cantidad con botón “+” y actualización de costo/total



Figura 10. Decremento de cantidad con botón “-” y actualización de costo/total

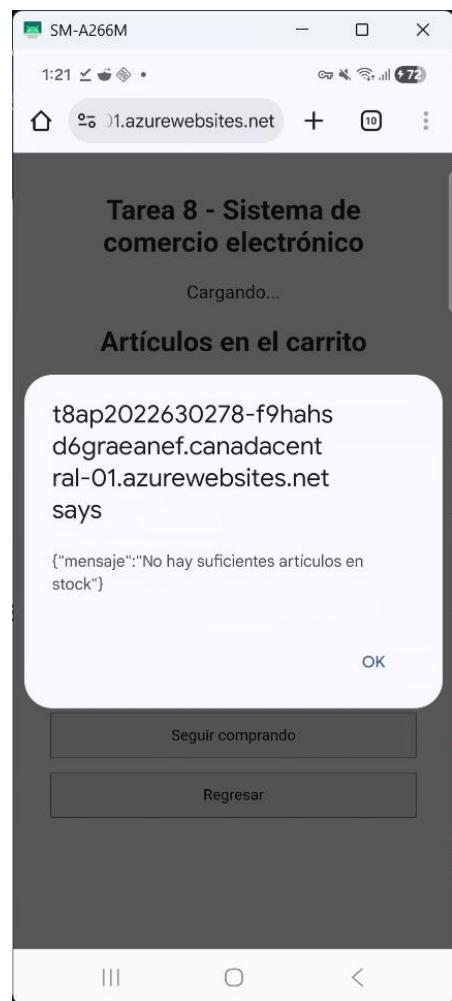


Figura 11. Mensaje de error “No hay suficientes artículos en stock” al intentar exceder existencias

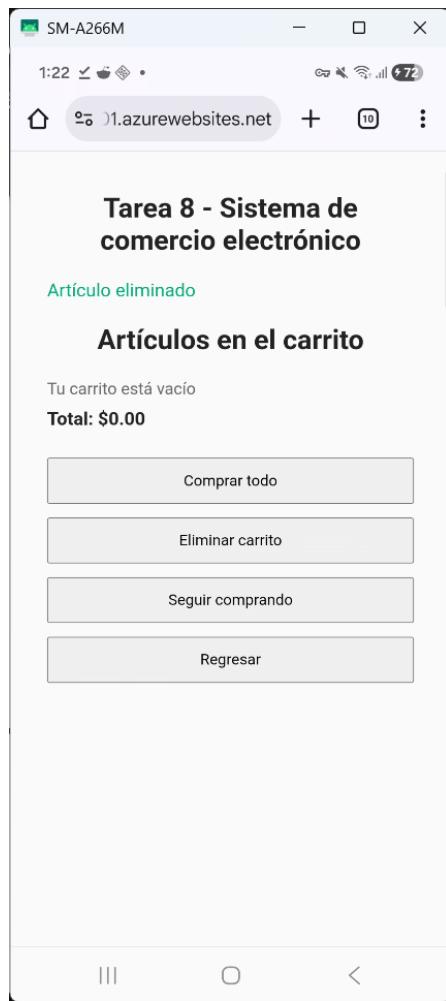


Figura 12. Mensaje de error “No hay más artículos en el carrito” al intentar decrementar por debajo de 0

9.2.6 Requerimiento 6 (extensión coherente) y 7: Eliminar artículo y eliminar carrito

- Descripción:
 - “Eliminar artículo”: Invoca `elimina_articulo_carrito_compra`, regresa la cantidad al stock y borra el artículo dentro del carrito dentro de una transacción.
 - “Eliminar carrito”: Invoca `elimina_carrito_compra`, regresa todas las cantidades al stock y borra todos los registros del carrito dentro de una transacción.

- Evidencia:

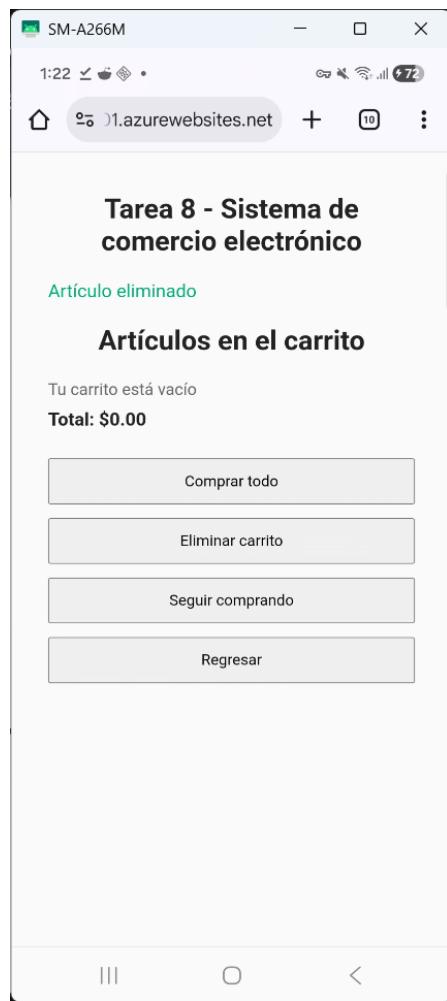


Figura 13. Eliminación de artículo individual del carrito y actualización del total



Figura 14. Eliminación completa del carrito y confirmación de carrito vacío

9.2.7 Botón adicional “Comprar todo” (checkout)

- Descripción: El botón “Comprar todo” realiza el checkout creando una orden (ordenes) y detalles (orden_detalle) a partir del carrito, y lo vacía sin regresar stock. Se invoca la función finaliza_compra (back-end). Devuelve id_orden y total.
- Evidencia:



Figura 15. Ejecución de “Comprar todo” y confirmación con número de orden y total



Figura 16. Carrito vacío tras finalizar la compra

9.2.8 Navegación entre pantallas y login/registro

- Descripción:
 - Desde el menú principal se accede a Perfil, Captura de artículo y Compra de artículos.
 - El login genera token y se obtiene id_usuario con consulta_usuario.
 - El registro de usuario y modificación de perfil usan las funciones existentes (alta_usuario, consulta_usuario, modifica_usuario, borra_usuario).
- Evidencia:

Figura 17. Pantalla de login en móvil

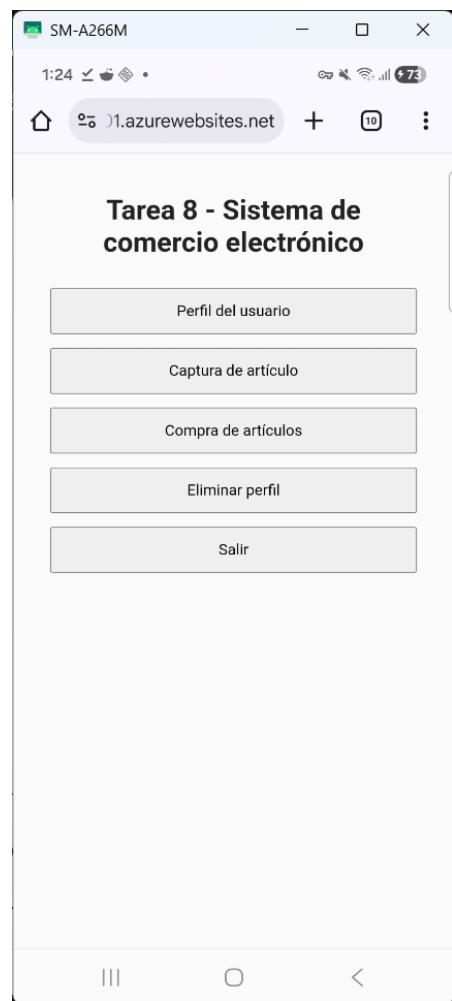


Figura 18. Menú principal con opciones “Perfil del usuario”, “Captura de artículo”, “Compra de artículos”

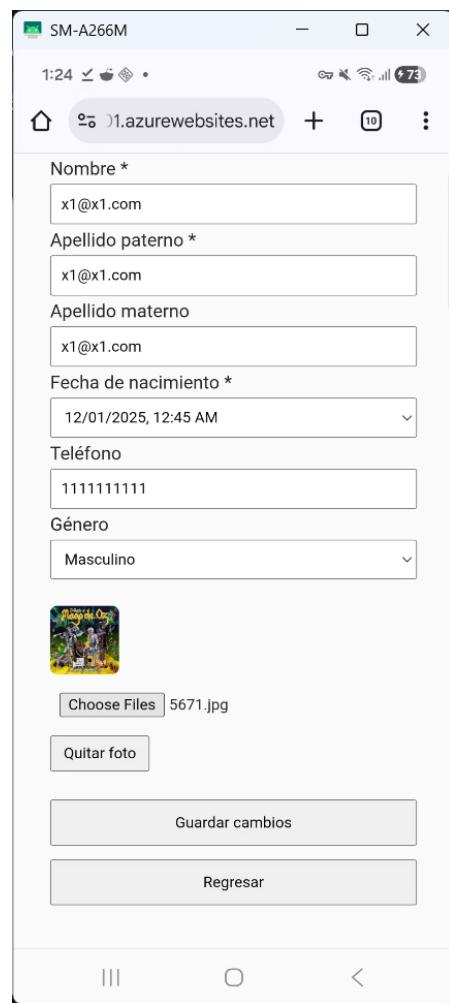


Figura 19. Pantalla “Perfil del usuario” con edición de datos e imagen

9.3 Pruebas unitarias del despliegue del back-end con curl

Se realizaron pruebas locales invocando cada endpoint del back-end con curl, capturando la consola con los comandos y las respuestas en JSON, además del código HTTP (cuando sea visible). Las capturas deben ser de la terminal en pantalla completa, con barra de tareas y fecha/hora.

Tabla de endpoints a evidenciar con ejemplo de llamada:

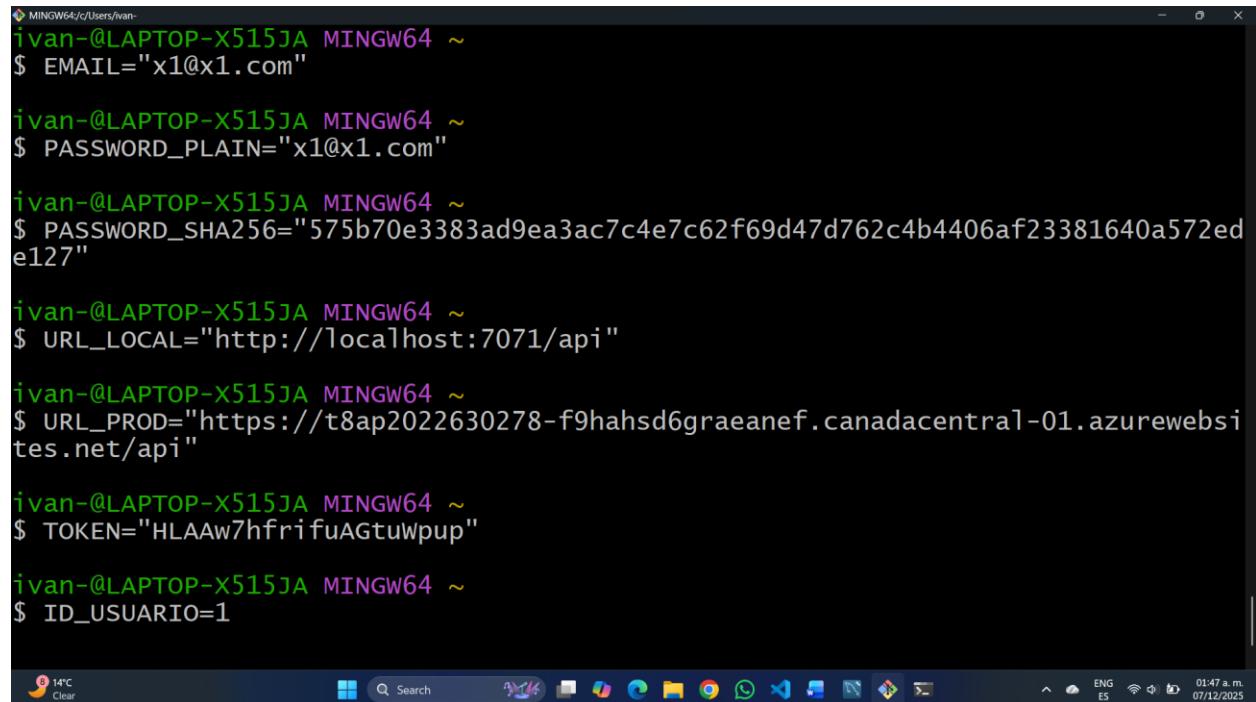
Función	Método	Ejemplo de curl (local)
login	GET	curl "http://localhost:7071/api/login?email=...&password=...sha256..."
alta_usuario	POST	curl -X POST -H "Content-Type: application/json" -d "..." http://localhost:7071/api/alta_usuario
consulta_usuario	GET	curl "http://localhost:7071/api/consulta_usuario?email=...&token=..."
modifica_usuario	PUT	curl -X PUT -H "Content-Type: application/json" "http://localhost:7071/api/modifica_usuario?email=...&token=..." -d "..."
borra_usuario	DELETE	curl -X DELETE "http://localhost:7071/api/borra_usuario?email=...&token=..."
alta_articulo	POST	curl -X POST -H "Content-Type: application/json" -d "..." http://localhost:7071/api/alta_articulo
consulta_articulos	GET	curl "http://localhost:7071/api/consulta_articulos?palabra_clave=...&id_usuario=...&token=..."
compra_articulo	POST	curl -X POST -H "Content-Type: application/json" -d "..." http://localhost:7071/api/compra_articulo
elimina_articulo_carrito_compra	DELETE	curl -X DELETE "http://localhost:7071/api/elimina_articulo_carrito_compra?id_usuario=...&id_articulo=...&token=..."
elimina_carrito_compra	DELETE	curl -X DELETE "http://localhost:7071/api/elimina_carrito_compra?id_usuario=...&token=..."
modifica_carrito_compra	PUT	curl -X PUT -H "Content-Type: application/json" "http://localhost:7071/api/modifica_carrito_compra?id_usuario=...&token=..." -d "{"id_articulo":..., incremento:...}"

Tabla 15 Tabla de endpoints

9.4 Pruebas unitarias locales del back-end con curl (detalladas)

Las siguientes pruebas se ejecutan en Git Bash, contra el entorno local (<http://localhost:7071/api>) y/o producción según se requiera evidencia. Sustituye los valores de ejemplo por tus datos reales.

Variables de ejemplo a usar en Git Bash:



The screenshot shows a Windows desktop environment with a terminal window open in Git Bash. The terminal window displays several environment variable assignments:

```
ivan-@LAPTOP-X515JA MINGW64 ~
$ EMAIL="x1@x1.com"

ivan-@LAPTOP-X515JA MINGW64 ~
$ PASSWORD_PLAIN="x1@x1.com"

ivan-@LAPTOP-X515JA MINGW64 ~
$ PASSWORD_SHA256="575b70e3383ad9ea3ac7c4e7c62f69d47d762c4b4406af23381640a572ed
e127"

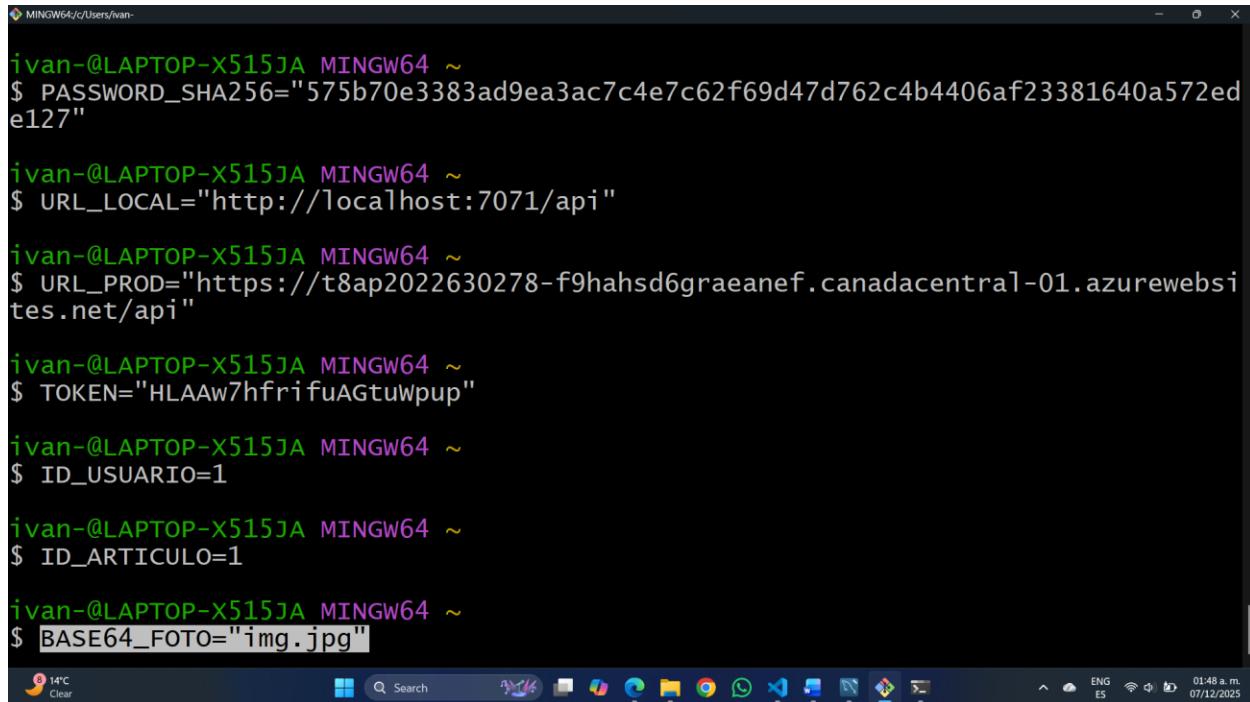
ivan-@LAPTOP-X515JA MINGW64 ~
$ URL_LOCAL="http://localhost:7071/api"

ivan-@LAPTOP-X515JA MINGW64 ~
$ URL_PROD="https://t8ap2022630278-f9hahsd6graeane.f.canadacentral-01.azurewebsi
tes.net/api"

ivan-@LAPTOP-X515JA MINGW64 ~
$ TOKEN="HLAAw7hfrijuAGtuwpup"

ivan-@LAPTOP-X515JA MINGW64 ~
$ ID_USUARIO=1
```

The taskbar at the bottom of the screen shows various icons for system functions like weather, search, and file explorer, along with the current date and time (01:47 a.m. 07/12/2025).



```
ivan-@LAPTOP-X515JA MINGW64 ~
$ PASSWORD_SHA256="575b70e3383ad9ea3ac7c4e7c62f69d47d762c4b4406af23381640a572ed
e127"

ivan-@LAPTOP-X515JA MINGW64 ~
$ URL_LOCAL="http://localhost:7071/api"

ivan-@LAPTOP-X515JA MINGW64 ~
$ URL_PROD="https://t8ap2022630278-f9hahsd6graeanef.canadacentral-01.azurewebsi
tes.net/api"

ivan-@LAPTOP-X515JA MINGW64 ~
$ TOKEN="HLAAw7hfrifuAGtuWpup"

ivan-@LAPTOP-X515JA MINGW64 ~
$ ID_USUARIO=1

ivan-@LAPTOP-X515JA MINGW64 ~
$ ID_ARTICULO=1

ivan-@LAPTOP-X515JA MINGW64 ~
$ BASE64_FOTO="img.jpg"
```

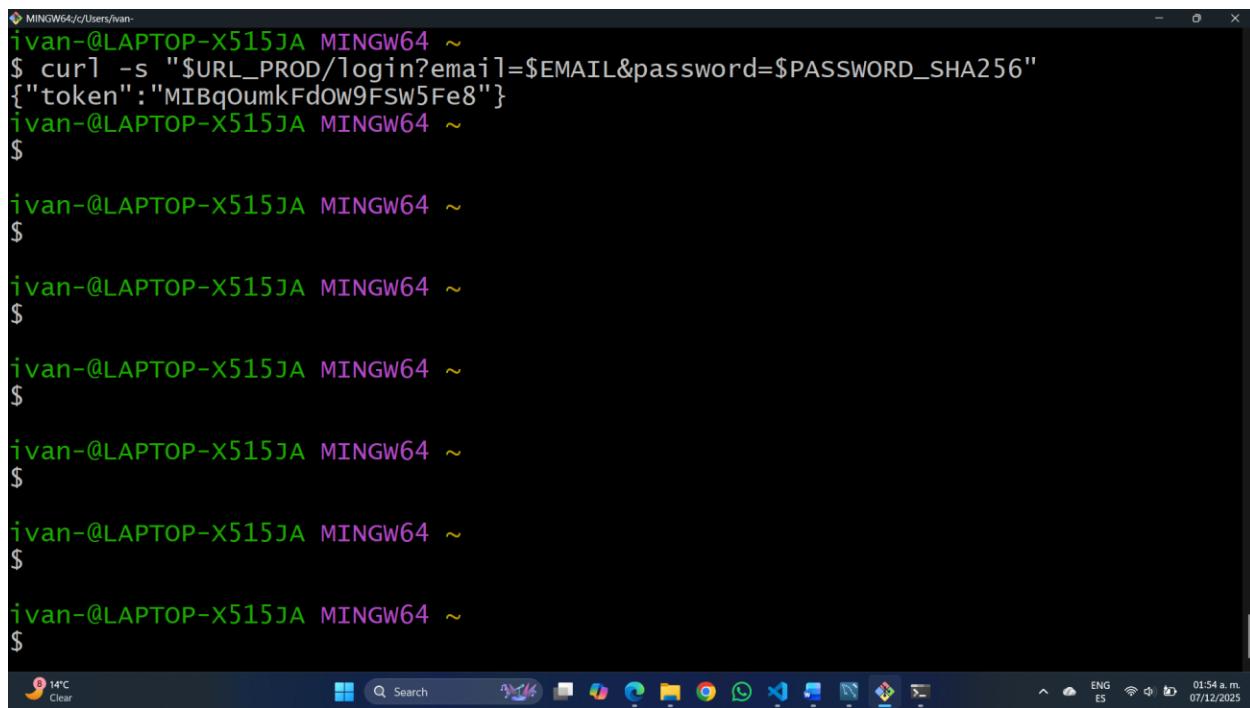
Figura 20. Preparación de variables

9.4.1 Prueba 0. Login y obtención de token (previo a las demás)

- Descripción: Verifica login y obtención de token.

```
curl -s "$URL_PROD/login?email=$EMAIL&password=$PASSWORD_SHA256"
```

- Evidencia:



```
MINGW64/c/Users/ivan-  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ curl -s "$URL_PROD/login?email=$EMAIL&password=$PASSWORD_SHA256"  
{\"token\":\"MIBqOumkFdOW9FSW5Fe8\"}  
ivan-@LAPTOP-X515JA MINGW64 ~  
$  
  
14°C Clear
```

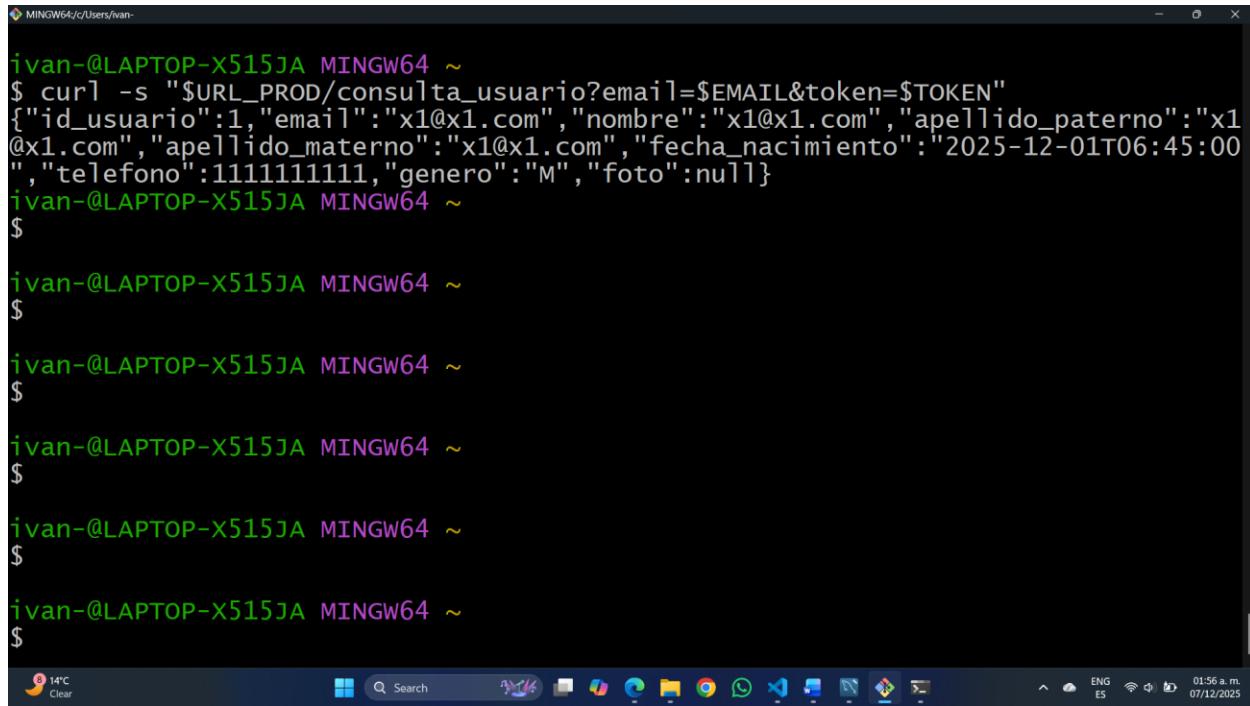
Figura 20. Respuesta JSON de login con token

9.4.2 Prueba 0.1 Consulta de usuario para obtener id_usuario

- Descripción: Verifica consulta_usuario y lectura de id_usuario.

```
curl -s "$URL_PROD/consulta_usuario?email=$EMAIL&token=$TOKEN"
```

- Evidencia:



```
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s "$URL_PROD/consulta_usuario?email=$EMAIL&token=$TOKEN"
{"id_usuario":1,"email":"x1@x1.com","nombre":"x1@x1.com","apellido_paterno":"x1@x1.com","apellido_materno":"x1@x1.com","fecha_nacimiento":"2025-12-01T06:45:00","telefono":1111111111,"genero":"M","foto":null}
ivan-@LAPTOP-X515JA MINGW64 ~
$
```

```
14°C Clear
```

```
Search
```

```
Windows Start
```

```
File Explorer
```

```
Calculator
```

```
Edge
```

```
Google Chrome
```

```
PowerShell
```

```
Notepad
```

```
Snipping Tool
```

```
File History
```

```
Task View
```

```
System
```

```
Control Panel
```

```
Network
```

```
File
```

```
Help
```

```
01:56 a.m.
```

```
ENG ES
```

```
Wi-Fi
```

```
07/12/2025
```

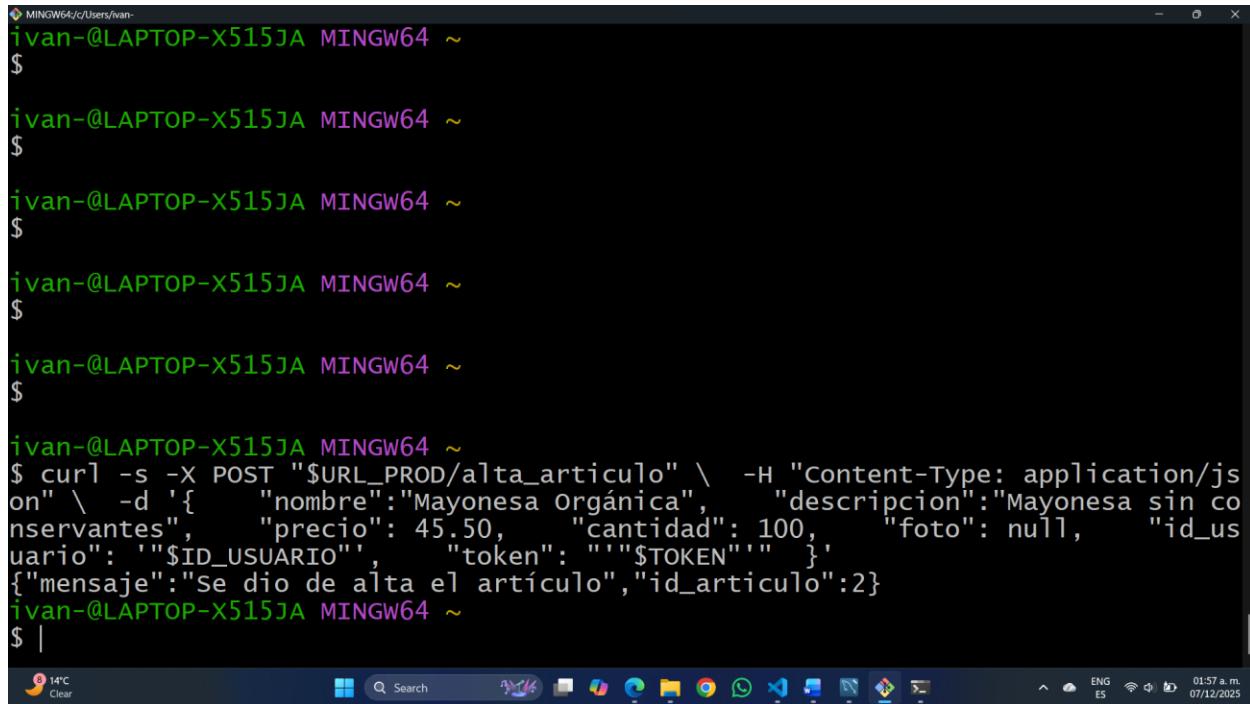
Figura 21. Respuesta JSON con id_usuario y perfil

9.4.3 Requerimiento 1 (back-end): alta_articulo

- Descripción: Alta de artículos en stock y opcionalmente foto en fotos_articulos.

```
curl -s -X POST "$URL_PROD/alta_articulo" \ -H "Content-Type: application/json" \ -d '{ "nombre": "Mayonesa Orgánica", "descripcion": "Mayonesa sin conservantes", "precio": 45.50, "cantidad": 100, "foto": null, "id_usuario": "'"$ID_USUARIO"'", "token": "'"$TOKEN"'"}'
```

- Evidencia:



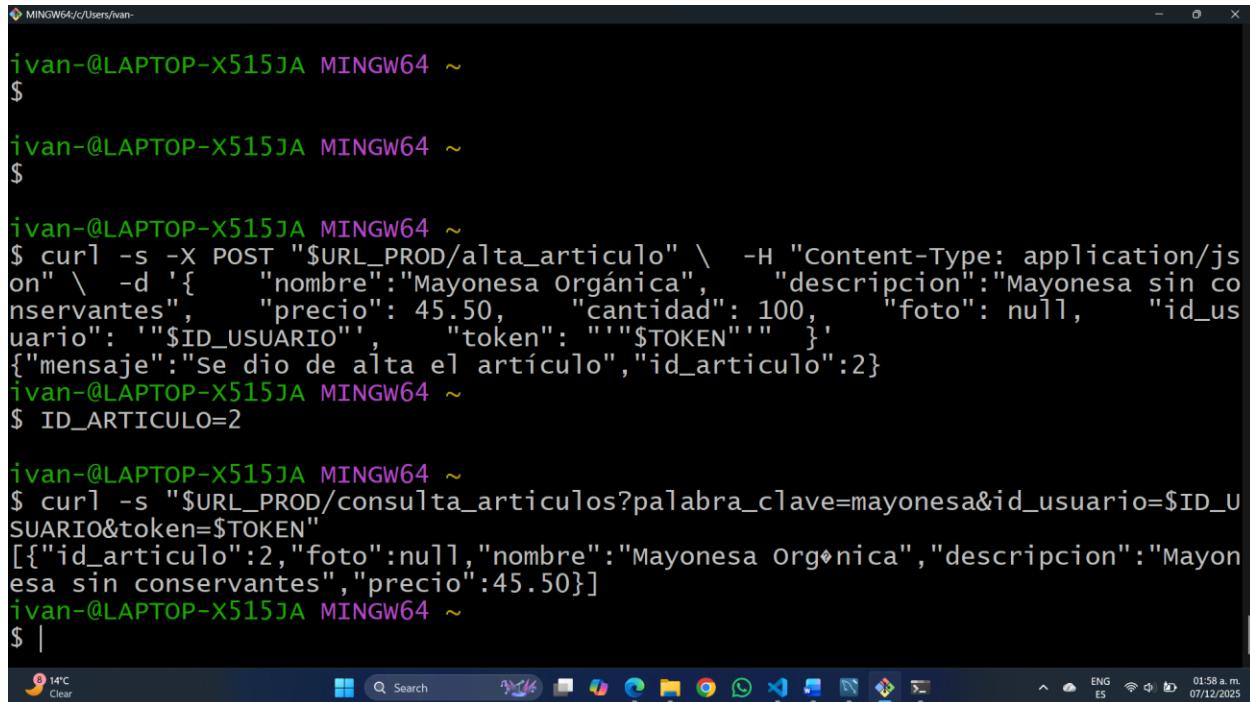
```
MINGW64/c/Users/ivan-  
ivan-@LAPTOP-X515JA MINGW64 ~  
$  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ curl -s -X POST "$URL_PROD/alta_articulo" \ -H "Content-Type: application/json" \ -d '{ "nombre": "Mayonesa Orgánica", "descripcion": "Mayonesa sin conservantes", "precio": 45.50, "cantidad": 100, "foto": null, "id_usuario": "'.$ID_USUARIO'", "token": "'".$TOKEN"' }'  
{ "mensaje": "Se dio de alta el artículo", "id_articulo": 2 }  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ |
```

Figura 22. Respuesta de alta de artículo con id_articulo

9.4.4 Requerimiento 2: consulta_articulos (búsqueda por palabra clave)

```
curl -s  
"$URL_PROD/consulta_articulos?palabra_clave=mayonesa&id_usuario=$ID_USUARIO&token=$TOKEN"
```

- Evidencia:



A screenshot of a Windows desktop environment showing a terminal window titled 'MINGW64/c/Users/ivan-'. The terminal window displays a series of curl commands being run in a terminal session. The commands are used to perform a POST request to '\$URL_PROD/alta_articulo' and a GET request to '\$URL_PROD/consulta_articulos?palabra_clave=mayonesa&id_usuario=\$ID_USUARIO&token=\$TOKEN'. The output of the GET request is a JSON array containing one element, which is a JSON object representing an article with id 2, name 'Mayonesa Orgánica', price 45.50, and quantity 100.

```
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$URL_PROD/alta_articulo" \ -H "Content-Type: application/json" \ -d '{ "nombre":"Mayonesa Orgánica", "descripcion":"Mayonesa sin conservantes", "precio": 45.50, "cantidad": 100, "foto": null, "id_usuario": "'.$ID_USUARIO'", "token": "'".$TOKEN"' }' {"mensaje":"Se dio de alta el artículo","id_articulo":2}
ivan-@LAPTOP-X515JA MINGW64 ~
$ ID_ARTICULO=2

ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s "$URL_PROD/consulta_articulos?palabra_clave=mayonesa&id_usuario=$ID_USUARIO&token=$TOKEN"
[{"id_articulo":2,"foto":null,"nombre":"Mayonesa Orgánica","descripcion":"Mayonesa sin conservantes","precio":45.50}]
ivan-@LAPTOP-X515JA MINGW64 ~
$ |
```

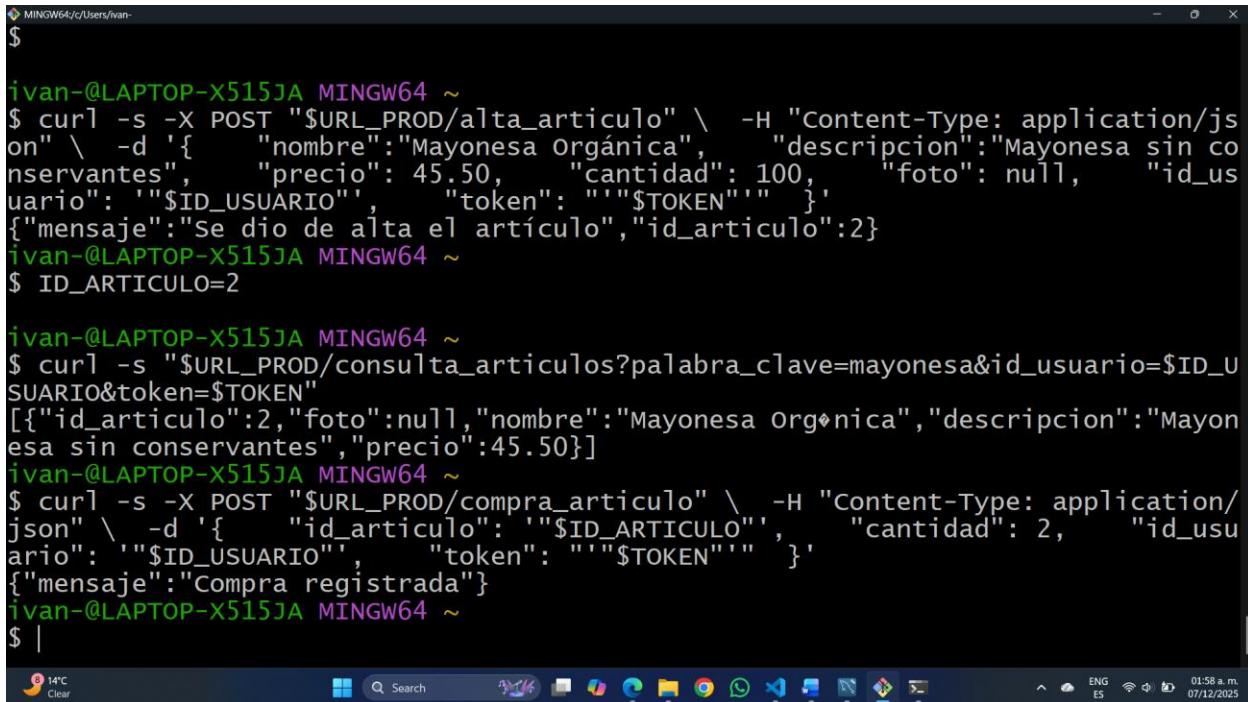
Figura 23. Respuesta JSON con lista de artículos filtrados por palabra clave

9.4.5 Requerimiento 3: compra_articulo

- Compra 2 unidades:

```
curl -s -X POST "$URL_PROD/compra_articulo" \ -H "Content-Type: application/json" \
-d '{ "id_articulo": "'.$ID_ARTICULO'", "cantidad": 2, "id_usuario": "'.$ID_USUARIO'", "token": "'.$TOKEN.'" }'
```

- Evidencia:



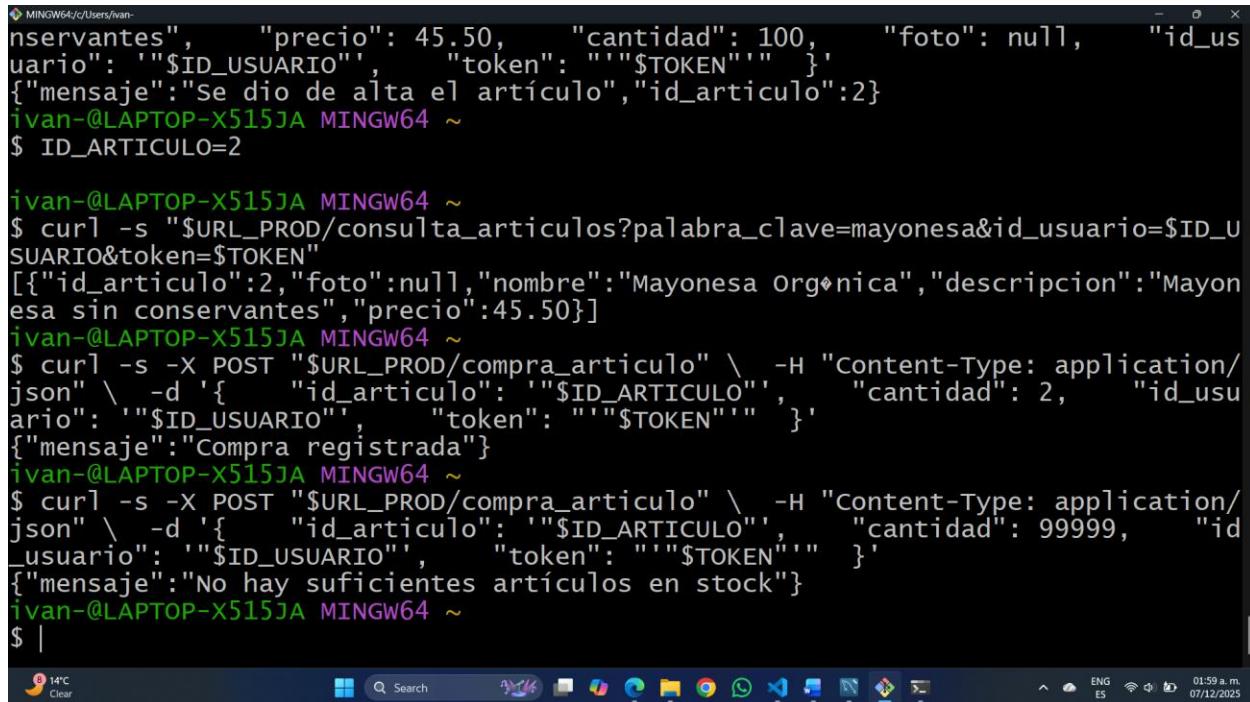
```
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$URL_PROD/alta_articulo" \ -H "Content-Type: application/json" \
-d '{ "nombre":"Mayonesa Orgánica", "descripcion":"Mayonesa sin conservantes", "precio": 45.50, "cantidad": 100, "foto": null, "id_usuario": "'.$ID_USUARIO'", "token": "'".$TOKEN"" }'
{"mensaje":"Se dio de alta el artículo","id_articulo":2}
ivan-@LAPTOP-X515JA MINGW64 ~
$ ID_ARTICULO=2

ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s "$URL_PROD/consulta_articulos?palabra_clave=mayonesa&id_usuario=$ID_USUARIO&token=$TOKEN"
[{"id_articulo":2,"foto":null,"nombre":"Mayonesa Orgánica","descripcion":"Mayonesa sin conservantes","precio":45.50}]
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$URL_PROD/compra_articulo" \ -H "Content-Type: application/json" \
-d '{ "id_articulo": "'.$ID_ARTICULO'", "cantidad": 2, "id_usuario": "'.$ID_USUARIO'", "token": "'".$TOKEN"" }'
{"mensaje":"Compra registrada"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ |
```

Figura 24. Respuesta de compra registrada

- Error de stock insuficiente (ejemplo compra de 99999 unidades):

```
curl -s -X POST "$URL_PROD/compra_articulo" \ -H "Content-Type: application/json" \
-d '{ "id_articulo": "'.$ID_ARTICULO'", "cantidad": 99999, "id_usuario": "'.$ID_USUARIO'", "token": "'".$TOKEN"" }'
```



```

MINGW64/c/Users/ivan-
nservantes", "precio": 45.50, "cantidad": 100, "foto": null, "id_us
uario": "$ID_USUARIO", "token": "$TOKEN"}'
>{"mensaje":"Se dio de alta el artículo","id_articulo":2}
ivan-@LAPTOP-X515JA MINGW64 ~
$ ID_ARTICULO=2

ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s "$URL_PROD/consulta_articulos?palabra_clave=mayonesa&id_usuario=$ID_U
SUARIO&token=$TOKEN"
[{"id_articulo":2,"foto":null,"nombre":"Mayonesa Orgánica","descripcion":"Mayon
esa sin conservantes","precio":45.50}]
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$URL_PROD/compra_articulo" \
-H "Content-Type: application/
json" \
-d '{
    "id_articulo": "$ID_ARTICULO",
    "cantidad": 2,
    "id_usu
ario": "$ID_USUARIO",
    "token": "$TOKEN"}'
>{"mensaje":"Compra registrada"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$URL_PROD/compra_articulo" \
-H "Content-Type: application/
json" \
-d '{
    "id_articulo": "$ID_ARTICULO",
    "cantidad": 99999,
    "id_
usuario": "$ID_USUARIO",
    "token": "$TOKEN"}'
>{"mensaje":"No hay suficientes artículos en stock"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ |

```

Figura 25. Error de stock insuficiente (HTTP 400 con mensaje)

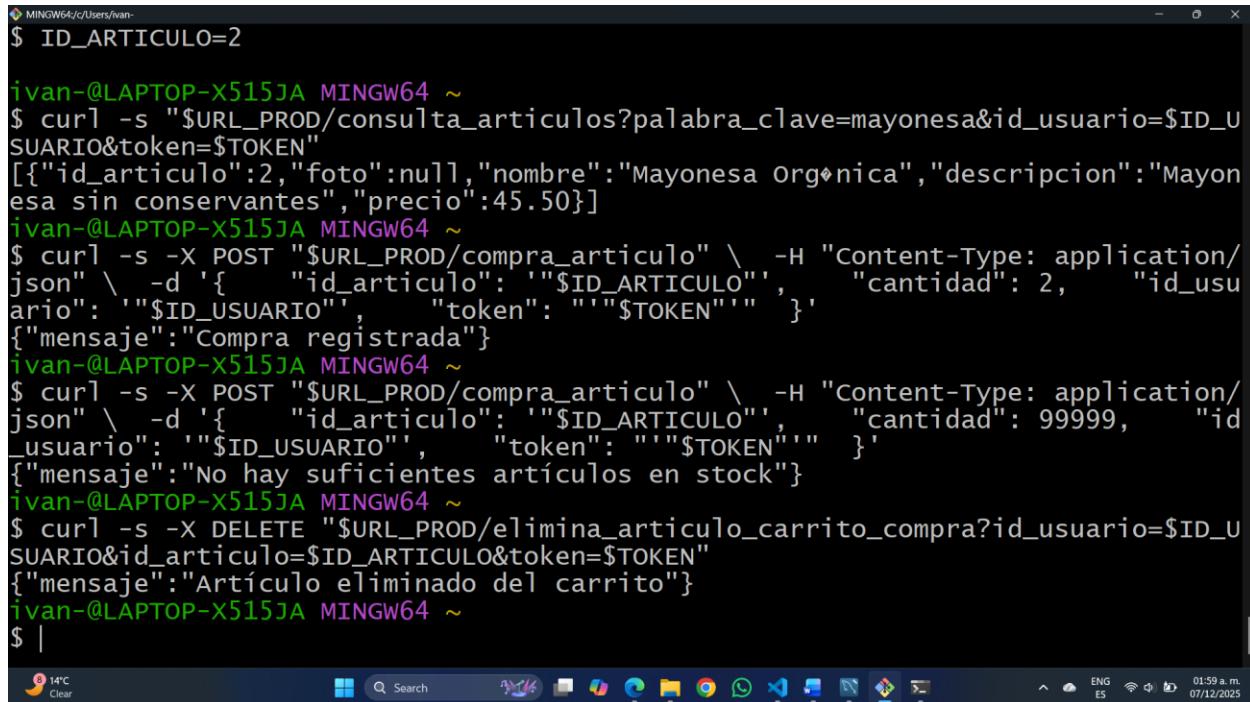
9.4.6 Requerimiento 4: elimina_articulo_carrito_compra

```

curl           -s           -X           DELETE
"$URL_PROD/elimina_articulo_carrito_compra?id_usuario=$ID_USUARIO&id_articulo
=$ID_ARTICULO&token=$TOKEN"

```

- Evidencia:



```
$ ID_ARTICULO=2
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s "$URL_PROD/consulta_articulos?palabra_clave=mayonesa&id_usuario=$ID_USUARIO&token=$TOKEN"
[{"id_articulo":2,"foto":null,"nombre":"Mayonesa Orgánica","descripcion":"Mayonesa sin conservantes","precio":45.50}]
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$URL_PROD/compra_articulo" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": "'"$ID_ARTICULO"'", "cantidad": 2, "id_usuario": "'"$ID_USUARIO"'", "token": "'"$TOKEN"'"}'
{"mensaje":"Compra registrada"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$URL_PROD/compra_articulo" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": "'"$ID_ARTICULO"'", "cantidad": 99999, "id_usuario": "'"$ID_USUARIO"'", "token": "'"$TOKEN"'"}'
{"mensaje":"No hay suficientes artículos en stock"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X DELETE "$URL_PROD/elimina_articulo_carrito_compra?id_usuario=$ID_USUARIO&id_articulo=$ID_ARTICULO&token=$TOKEN"
{"mensaje":"Artículo eliminado del carrito"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ |
```

Figura 26. Respuesta de eliminación de artículo del carrito

9.4.7 Requerimiento 5: elimina_carrito_compra

```
curl           -s           -X           DELETE
"$URL_PROD/elimina_carrito_compra?id_usuario=$ID_USUARIO&token=$TOKEN"
```

- Evidencia:

```

MINGW64/c/Users/ivan-
SUARIO&token=$TOKEN"
[{"id_articulo":2,"foto":null,"nombre":"Mayonesa Orgánica","descripcion":"Mayonesa sin conservantes","precio":45.50}]
ivan@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$URL_PROD/compra_articulo" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": "'.$ID_ARTICULO'", "cantidad": 2, "id_usuario": "'.$ID_USUARIO'", "token": "'.$TOKEN'"}'
{"mensaje":"Compra registrada"}
ivan@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$URL_PROD/compra_articulo" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": "'.$ID_ARTICULO'", "cantidad": 99999, "id_usuario": "'.$ID_USUARIO'", "token": "'.$TOKEN'"}'
{"mensaje":"No hay suficientes artículos en stock"}
ivan@LAPTOP-X515JA MINGW64 ~
$ curl -s -X DELETE "$URL_PROD/elimina_articulo_carrito_compra?id_usuario=$ID_USUARIO&id_articulo=$ID_ARTICULO&token=$TOKEN"
{"mensaje":"Artículo eliminado del carrito"}
ivan@LAPTOP-X515JA MINGW64 ~
$ curl -s -X DELETE "$URL_PROD/elimina_carrito_compra?id_usuario=$ID_USUARIO&token=$TOKEN"
{"mensaje":"Carrito eliminado"}
ivan@LAPTOP-X515JA MINGW64 ~
$
```

The screenshot shows a terminal window on a Windows 10 desktop. The taskbar at the bottom includes icons for File Explorer, Task View, Start, Search, Edge, and other system tools. The system tray shows the date (07/12/2025), time (01:59 a.m.), battery level (ENG ES), and signal strength.

Figura 27. Respuesta de eliminación completa del carrito

9.4.8 Requerimiento 6: modifica_carrito_compra (+1/-1)

- Prepara el carrito con una compra mínima:

```
curl -s -X POST "$URL_PROD/compra_articulo" \ -H "Content-Type: application/json" \
-d '{ "id_articulo": "'.$ID_ARTICULO'", "cantidad": 1, "id_usuario": "'.$ID_USUARIO'", \
"token": "'.$TOKEN'"}'
```

- Incremento (+1):

```
curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \ -H "Content-Type: application/json" \
-d '{ "id_articulo": "'.$ID_ARTICULO'", "incremento": 1, \
"id_usuario": "'.$ID_USUARIO'", "token": "'.$TOKEN'"}'
```

```

MINGW64/c/Users/ivan-
$ curl -s -d '{ "id_articulo": "'.$ID_ARTICULO'", "cantidad": 99999, "id_usuario": "'.$ID_USUARIO'", "token": "'.$TOKEN.'" }' -H "Content-Type: application/json" -X DELETE "$URL_PROD/elimina_articulo_carrito_compra?id_usuario=$ID_USUARIO&id_articulo=$ID_ARTICULO&token=$TOKEN"
{"mensaje":"No hay suficientes artículos en stock"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X DELETE "$URL_PROD/elimina_carrito_compra?id_usuario=$ID_USUARIO&token=$TOKEN"
{"mensaje":"Artículo eliminado del carrito"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X DELETE "$URL_PROD/elimina_carrito_compra?id_usuario=$ID_USUARIO&token=$TOKEN"
{"mensaje":"Carrito eliminado"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$URL_PROD/compra_articulo" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": "'.$ID_ARTICULO'", "cantidad": 2, "id_usuario": "'.$ID_USUARIO'", "token": "'.$TOKEN.'" }' -H "Content-Type: application/json"
{"mensaje":"Compra registrada"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": "'.$ID_ARTICULO'", "incremento": 1, "id_usuario": "'.$ID_USUARIO'", "token": "'.$TOKEN.'" }'
{"mensaje":"Se modificó la cantidad del carrito"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ |

```

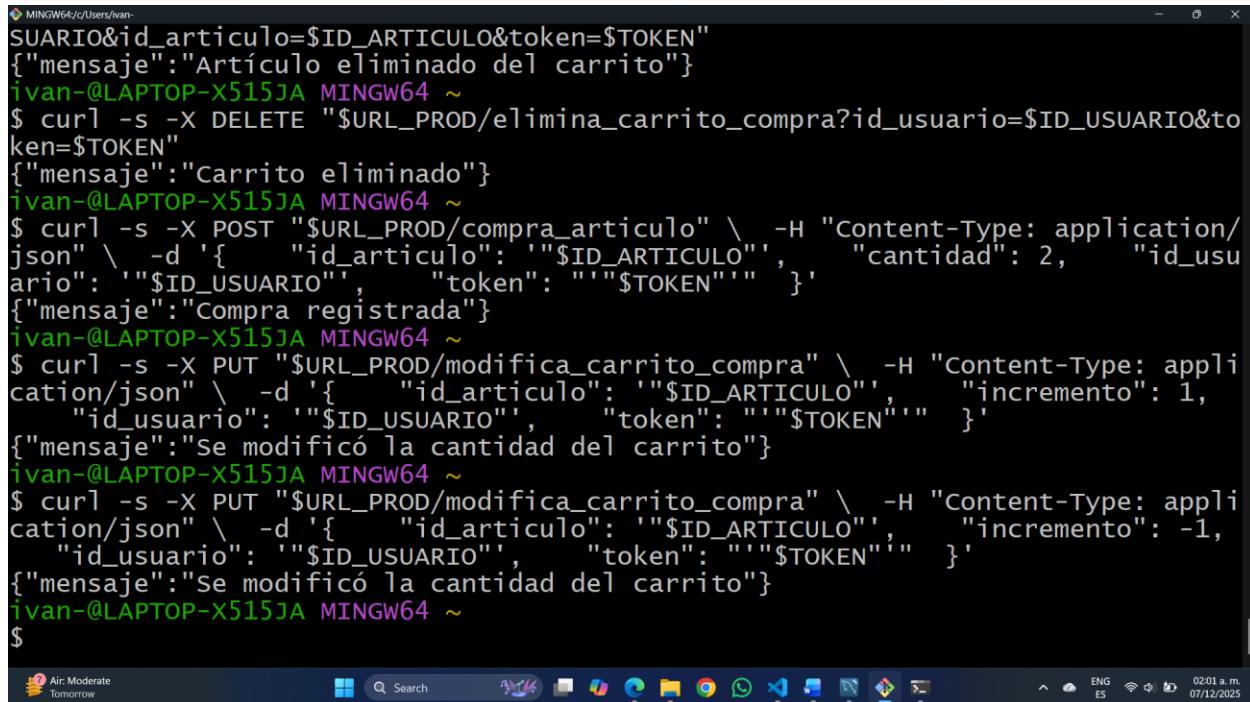
Figura 28. Respuesta de modificación +1

- Decremento (-1):

```

curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": "'.$ID_ARTICULO'", "incremento": -1, "id_usuario": "'.$ID_USUARIO'", "token": "'.$TOKEN.'" }'

```

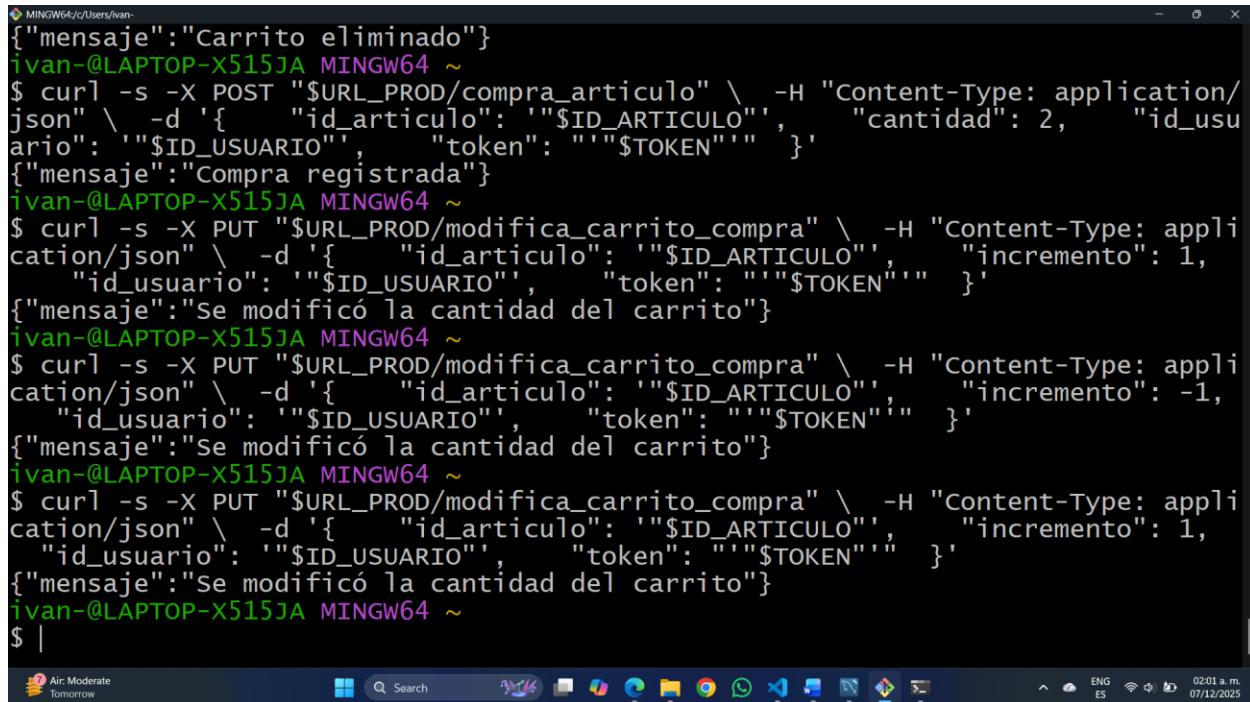


```
MINGW64/c/Users/ivan-
SUARIO&id_articulo=$ID_ARTICULO&token=$TOKEN"
{"mensaje":"Artículo eliminado del carrito"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X DELETE "$URL_PROD/elimina_carrito_compra?id_usuario=$ID_USUARIO&token=$TOKEN"
{"mensaje":"Carrito eliminado"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$URL_PROD/compra_articulo" \
-H "Content-Type: application/json" \
-d '{ "id_articulo": "'.$ID_ARTICULO.'", "cantidad": 2, "id_usuario": "'.$ID_USUARIO.'", "token": "'.$TOKEN.'" }'
{"mensaje":"Compra registrada"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \
-H "Content-Type: application/json" \
-d '{ "id_articulo": "'.$ID_ARTICULO.'", "incremento": 1, "id_usuario": "'.$ID_USUARIO.'", "token": "'.$TOKEN.'" }'
{"mensaje":"Se modificó la cantidad del carrito"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \
-H "Content-Type: application/json" \
-d '{ "id_articulo": "'.$ID_ARTICULO.'", "incremento": -1, "id_usuario": "'.$ID_USUARIO.'", "token": "'.$TOKEN.'" }'
{"mensaje":"Se modificó la cantidad del carrito"}
ivan-@LAPTOP-X515JA MINGW64 ~
$
```

Figura 29. Respuesta de modificación -1

- Error “No hay suficientes artículos en stock” (si stock = 0 y se intenta +1):

```
curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \
-H "Content-Type: application/json" \
-d '{ "id_articulo": "'.$ID_ARTICULO.'", "incremento": 1, "id_usuario": "'.$ID_USUARIO.'", "token": "'.$TOKEN.'" }'
```



```
MINGW64/c/Users/ivan-  
{"mensaje":"carrito eliminado"}  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ curl -s -X POST "$URL_PROD/compra_articulo" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": "'.$ID_ARTICULO'", "cantidad": 2, "id_usuario": "'.$ID_USUARIO'", "token": "'.$TOKEN' "' }'  
{"mensaje":"Compra registrada"}  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": "'.$ID_ARTICULO'", "incremento": 1, "id_usuario": "'.$ID_USUARIO'", "token": "'.$TOKEN' "' }'  
{"mensaje":"Se modificó la cantidad del carrito"}  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": "'.$ID_ARTICULO'", "incremento": -1, "id_usuario": "'.$ID_USUARIO'", "token": "'.$TOKEN' "' }'  
{"mensaje":"Se modificó la cantidad del carrito"}  
ivan-@LAPTOP-X515JA MINGW64 ~  
$ |  
Air: Moderate Tomorrow 02:01 a.m. 07/12/2025
```

Figura 30. Error “No hay suficientes artículos en stock”

- Error “No hay más artículos en el carrito” (si cantidad ya es 0 y se intenta -1):

```
curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": "'.$ID_ARTICULO'", "incremento": -1, "id_usuario": "'.$ID_USUARIO'", "token": "'.$TOKEN' "' }'
```

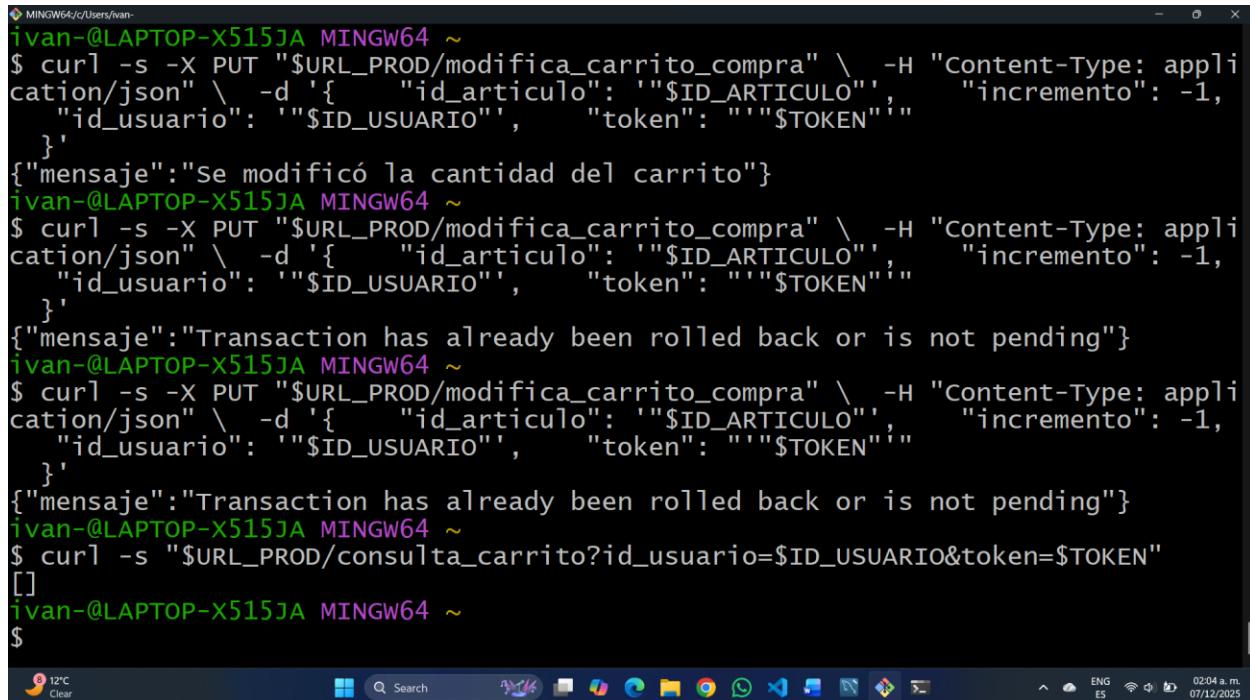
```
MINGW64/c/Users/ivan-
  "id_usuario": '$ID_USUARIO',    "token": "'$TOKEN'"
}'
{"mensaje":"Se modificó la cantidad del carrito"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \ -H "Content-Type: application/json" \ -d '{   "id_articulo": '$ID_ARTICULO',   "incremento": -1,
  "id_usuario": '$ID_USUARIO',    "token": "'$TOKEN'"
}'
{"mensaje":"Se modificó la cantidad del carrito"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \ -H "Content-Type: application/json" \ -d '{   "id_articulo": '$ID_ARTICULO',   "incremento": -1,
  "id_usuario": '$ID_USUARIO',    "token": "'$TOKEN'"
}'
 {"mensaje":"Transaction has already been rolled back or is not pending"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \ -H "Content-Type: application/json" \ -d '{   "id_articulo": '$ID_ARTICULO',   "incremento": -1,
  "id_usuario": '$ID_USUARIO',    "token": "'$TOKEN'"
}'
 {"mensaje":"Transaction has already been rolled back or is not pending"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ |
```

- **Figura 31. Error “No hay más artículos en el carrito”**

9.4.9 Función de apoyo: consulta_carrito

```
curl -s "$URL_PROD/consulta_carrito?id_usuario=$ID_USUARIO&token=$TOKEN"
```

- Evidencia:



```
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": "'.$ID_ARTICULO'", "incremento": -1, "id_usuario": "'.$ID_USUARIO'", "token": "'.$TOKEN'"}'
{"mensaje":"Se modificó la cantidad del carrito"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": "'.$ID_ARTICULO'", "incremento": -1, "id_usuario": "'.$ID_USUARIO'", "token": "'.$TOKEN'"}'
{"mensaje":"Transaction has already been rolled back or is not pending"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": "'.$ID_ARTICULO'", "incremento": -1, "id_usuario": "'.$ID_USUARIO'", "token": "'.$TOKEN'"}'
{"mensaje":"Transaction has already been rolled back or is not pending"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s "$URL_PROD/consulta_carrito?id_usuario=$ID_USUARIO&token=$TOKEN"
[]
ivan-@LAPTOP-X515JA MINGW64 ~
$
```

Figura 32. Respuesta JSON del carrito con artículos y cantidades

9.4.10 Checkout opcional: finaliza_compra (“Comprar todo”)

```
curl           -s           -X           POST
"$URL_PROD/finaliza_compra?id_usuario=$ID_USUARIO&token=$TOKEN" \   -H
"Content-Type: application/json" \ -d "{}"
```

- Evidencias:

```
MINGW64/c/Users/ivan-
"id_usuario": '$ID_USUARIO', "token": '$TOKEN'
}'
{"mensaje": "Transaction has already been rolled back or is not pending"}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": '$ID_ARTICULO', "incremento": -1, "id_usuario": '$ID_USUARIO', "token": '$TOKEN' }'
{
"mensaje": "Transaction has already been rolled back or is not pending"
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s "$URL_PROD/consulta_carrito?id_usuario=$ID_USUARIO&token=$TOKEN"
[]
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$URL_PROD/compra_articulo" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": '$ID_ARTICULO', "cantidad": 2, "id_usuario": '$ID_USUARIO', "token": '$TOKEN' }'
{
"mensaje": "Compra registrada"
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$URL_PROD/finaliza_compra?id_usuario=$ID_USUARIO&token=$TOKEN" \ -H "Content-Type: application/json" \ -d '{}'
{
"id_orden": 2, "total": 91.00
}
ivan-@LAPTOP-X515JA MINGW64 ~
$
```

Figura 33. Respuesta JSON con id_orden y total de la compra

```
MINGW64/c/Users/ivan-
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X PUT "$URL_PROD/modifica_carrito_compra" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": '$ID_ARTICULO', "incremento": -1, "id_usuario": '$ID_USUARIO', "token": '$TOKEN' }'
{
"mensaje": "Transaction has already been rolled back or is not pending"
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s "$URL_PROD/consulta_carrito?id_usuario=$ID_USUARIO&token=$TOKEN"
[]
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$URL_PROD/compra_articulo" \ -H "Content-Type: application/json" \ -d '{ "id_articulo": '$ID_ARTICULO', "cantidad": 2, "id_usuario": '$ID_USUARIO', "token": '$TOKEN' }'
{
"mensaje": "Compra registrada"
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s -X POST "$URL_PROD/finaliza_compra?id_usuario=$ID_USUARIO&token=$TOKEN" \ -H "Content-Type: application/json" \ -d '{}'
{
"id_orden": 2, "total": 91.00
}
ivan-@LAPTOP-X515JA MINGW64 ~
$ curl -s "$URL_PROD/consulta_carrito?id_usuario=$ID_USUARIO&token=$TOKEN"
[]
ivan-@LAPTOP-X515JA MINGW64 ~
$
```

Figura 34. Carrito vacío después del checkout (consulta_carrito)

10 Conclusiones

Se realizó el desarrollo y despliegue de un prototipo serverless de comercio electrónico, integrando Azure Functions con MySQL PaaS y un front-end HTML/JavaScript servido desde Azure Files. Se accedió al conjunto de servicios en la región Canada, aplicando la nomenclatura obligatoria y garantizando consistencia en las pruebas locales y móviles. El uso de transacciones en operaciones críticas como la compra, eliminación y modificación del carrito demostró ser fundamental para la integridad de datos, mientras que la verificación de acceso por token aseguró el control de seguridad.

La utilización de inteligencia artificial de GitHub Copilot aceleró la creación de plantillas de código, la estandarización de respuestas y la organización del proyecto, permitiendo concentrar esfuerzos en las reglas de negocio y la correcta orquestación entre front-end y back-end. Se accedió a las evidencias solicitadas y se instalaron configuraciones clave (variables y montajes) que facilitaron el funcionamiento en producción.

Enlace del chat de la IA generativa

Enlace: <https://github.com/copilot/share/421c1186-0264-88a1-a002-2047003800c2>

11 Referencias (Formato IEEE)

- [1] R. T. Fielding, "Architectural Styles and the Design of Network-based Software Architectures," Ph.D. dissertation, Univ. California, Irvine, 2000.
- [2] Oracle, "Java Platform Standard Edition 17 API Specification – Class SecureRandom," 2025. [Online]. Available: <https://docs.oracle.com/>
- [3] ISO/IEC 9075-1:2016, "Information technology — Database languages — SQL — Part 1: Framework (SQL/Framework)."
- [4] Oracle, "InnoDB Storage Engine," MySQL 8.0 Reference Manual, 2025. [Online]. Available: <https://dev.mysql.com/doc/>
- [5] Apache Software Foundation, "Apache Tomcat 10 Documentation," 2025. [Online]. Available: <https://tomcat.apache.org/>
- [6] Eclipse Foundation, "Jakarta RESTful Web Services (JAX-RS) Specification," 2025. [Online]. Available: <https://jakarta.ee/specifications/>
- [7] FasterXML, "Jackson Databind and Core Libraries," 2025. [Online]. Available: <https://github.com/FasterXML/jackson>
- [8] GitHub, "GitHub Copilot: AI Pair Programmer," 2025. [Online]. Available: <https://github.com/features/copilot>
- [9] NIST, "Secure Hash Standard (SHS)," FIPS PUB 180-4, Aug. 2015.
- [10] W3C, "File API," W3C Recommendation, 2015. [Online]. Available: <https://www.w3.org/TR/FileAPI/>
- [11] W3C, "Web Cryptography API," W3C Recommendation, 2017. [Online]. Available: <https://www.w3.org/TR/WebCryptoAPI/>
- [12] OWASP Foundation, "OWASP Top 10: Web Application Security Risks," 2023. [Online]. Available: <https://owasp.org/>
- [13] D. Thomas et al., "Bcrypt Password Hashing," OpenBSD Project Documentation, 2025. [Online]. Available: <https://man.openbsd.org/>
- [14] P. Wuille et al., "Argon2: The Memory-Hard Function for Password Hashing," RFC Draft (argon2), 2025.
- [15] Google, "Fetch API Living Standard," WHATWG, 2025. [Online]. Available: <https://fetch.spec.whatwg.org/>

- [16] Mozilla, “Content Security Policy (CSP) Guide,” MDN Web Docs, 2025. [Online]. Available: <https://developer.mozilla.org/>
- [17] Git, “Distributed Version Control System,” Git Project, 2025. [Online]. Available: <https://git-scm.com/>
- [18] Azure, “Azure Virtual Machines Documentation,” Microsoft Learn, 2025. [Online]. Available: <https://learn.microsoft.com/azure/virtual-machines/>