Evaluación Sumativa 4 – Tienda Online con API y Deploy

Asignatura	Programación Back End
Unidad III	API REST, Reportes y Deploy (continuidad Evaluación 3)
Criterios a Evaluar	Reporte del sistema, Deploy, APIs REST (DRF) y evaluación individual
Modalidad	Trabajo en parejas (mismo proyecto y puntaje por grupo) + evaluación individual (puntaje asociado)
Duración estimada	240 minutos (evaluación en clases + verificación del deploy y APIs)
Ponderación	Según programa de asignatura

Instrucciones Generales:

• La nota 4,0 se obtiene logrando al menos el 60% del puntaje total (60 puntos de 100).

• Esta prueba consiste en continuar y mejorar el sistema desarrollado en la Evaluación 3. No se acepta un proyecto nuevo.

• Utilice Visual Studio Code, Python 3 y Django según las versiones trabajadas en clases.

• Se permite utilizar documentación oficial de Django y Django REST Framework, además de recursos vinculados desde la documentación oficial.

• El proyecto debe ejecutar correctamente: python manage.py migrate y python manage.py runserver (entorno local).

• El trabajo se entrega en un repositorio público de GitHub e incluye URL pública del deploy y evidencias solicitadas en el README.

• Está permitido el uso de IA generativa hasta un 50% del trabajo, declarando su uso en el Anexo de Declaración de IA.

Caso: Tienda de Artículos Personalizados – Extensión con API, Reportes y Deploy

Este proyecto consiste en continuar y mejorar el sistema desarrollado en la Evaluación 3. Cada grupo debe trabajar sobre su proyecto existente, ampliando sus funcionalidades e incorporando

herramientas adicionales que permitan que el sistema evolucione hacia una solución más completa y cercana a un entorno real de producción.

Objetivos de la Extensión

El sistema debe incorporar:

• Un reporte dinámico basado en datos reales almacenados en la base de datos.

• APIs REST implementadas con Django REST Framework (DRF), siguiendo restricciones específicas por endpoint.

• Deploy funcional del sistema en un hosting online gratuito (URL pública).

• Evaluación individual en clases para verificar dominio del proyecto y capacidad de modificación.

Requerimientos del Proyecto

1. Reporte del Sistema (20 pts)

Cada grupo debe crear un reporte dinámico que entregue información relevante del sistema. Algunas opciones:

• Cantidad de pedidos por estado.

• Productos más solicitados.

• Pedidos agrupados por plataforma (Web, Instagram, WhatsApp, etc.).

• Cualquier otra métrica útil para el negocio, coherente con el modelo de datos del proyecto.

El reporte debe cumplir como mínimo con lo siguiente:

• Usar datos reales almacenados en la base (no datos simulados ni hardcodeados).

• Ser accesible desde una vista protegida (por ejemplo: login requerido y/o permisos).

• Permitir personalización del resultado (por ejemplo: rango de fechas, selección de estado, plataforma, etc.).

• Presentar resultados de forma clara: tabla + al menos un gráfico (por ejemplo: barras, torta, líneas).

2. Deploy del Sistema (20 pts)

El sistema debe ser publicado en un servicio gratuito, por ejemplo:

• Render

• Railway

• PythonAnywhere

• Vercel

• Otro servicio adecuado (previa validación técnica de que soporte Django/DB)

Requisitos mínimos verificables:

• El sistema debe ser accesible públicamente mediante una URL.

• Deben funcionar las rutas principales: catálogo, detalle de producto, formulario de solicitud, página de seguimiento, administrador.

• Deben funcionar las APIs solicitadas (rutas publicadas y operativas).

• El repositorio debe incluir un README con instrucciones de deploy (pasos y configuración).

3. APIs REST con Django REST Framework (30 pts)

Se deben implementar tres APIs, todas utilizando Django REST Framework. Se sugiere incluir evidencias en el README (capturas, Postman, curl).

API 1 – CRUD de Insumos (5 pts)

Ruta sugerida: /api/insumos/

Debe permitir: Crear, Listar, Ver detalle, Modificar, Eliminar.

API 2 – Crear y modificar pedidos (10 pts)

Ruta sugerida: /api/pedidos/

Debe permitir: Crear pedidos vía JSON y Modificar pedidos existentes.

Restricciones obligatorias:

• No permitir listar pedidos (GET a colección).

• No permitir eliminar pedidos (DELETE).

API 3 – Filtro de pedidos por parámetros (15 pts)

Ruta sugerida: /api/pedidos/filtrar/

Debe recibir (por query params o body, según diseño consistente):

• Rango de fechas

• Estados

• Cantidad máxima de resultados

Debe retornar solo los pedidos que cumplan con los filtros, con validación de entradas y respuesta consistente (JSON).

4. Evaluación Individual (30 pts)

Cada estudiante será evaluado sobre su dominio del proyecto mediante:

• Interrogación técnica del sistema, APIs, deploy y reporte (decisiones, configuración, restricciones).

• Implementación práctica de modificaciones solicitadas en sala (cambios acotados y verificables).

La evaluación individual se aplicará dentro del tiempo de la clase y se considerará evidencia funcional del cambio solicitado.

Requisitos Técnicos

N°	Elemento o Configuración Solicitada	Descripción Detallada / Criterio de Cumplimiento	Verificación (✓)
1	Continuidad del proyecto ES3	Se trabaja sobre el mismo repositorio/proyecto de la Evaluación 3. Se conserva el catálogo, formulario, seguimiento y admin ya implementados.	☐

2	Repositorio GitHub público	Repositorio accesible públicamente, con commits y estructura ordenada del proyecto.	☐
3	README completo	Incluye: instalación local, configuración, credenciales de prueba (si corresponde), URL de deploy, endpoints API y guía de deploy (pasos y variables).	☐
4	Vista de reporte protegida	Reporte accesible solo con autenticación (login_required, permisos o grupos). Debe existir ruta clara (ej.: /reporte/).	☐
5	Reporte con datos reales (ORM)	El reporte consulta datos de la base usando ORM/aggregations. No se aceptan métricas hardcodeadas.	☐
6	Personalización del reporte	Permite filtrar o parametrizar (rango de fechas y/o estado y/o plataforma). La personalización impacta el resultado mostrado.	☐
7	Tabla + gráfico en reporte	El reporte presenta tabla y al menos un gráfico (Chart.js u otra librería equivalente).	☐
8	DRF instalado y configurado	djangorestframework agregado a INSTALLED_APPS, configuración base y urls de /api/ definidas.	☐
9	API1 Insumos CRUD	Endpoints completos (list/create/retrieve/update/destroy) funcionales, con serializer y validaciones básicas.	☐
10	API2 Pedidos create/update	Permite POST y PUT/PATCH. Bloquea LIST (GET colección) y DELETE. Responde con códigos HTTP correctos.	☐
11	API3 Filtrar pedidos	Endpoint recibe parámetros (fechas, estado(s), máximo resultados). Aplica filtros y retorna solo coincidencias.	☐
12	Validación de parámetros API3	Maneja fechas inválidas, estados inexistentes, max_resultados fuera de rango (respuesta 400 con mensaje).	☐
13	Evidencia de consumo de APIs	README incluye ejemplos: curl/Postman y/o capturas de respuesta JSON por cada API.	☐
14	Deploy con URL pública	La aplicación está publicada y accesible desde internet con URL funcional.	☐
15	Rutas web principales operativas	En producción funcionan catálogo, detalle, formulario solicitud, seguimiento y admin.	☐

16	APIs operativas en producción	Las rutas /api/insumos/, /api/pedidos/ y /api/pedidos/filtrar/ responden desde el deploy.	☐
17	Configuración mínima de producción	DEBUG=False, ALLOWED_HOSTS correcto y variables sensibles en entorno (no hardcodeadas en repo).	☐
18	Archivos estáticos/medios	El deploy gestiona estáticos y (si aplica) media de forma coherente (no rota la UI ni el admin).	☐
19	Migraciones y DB en producción	Migraciones aplicadas en entorno de deploy y BD operativa (según hosting).	☐
20	Preparación para evaluación individual	Cada integrante puede explicar arquitectura, endpoints, decisiones y realizar un cambio solicitado en sala.	☐

Entregables

• Repositorio público en GitHub con el código completo del proyecto.

• Archivo README.md con: instalación local, instrucciones de deploy, URL pública del sistema y endpoints API con ejemplos.

• URL pública del deploy (hosting gratuito).

• Evidencia del reporte (capturas) y evidencia de consumo de APIs (capturas/ejemplos).

Usuario administrador para evaluación

Usuario: admin

Contraseña: admin

(Si su deploy requiere credenciales distintas, deben indicarlas claramente en el README.)

Rúbrica de Evaluación (máximo 100 puntos)

Criterio	Excelente	Muy Bueno	Bueno	Regular	Insuficiente
Reporte del Sistema (20 pts)	Reporte dinámico con datos reales; vista protegida; filtros funcionales (p. ej., fechas/estado/platafor	Cumple datos reales y vista protegida; incluye	Reporte funciona pero limitado: solo tabla o gráfico,	Reporte incompleto: sin personalización real o con	Muy básico/inexistente; no usa datos reales o no posee gráficos/person

	ma); tabla + gráficos claros y coherentes. (20)	tabla + gráfico; personalización parcial o con detalle menor. (16)	o filtros poco claros; presenta información útil pero con debilidades. (12)	errores en visualización; métrica poco representativa. (8)	alización; no es verificable. (0)
Deploy del Sistema (20 pts)	Deploy estable con URL pública; rutas principales web + admin + APIs operativas; configuración mínima de producción (DEBUG off, ALLOWED_HOSTS, variables). README describe deploy. (20)	Deploy accesible y mayormente estable; funciona la mayoría de rutas y APIs; README con pasos pero incompleto o con un detalle menor. (16)	Deploy disponible pero con intermitencias o fallas parciales (alguna ruta/API no opera); configuración o README con omisiones. (12)	Deploy con errores relevantes: solo algunas rutas funcionan; APIs no verificables o admin inaccesible. (8)	No existe deploy o no es accesible públicamente. (0)
API 1 – CRUD de Insumos (5 pts)	CRUD completo (list/create/retrieve/update/destroy) operativo, serializer correcto y respuestas HTTP adecuadas. (5)	CRUD casi completo, con una omisión menor (p. ej., validación o un método con detalle menor). (4)	Funciona parcialmente (p. ej., crear y listar) pero faltan operaciones o hay errores en alguna acción. (3)	Existe endpoint pero falla con frecuencia o no respeta estructura REST; errores de serialización. (1)	No se encuentra o tiene muchos errores. (0)
API 2 – Crear y modificar pedidos (10 pts)	Permite POST y PUT/PATCH; bloquea GET colección y DELETE según restricción; validación y códigos HTTP correctos. (10)	Cumple restricción principal (bloquea list y delete) y permite crear/modificar;	Permite crear/modificar, pero una restricción está mal implementada o hay omisiones	Endpoint existe pero presenta errores frecuentes, permite operaciones no permitidas	No se encuentra o tiene muchos errores. (0)

		presenta detalle menor en validación o respuesta. (8)	de validación relevantes. (6)	o no actualiza correctamente. (3)	
API 3 – Filtro de pedidos por parámetros (15 pts)	Endpoint filtra por rango de fechas, estados y máximo resultados; valida parámetros; retorna solo coincidencias con respuesta consistente. (15)	Filtra correctamente por la mayoría de parámetros; validación parcial o un parámetro con implementación menor. (12)	Filtrado básico (solo uno o dos parámetros) o validación débil; resultados parcialmente correctos. (9)	Filtro con errores o comportamiento inconsistente; no respeta uno o más parámetros obligatorios. (5)	No se encuentra o tiene muchos errores. (0)
Evaluación Individual en clases (30 pts)	Demuestra dominio sólido: explica decisiones (reporte, DRF, deploy), identifica fallas y aplica modificación solicitada correctamente dentro del tiempo. (30)	Buen dominio: responde correctamente la mayoría; realiza modificación con apoyo menor o con un ajuste final. (24)	Dominio parcial: responde con vacíos o confusiones; logra modificación parcial o con errores que requieren corrección. (18)	Dominio bajo: respuestas inseguras o sin trazabilidad; no logra implementar la modificación solicitada. (12)	No responde o no logra realizar el requerimiento; sin claridad en sus respuestas. (0)

Escala de Apreciación

La calificación final se determina en base al puntaje obtenido sobre 100 puntos. Se utiliza el criterio: 60 puntos equivalen a nota 4,0.

Declaración de Uso de Inteligencia Artificial (IA)

Yo, _____ (nombre completo del/los estudiante/s), declaro que en el desarrollo de la Prueba N° 4 de la asignatura Programación Back End:

Herramienta(s) de IA utilizada(s):

_____

Parte(s) asistida(s) por IA:

_____

Nivel estimado de asistencia de IA: ____ % (máximo permitido: 50%)

Parte(s) desarrolladas sin asistencia de IA:

_____

Confirmo que la información aquí entregada es veraz. El incumplimiento puede implicar calificación mínima 1,1 y posibles sanciones académicas.

Firma del estudiante: _____ Fecha: ____ / ____ / 2025