

# Documentación de los escenarios de prueba

Last updated by | Juan Andres Moreno | 6 nov 2025 at 15:01 GMT-5

## Documentación de los escenarios de prueba:

### 1. Crear orden manual:

POST `http://127.0.0.1:8000/api/orders/create/`

#### Body (JSON):

```
{ "product": "Botas dieléctricas Provasi 10 kV", "units": 3, "lat": 4.62, "lon": -74.08 }
```

Crea una orden simple; si no hay stock, la orden se marca como REJECTED.

### 2. Crear orden con selección automática (ASR):

POST `http://127.0.0.1:8000/api/auto_order/`

#### Body (JSON):

```
{ "product": "Botas dieléctricas Provasi 10 kV", "units": 3, "lat": 4.62, "lon": -74.08, "mainwarehouse":  
"Bodega Sur" }
```

Verifica disponibilidad en todas las bodegas, selecciona la más cercana con stock y actualiza inventario.

#### Respuesta exitosa:

```
{ "order_id": 2, "product": "Botas dieléctricas Provasi 10 kV", "units": 3, "status": "CONFIRMED",  
"assigned_warehouse": "Bodega Sur", "confirmed": true, "execution_time_seconds": 0.11, "meets_performance_ASR":  
true }
```

## Ubicación de las rutas en el proyecto

Las rutas se encuentran `orders/urls.py`:

```
from django.urls import path from . import views urlpatterns = [ path('orders/create/', views.create_order,  
name='create_order'), path('auto_order/', views.auto_order, name='auto_order'), ]
```

## Funciones que manejan las rutas

En `orders/views.py`:

- `create_order(request)` → crea una orden básica.
- `auto_order(request)` → implementa el ASR: selecciona la bodega más cercana con stock, confirma la orden y descuenta inventario.

## Lógica del ASR y manejo del inventario

La lógica principal del escenario probado se encuentra en `orders/logic.py` , donde se definen las funciones que implementan el comportamiento del ASR.

El código utiliza transacciones atómicas ( `transaction.atomic()` ) para garantizar consistencia en la base de datos y la fórmula de **Haversine** para calcular la distancia entre el cliente y las bodegas registradas.

### Funciones principales:

- `find_nearest_with_stock()` → busca la bodega más cercana con stock suficiente.
- `place_order_atomic()` → realiza la orden completa:
  - Verifica stock en la bodega principal.
  - Si no hay disponibilidad, selecciona automáticamente la más cercana.
  - Si ninguna tiene stock, la orden se marca como **REJECTED**.
  - Todo el proceso ocurre en menos de 5 s y de forma segura (transacción atómica).

El cálculo de distancia usa la función `haversine_km(lon1, lat1, lon2, lat2)` , basada en trigonometría esférica.

---

## Ejecución del entorno

El entorno del proyecto se ejecuta dentro de un entorno virtual de Django creado con `venv` . Para correr la aplicación, se activa el entorno con:

```
venv\Scripts\activate
```

y luego se inicia el servidor con:

```
python manage.py runserver
```

Desde el navegador se accede al panel de administración en <http://127.0.0.1:8000/admin/> , donde se gestionan los productos, bodegas e inventarios, y se validan las órdenes generadas mediante las pruebas en Postman.