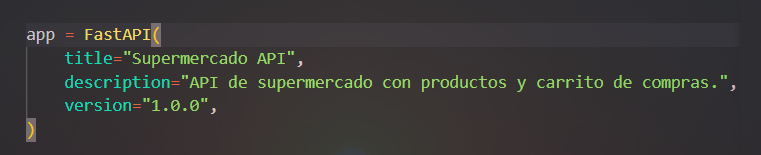
**Documentación del código:**

1. **Explicación general del proyecto**

* Que es el proyecto: es una página web de registro de pedidos de una cafetería.
* Tecnologías principales: FastAPI para backend, SQLite como base de datos, tailwind y Streamlit para el frontend

1. Main.py:



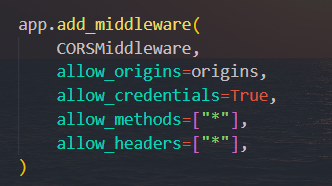
**Inicialización de la aplicación**

* Se crea una instancia de FastAPI, asignando un titulo, descripción y versión de la API.
* Esta instancia(app) será utilizada por el servidor para ejecutar y exponer los endpoints.



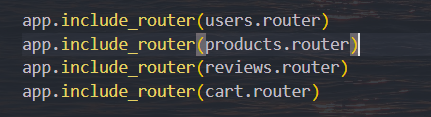
**Creación de tablas en base de datos:**

* Usa SQLAlchemy para crear automáticamente todas las tablas definidas en app/models/ si no existen en la base de datos

**Configuración de CORS:**

* Se agrega el middleware CORS para permitir peticiones desde cualquier origen (\*) lo cual facilita el acceso desde el fronted o cliente externos.
* Se eprmite el uso de credenciales y cualquier tipo de método HTTP y cabecera.

**Inclusión de Routers:**



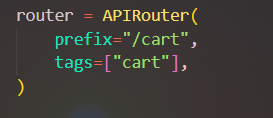
organiza los endpoints en distintos módulos (routers):

* Users: gestion de usuarios
* Products: gestión de productos
* Reviews: reseñas o comentarios de productos.
* Cart: operaciones de carrito de compras.

1. **app/routers/:**

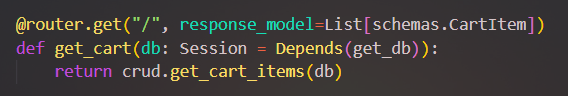
cart.py: este archivo define las rutas para gestionar el carrito de compras dentro de la API. Utiliza FastAPI junto con SQLAlchemy para manejar operaciones CRUD sobre la base de datos.

las configuraciones del router:

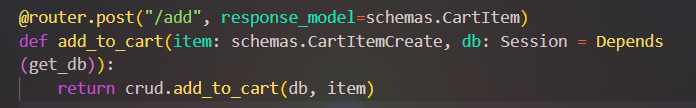


* todas las rutas estarán bajo el prefijo /cart

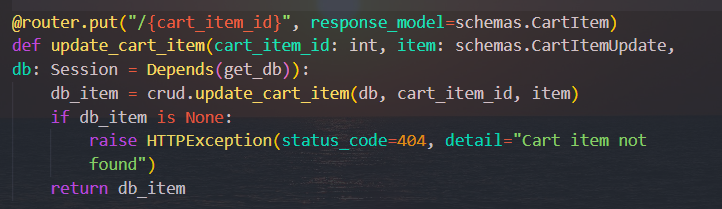
**obtener todos los ítems del carrito**



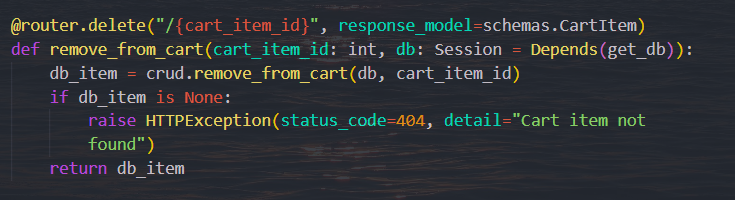
* método: GET/cart/
* descripción: devuelve todos los ítems que se encuentran actualmente en el carrito.
* Respuesta: lista de objetos CartItem.

**Agregar un ítem al carrito:  
**

* Método: POST/cart/add
* Agrega un producto al carrito.
* El cuerpo de la petición, objeto CartItemCreate con la información del producto.
* La respuesya es el item agregado (CartItem)

**Actualizar un ítem del carrito:**

* Metodo: PUT/cart/{cart\_item\_id}
* Actualiza la cantidad u otro datos de un item existente en el carrito
* Parámetro de ruta: cart\_item\_id
* La respuesta el item actualiza
* Errores posibles: 404 Not Found si el item no existe.

**Eliminar un ítem del carrito:**

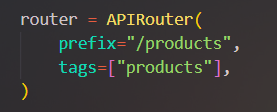
**-**metodo: DELETE/cart/{cart\_item\_id}

-elimina un item de carrito

-errores posibles: 404 Not Found si el item no existe

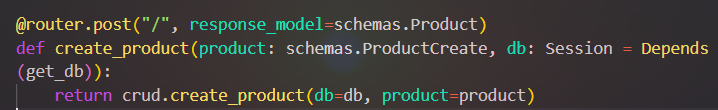
Products.py:

**Configuración de router:**



* Todas las rutas estarán bajo el prefijo /products
* En Swagger aparecerá agrupadas bajo la etiqueta products.

Endpoints:

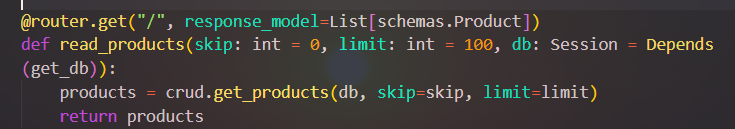
**Crear un producto:**

-Metodo: POST/products/

- crea un nuevo producto en la base de datos.

-el cuerpo de la petición es objeto producCreate con los datos del producto

-respuesta el producto creado (product)

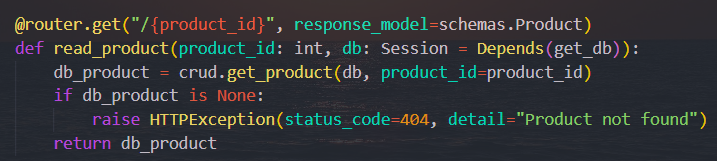
**Obtener todos los productos:**

-Método: GET/products/

-descripcion: devuelve la lista de productos disponibles

-respuesta es lista de objetos producto

**Obtener un producto por ID:**



-Método: GET/products/{producto\_id}

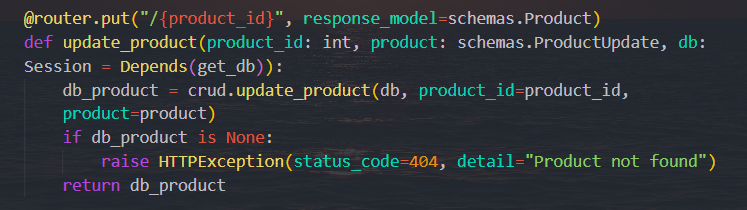
-devuelve un producto especifico por su ID

-parametros de la ruta: producto\_id(int)

-respuesta es un objeto producto

-errores posibles: 404 Not Found si el producto no existe.

**Actualizar un producto:**



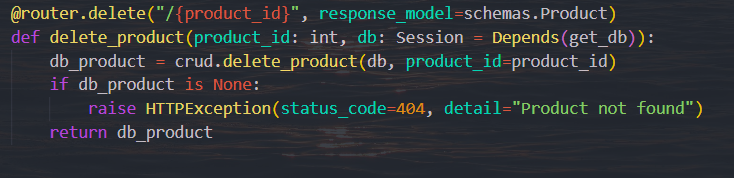
-Metodo: PUT/products/{producto\_id}

-actualizar los datos de un producto existente

-parámetros de ruta: producto\_id (int).

-Respuesta es producto actualizado

**Eliminar un producto:**

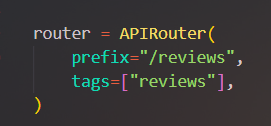
****

-Metodo: DELETE/products/{producto\_id}

-elimina un producto de la base de datos.

-respuesta producto eliminado

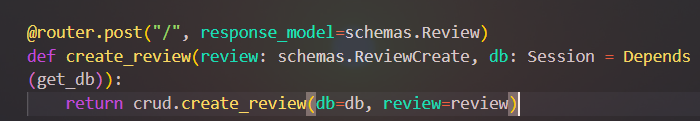
-errores posibles: 404 Not Found si el producto no existe

Reviews.py:  
configuración de router

- Todas las rutas estarán bajo el prefijo /reviews.

- En la documentación de Swagger aparecerán agrupadas bajo la etiqueta **reviews**

**Crear una reseña**

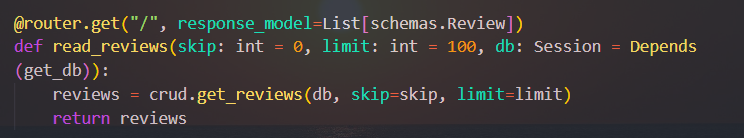


- Método: POST /reviews/

- Descripción: Crea una nueva reseña asociada a un producto.

- Cuerpo de la petición: Objeto ReviewCreate (ejemplo: producto\_id, usuario\_id, comentario, calificación).

- Respuesta: La reseña creada (Review).

**Obtener todas las reseñas:**

Método: GET /reviews/

Descripción: Devuelve la lista de todas las reseñas registradas.

Respuesta: Lista de objetos Review.