**SUPERMERCADOS FRUVER**

# Development Solution

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Rol** | **Nombres y apellidos** | **No. de identificación** | **E-mail** | **Grupo** |
| **Líder del equipo, Scrum master** | David Baena Castro | CC 1037636841 | [david16baena@gmail.com](mailto:david16baena@gmail.com) | 33-34 |
| **Diseñador de software** | Juan Pablo Cardona Torres | CC 1214713041 | [juan-c16@hotmail.com](mailto:juan-c16@hotmail.com) | 33-34 |
| **Diseñador UI** | Andrea Isabel Pérez Días | CC 1014244261 | [aisabelperezd14@gmail.com](mailto:aisabelperezd14@gmail.com) | 33-34 |
| **Tester** | Wilmar Arley Zapata Villa | CC 15446089 | [carewaz23@hotmail.com](mailto:carewaz23@hotmail.com) | 33-34 |
| **Administrador configuración** | Geraldine Álvarez Chabur | CC 1003845186 | [geraldinealvarez041@gmail.com](mailto:geraldinealvarez041@gmail.com) | 33-34 |
| **URL github** | https://github.com/Development-Solution/Supermercado.git | | | |

# Aplicación

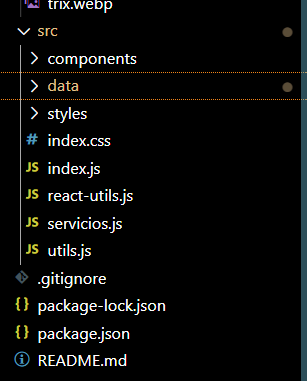
Estimado cliente en esta segunda entrega hemos desarrollado los servicios que tienen la lógica interna de la página. Con los diferentes servicios usted, **como administrador,** podrá: **listar todos los productos**, **agregar productos**, **modificar productos** o **agregar unidades de algún producto**. De igual manera, **como usuario**, podrá: **listar todos los productos disponibles**, **seleccionar un subconjunto de los productos**, **recibir un resumen de cuánto le costaría teniendo en cuenta los productos que hay en stock**, **terminar la compra modificando el stock de productos actual** y **actualizando la tabla de ventas del administrador**.

# Estructura de los archivos que contienen el código fuente

Al usted abrir el código de la carpeta apreciará que en el código fuente se adicionaron los siguientes archivos:

1. Un archivo *utils.js* que contiene toda la lógica detrás de los servicios, que modifica los archivos json donde está almacenada la información
2. Un archivo *servicios.js* que tiene la implementación de servidor en *Express*, así como la implementación de todos los servicios que se llamarán por el API
3. Un archivo *react-utils.js* que contiene una clase con las funciones para hacer peticiones get y peticiones post desde *React*

**Imagen 1. Estructura carpetas**

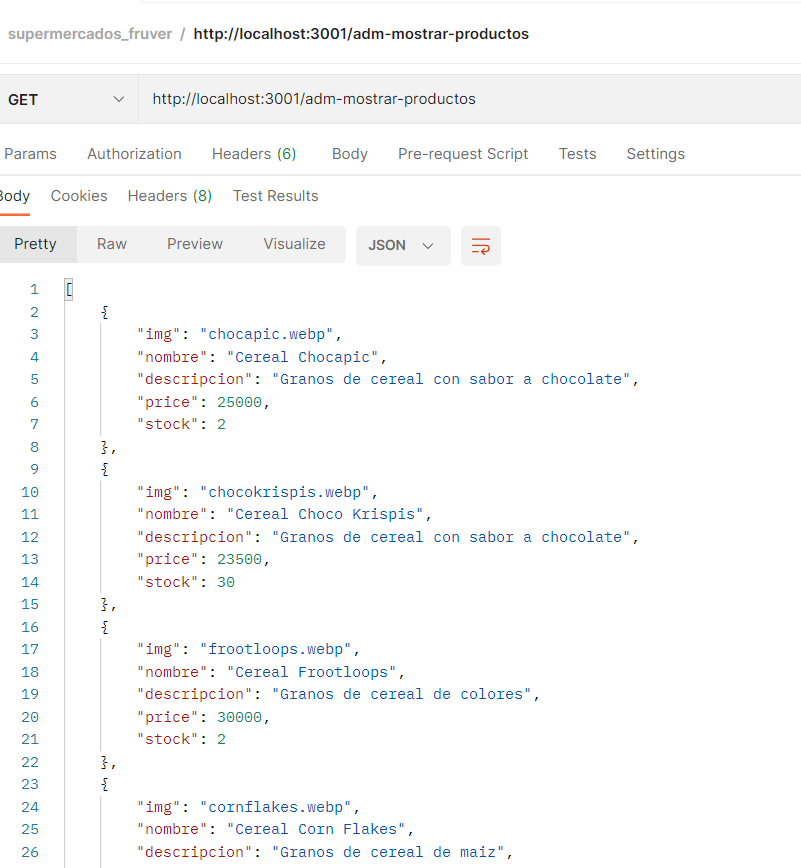


# Servicios

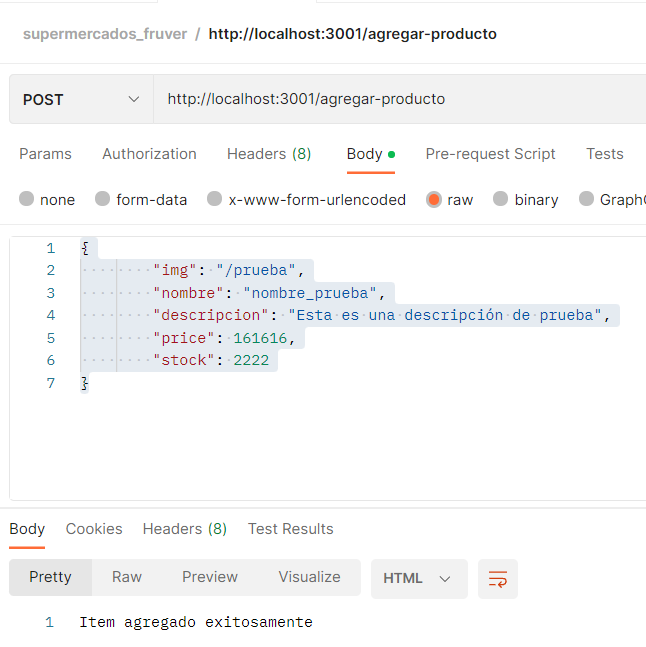
Para implementar los servicios como API Rest se usó Express. Note que si se para en la carpeta del proyecto y escribe *node src/servicios.js*

Le aparecerá un mensaje que le indicará lo siguiente *La app está escuchando en el puerto 3001!* Con lo anterior, podemos probar los servicios en Postman, como se ilustrará a continuación:

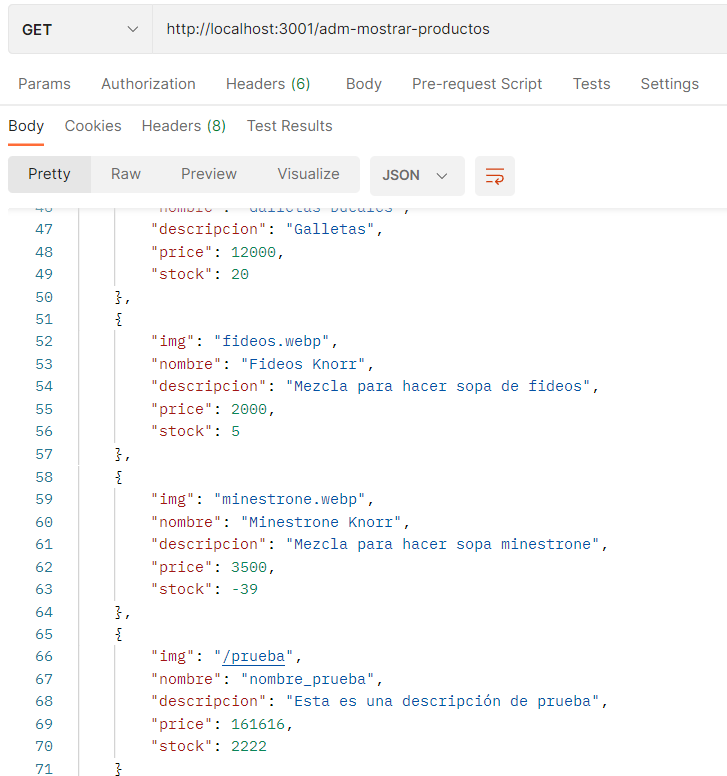
*GET* [*http://localhost:3001/adm-mostrar-productos*](http://localhost:3001/adm-mostrar-productos)Este servicio enlista los productos que están disponibles (aunque el stock sea nulo o negativo aparecerá)



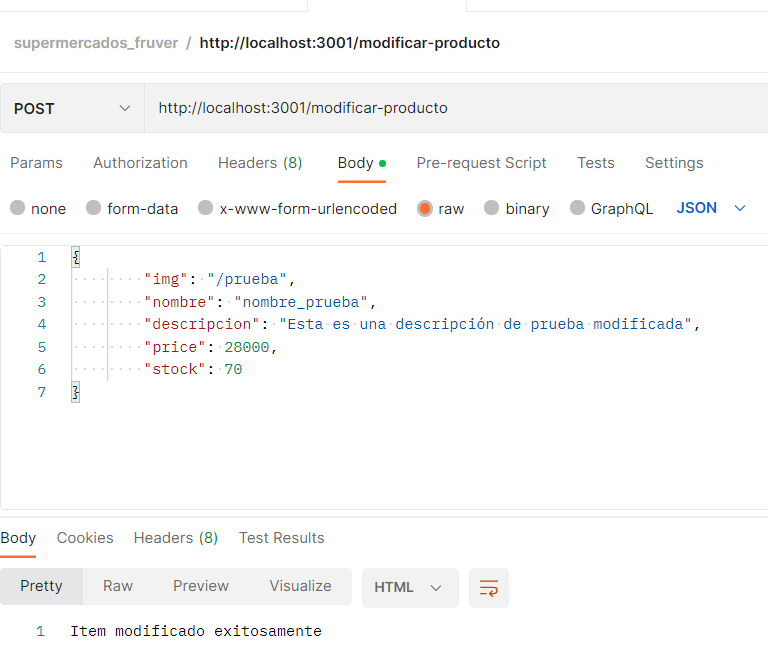
*POST http://localhost:3001/agregar-producto* Este servicio agregará al listado anterior el cuerpo que se envíe en el request



Note que efectivamente el producto “*nombre\_prueba*” fue agregado



*POST http://localhost:3001/modificar-producto* Este servicio modificará el ítem que se llame igual al ítem que se envíe en el cuerpo del request, y sus atributos se sobrescribirán tal como indique en el request

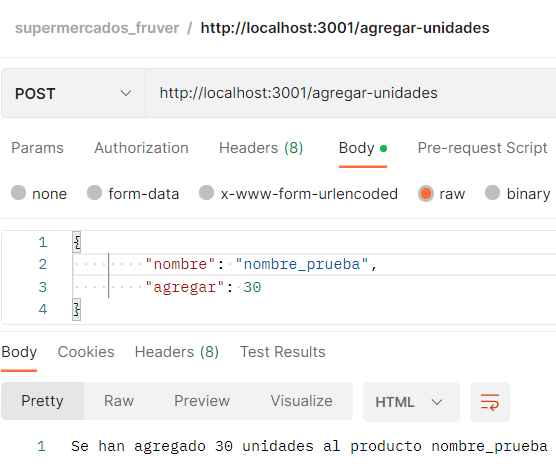
**

Note que efectivamente el producto “*nombre\_prueba*” fue modificado

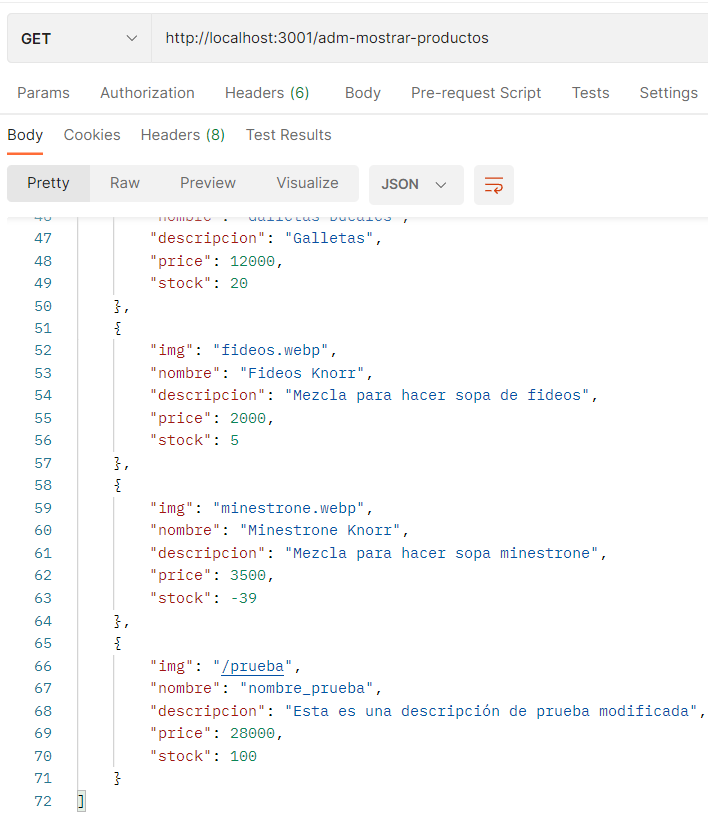
Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

*POST http://localhost:3001/agregar-unidades* Este servicio agregará al ítem que se llame igual al ítem que se envíe en el cuerpo del request la cantidad de stock que se especifique en su cuerpo



Note que efectivamente el producto “*nombre\_prueba*” tiene 30 unidades más, ahora tiene 100 unidades



*GET* [*http://localhost:3001/usr-mostrar-productos*](http://localhost:3001/usr-mostrar-productos)Este servicio enlista los productos que están disponibles (ítems con stock nulo o negativo NO aparecerán)

*Interfaz de usuario gráfica, Texto

Descripción generada automáticamente*

*POST http://localhost:3001/seleccionar-carrito* Este servicio seleccionará los ítems de todo el inventario disponible y mostrará en stock el mínimo entre la cantidad disponible en inventario y la cantidad que se requirió en el request

*Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

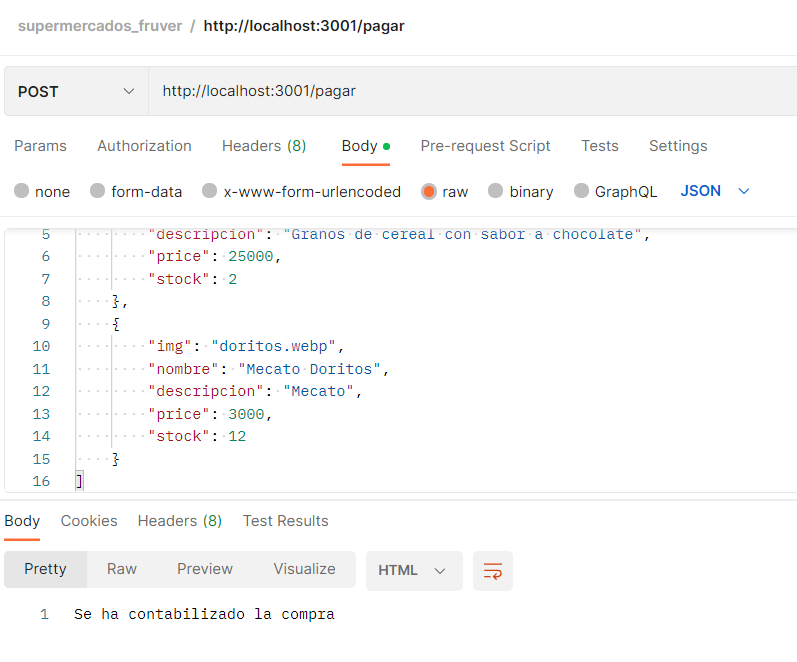
Descripción generada automáticamente*

*POST* [*http://localhost:3001/finalizar-compra*](http://localhost:3001/finalizar-compra)Este servicio recibe la respuesta del request al endpoint *http://localhost:3001/seleccionar-carrito* y mostrará el total así como un resumen de la compra, y la fecha en que ésta fue hecha

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

*POST* [*http://localhost:3001/pagar*](http://localhost:3001/pagar)Este servicio recibe la respuesta del request al endpoint *http://localhost:3001/seleccionar-carrito* y contabilizará la compra y finalmente afectará los inventarios que se pueden observar usando los servicios GET



**SE DEJARÁN TODOS ESTOS SERVICIOS EN UN POST-COLLECTION PARA QUE PUEDAN JUGAR CON ELLOS.**