

Manual del Sistema de Gestión de Pedidos de Compra

Vivero Aranjuez - Sistema Automatizado V2

Versión del Documento: 1.0

Fecha de Creación: 6 de febrero de 2026

Autor: Sistema de Pedidos Vivero V2

1. Introducción

Este documento describe el funcionamiento completo del Sistema de Gestión de Pedidos de Compra automatizado para Vivero Aranjuez. El sistema está diseñado para gestionar el proceso de generación de pedidos de compra de manera eficiente, utilizando clasificación ABC+D y forecast de ventas.

1.1 Propósito del Sistema

El sistema tiene como objetivo principal automatizar el proceso de generación de pedidos de compra para las diferentes secciones del vivero. Esto incluye el análisis de ventas históricas, la clasificación de artículos por importancia (categorías A, B, C y D), y la generación de pedidos optimizados basados en forecast y corrección por tendencia de ventas.

El sistema se ejecuta en dos niveles complementarios que trabajan juntos para proporcionar una gestión integral del inventario y los pedidos. Por un lado, existe un proceso trimestral que genera las clasificaciones ABC+D para cada sección, analizando el desempeño de todos los artículos durante un período determinado. Por otro lado, hay un proceso semanal que genera los pedidos específicos basándose en las clasificaciones ABC+D y los datos de ventas más recientes.

1.2 Estructura General del Sistema

El sistema está compuesto por varios scripts de Python que trabajan de manera coordinada para procesar los datos y generar los pedidos. Cada script tiene una función específica dentro del flujo de trabajo general. El script principal `clasificacionABC.py` se encarga de generar las clasificaciones ABC+D cuatro veces al año, mientras que `main.py` genera los pedidos semanalmente basándose en esas clasificaciones. Existen además scripts complementarios como `INFORME.py` y `PRESENTACION.py` que generan reportes y presentaciones cuando se requieren.

Todos estos scripts comparten una estructura de directorios común que organiza los datos de entrada y salida de manera clara y sistemática. Esta organización facilita el mantenimiento del sistema y permite mantener un histórico completo de todas las operaciones realizadas.

2. Arquitectura del Sistema

2.1 Estructura de Directorios

La estructura de directorios del sistema está diseñada para separar claramente los datos de cada período y mantener un histórico completo de todas las operaciones. A continuación se detalla la organización completa de carpetas y archivos.

El directorio raíz del proyecto es PROGRAMA_PEDIDOS, que contiene todos los componentes del sistema. En este nivel superior se encuentran los scripts principales que pueden ser ejecutados directamente por el usuario. El directorio config/ almacena los archivos de configuración del sistema, incluyendo la configuración general, la configuración común y la información de los encargados de cada sección. El directorio src/ contiene todos los módulos Python que implementan la lógica del sistema, organizados por funcionalidad.

El directorio data/ es el más importante para el funcionamiento del sistema, ya que almacena todos los datos de entrada y salida. Dentro de este directorio, se crean subdirectorios para cada período de análisis, siguiendo una nomenclatura estandarizada que incluye el número del período y las fechas de inicio y fin.

```

PROGRAMA_PEDIDOS/
├─ clasificacionABC.py
# Script principal - Clasificación ABC+D (4 veces/año)
├─ main.py
# Script principal - Generación de pedidos (semanal)
├─ INFORME.py # Generador de informes detallados
├─ PRESENTACION.py # Generador de presentaciones
├─ generar_informe_html.py # Generador de informes HTML
├─ run.bat
# Archivo por lotes para ejecución rápida
├─ requirements.txt # Dependencias Python
├─ README.md # Documentación inicial
|
├─ config/ # Configuración del sistema
|   ├─ config.json # Configuración general
|   ├─ config_comun.json # Configuración adicional
|   └─ encargados.json # Encargados de secciones
|
├─ src/ # Módulos del sistema
(compartidos)
|   ├─ config_loader.py # Cargador de configuración
centralizado
|   ├─ data_loader.py # Carga y procesamiento de datos
|   ├─ forecast_engine.py # Motor de forecast de ventas
|   ├─ correction_engine.py # Motor de corrección de pedidos
|   ├─ order_generator.py # Generador de pedidos
|   ├─ correction_data_loader.py # Cargador de datos de corrección
|   ├─ email_service.py # Servicio de envío de emails
|   ├─ scheduler_service.py # Servicio de programación
|   └─ state_manager.py # Gestor de estado del sistema
|
├─ data/ # Datos del sistema
|   ├─ PERIODO_1_2026-01-01_2026-02-28/
|   |   ├─ input/
|   |   |   ├─ compras.xlsx # Datos de compras del período
|   |   |   ├─ Ventas.xlsx # Datos de ventas del período
|   |   |   ├─ Stock.xlsx # Stock actual
|   |   |   └─ Coste.xlsx # Costes unitarios
|   |   └─ output/
|   |       ├─ CLASIFICACION_ABC+D_*.xlsx
|   |       ├─ INFORME_*.html
|   |       └─ PRESENTACION_*.html

```

```

|
|
| └─ PERIODO_2_2026-03-01_2026-05-31/
|   |
|   | └─ input/
|   |   |
|   |   └─ ...
|   |
|   └─ output/
|       |
|       └─ ...
|
|
| └─ PERIODO_3_2026-06-01_2026-08-31/
|   |
|   | └─ input/
|   |   |
|   |   └─ ...
|   |
|   └─ output/
|       |
|       └─ ...
|
|
| └─ PERIODO_4_2026-09-01_2026-12-31/
|   |
|   | └─ input/
|   |   |
|   |   └─ ...
|   |
|   └─ output/
|       |
|       └─ ...
|
| └─ COMPARTIDO/
|     |
|     | └─ state.json
|     |   |
|     |   └─ state.json.backup
|     |
|     └─ logs/
|
| └─ Documentacion/
|     |
|     └─ Manual_Sistema_Pedidos_V2.pdf

```

Recursos compartidos

Estado del sistema

Backup del estado

Logs de ejecución

Documentación del sistema

Este documento

2.2 Definición de Períodos

El sistema está diseñado para trabajar con cuatro períodos de análisis a lo largo del año. Cada período tiene una duración específica y se corresponde con una época del año diferente, lo que permite capturar las variaciones estacionales en las ventas y el comportamiento de compra de los clientes.

El primer período abarca desde el 1 de enero hasta el 28 de febrero, con una duración de 59 días. Este período cubre la temporada de inicio de año, que en el sector de jardinería es generalmente de menor actividad debido al clima invernal. Sin embargo, es importante para analizar el comportamiento post-navidad y preparar la transición hacia la primavera.

El segundo período va desde el 1 de marzo hasta el 31 de mayo, con una duración de 92 días. Este es uno de los períodos más importantes del año, ya que coincide con la primavera, que es la temporada alta de ventas en el sector de jardinería y plantas.

Durante estos meses se produce el mayor volumen de ventas de plantas de temporada, sustratos y productos relacionados con el jardín.

El tercer período comprende desde el 1 de junio hasta el 31 de agosto, también con 92 días de duración. Este período incluye el verano, donde las ventas pueden verse afectadas por el calor y las vacaciones, pero sigue siendo relevante para analizar el comportamiento del mercado durante los meses de mayor exposición solar.

El cuarto y último período va desde el 1 de septiembre hasta el 31 de diciembre, con una duración de 122 días. Este es el período más largo del año e incluye la vuelta al cole, el otoño y la temporada navideña. Es un período estratégico para analizar las tendencias de cierre de año y planificar el siguiente ciclo.

Período	Fechas	Días	Ejecución	Mes Principal
Periodo 1	1 enero - 28 febrero	59 días	Enero	Enero
Periodo 2	1 marzo - 31 mayo	92 días	Marzo	Marzo
Periodo 3	1 junio - 31 agosto	92 días	Junio	Junio
Periodo 4	1 septiembre - 31 diciembre	122 días	Septiembre	Septiembre

3. Descripción de Scripts

3.1 clasificacionABC.py

El script clasificacionABC.py es el componente fundamental del sistema para la clasificación de artículos. Su función principal es analizar los datos de compras, ventas, stock y costes de un período determinado para clasificar cada artículo en una categoría ABC+D. Esta clasificación es esencial para la gestión eficiente del inventario, ya que permite identificar los artículos más importantes (categoría A) y aquellos que no generan ventas (categoría D).

Frecuencia de ejecución: 4 veces al año, una vez por cada período de análisis.

Datos de entrada: El script requiere cuatro archivos de datos que deben estar ubicados en la carpeta de entrada del período correspondiente. El archivo compras.xlsx contiene todos los movimientos de compras del período, incluyendo código de artículo, nombre, fecha, unidades y precio. El archivo Ventas.xlsx registra todas las transacciones de venta, con información similar a compras pero desde la perspectiva de las ventas. El archivo Stock.xlsx proporciona el inventario actual de cada artículo. Finalmente, Coste.xlsx contiene los costes unitarios de cada artículo, necesarios para calcular el beneficio real de las ventas.

Datos de salida: El script genera 11 archivos Excel, uno para cada sección del vivero. Cada archivo contiene la clasificación ABC+D de todos los artículos de esa sección, con información detallada sobre ventas, stock, beneficio, riesgo de merma y acciones

sugeridas. Estos archivos se generan con un nombre que incluye el período de análisis, por ejemplo: CLASIFICACION_ABC+D_MAF_20260101-20260228.xlsx.

Categorías de clasificación: El sistema clasifica los artículos en cuatro categorías basadas en su contribución al beneficio total. La categoría A incluye los artículos que representan el 80% del beneficio, estos son los productos estrellas que no deben faltar nunca en el almacén. La categoría B comprende los artículos que representan el siguiente 15% del beneficio, estos forman el complemento de gama. La categoría C contiene los artículos con menor contribución, representando el 5% restante, y son productos de presencia mínima en las ventas. Finalmente, la categoría D incluye todos los artículos que no han tenido ventas durante el período de análisis.

Comandos de ejecución:

```
# Ejecutar para el Periodo 1 (Enero-Febrero)
python clasificacionABC.py --periodo 1

# Ejecutar para el Periodo 2 (Marzo-Mayo)
python clasificacionABC.py --periodo 2

# Ejecutar para el Periodo 3 (Junio-Agosto)
python clasificacionABC.py --periodo 3

# Ejecutar para el Periodo 4 (Septiembre-Diciembre)
python clasificacionABC.py --periodo 4

# Ejecutar modo verbose para ver detalles
python clasificacionABC.py --periodo 1 --verbose

# Procesar solo una sección específica
python clasificacionABC.py --periodo 1 --seccion vivero
```

Proceso interno: Cuando se ejecuta el script, sigue una secuencia de pasos claramente definida. Primero, carga los cuatro archivos de datos desde la carpeta del período indicado. Segundo, normaliza los datos, homogenizando códigos de artículo, nombres, tallas y colores. Tercero, filtra los artículos con códigos válidos (mínimo 10 dígitos) y elimina registros con unidades cero o valores nulos. Cuarto, para cada sección, procesa los artículos calculando métricas como ventas, beneficio, stock, tasa de rotación y riesgo de merma. Quinto, aplica la clasificación ABC+D basada en el beneficio acumulado. Sexto, genera las acciones sugeridas según el escenario de cada artículo. Séptimo, crea los archivos Excel con formato profesional y los guarda en la carpeta de salida del período. Finalmente, envía un email automáticamente al encargado de cada sección con el archivo correspondiente adjunto.

3.2 main.py

El script main.py es el motor de generación de pedidos semanales. Su función es crear los pedidos de compra para cada sección basándose en las clasificaciones ABC+D generadas por clasificacionABC.py, los datos de ventas actuales y los algoritmos de forecast y corrección.

Frecuencia de ejecución: Semanal, preferentemente los domingos a las 15:00 cuando el sistema está configurado en modo automático.

Datos de entrada: El script utiliza múltiples fuentes de datos para generar los pedidos. Los archivos CLASIFICACION_ABC+D.xlsx proporcionan la clasificación de artículos y sus parámetros de stock mínimo y máximo. El archivo Ventas.xlsx contiene los datos de ventas de la semana actual y anteriores, utilizados para el forecast. El archivo Stock_actual.xlsx proporciona el inventario actual de cada artículo. El archivo Coste.xlsx contiene los costes unitarios para calcular el importe de los pedidos. El archivo compras.xlsx registra las compras recientes, útil para evitar duplicidades. Adicionalmente, se pueden usar archivos de corrección opcionales como Stock_actual.xlsx, Ventas_semana.xlsx y Compras_semana_*.xlsx.

Datos de salida: El script genera múltiples archivos de salida organizados por sección y semana. Los archivos Pedido_Semana_XX_YYYY-MM-DD_seccion.xlsx contienen el detalle de pedidos para cada sección. El archivo Resumen_Pedidos_CONSOLIDADO_YYYY-MM-DD.xlsx contiene un resumen consolidado de todos los pedidos de la semana. Adicionalmente, se pueden generar archivos corregidos cuando se aplica la corrección FASE 2.

Fases del proceso:

FASE 1 - Forecast: En esta primera fase, el sistema analiza las ventas históricas de cada artículo para predecir las ventas de la próxima semana. El algoritmo considera múltiples factores, incluyendo las ventas de semanas anteriores, la estacionalidad del artículo (basada en su familia), y los días de cobertura objetivo según la categoría ABC del artículo. El resultado es un pedido teórico que cubre las necesidades predichas de cada artículo.

FASE 2 - Corrección: En esta segunda fase, el sistema ajusta el pedido teórico basándose en la realidad operativa actual. Se consideran factores como el stock real actual (que puede estar por encima o por debajo del mínimo recomendado), las ventas reales de la semana anterior (que pueden indicar tendencias de aumento o disminución), las compras recibidas recientemente, y las tendencias detectadas de aumento de ventas. La corrección por tendencia de ventas es especialmente importante: si un artículo está vendiendo por encima de lo esperado y ha consumido parte de su stock mínimo, el sistema incrementa el pedido para compensar.

Comandos de ejecución:

```
# Ejecución automática (domingo 15:00)
python main.py

# Forzar semana específica
python main.py --semana 14

# Con corrección habilitada
python main.py --semana 14 --con-correccion

# Sin corrección (solo FASE 1)
python main.py --semana 14 --sin-correccion

# Modo continuo (espera hasta el horario de ejecución)
python main.py --continuo

# Verificar configuración de email
python main.py --verificar-email

# Ver estado del sistema
python main.py --status

# Mostrar ayuda
python main.py --help
```

Parámetros importantes:

El parámetro `--semana` o `-s` permite especificar el número de semana a procesar (1-52). Este parámetro fuerza el procesamiento de una semana específica, ignorando el cálculo automático del sistema.

El parámetro `--con-correccion` habilita la FASE 2 de corrección del pedido. Si no se especifica, el sistema usa la configuración por defecto del archivo `config.json`.

El parámetro `--verbose` o `-v` activa el logging detallado, mostrando información de depuración durante la ejecución.

El parámetro `--log` permite especificar un archivo de log personalizado (por defecto: `logs/sistema.log`).

3.3 INFORME.py

El script `INFORME.py` genera informes detallados de ventas para análisis gerencial. Este script es útil para revisar el desempeño de cada sección y artículo durante un período específico.

Frecuencia de ejecución: Bajo demanda, cuando se requiere un análisis detallado.

Comandos de ejecución:

```
# Generar informe del Periodo 1
python INFORME.py --periodo 1

# Generar informe verbose
python INFORME.py --periodo 1 --verbose
```

3.4 PRESENTACION.py

El script PRESENTACION.py genera presentaciones en formato HTML para reuniones de equipo o presentaciones gerenciales. Las presentaciones incluyen gráficos y métricas clave del desempeño de cada sección.

Frecuencia de ejecución: Bajo demanda, para reuniones o presentaciones formales.

Comandos de ejecución:

```
# Generar presentación del Periodo 1
python PRESENTACION.py --periodo 1

# Generar presentación verbose
python PRESENTACION.py --periodo 1 --verbose
```

4. Configuración del Sistema

4.1 Archivo config.json

El archivo config.json contiene la configuración general del sistema. A continuación se describen las secciones principales de este archivo.

Configuración de rutas: Define las rutas relativas para los directorios de entrada, salida y otros recursos del sistema. Estas rutas son relativas al directorio del script que se está ejecutando.

Configuración de parámetros: Incluye parámetros globales como el porcentaje de crecimiento objetivo, el porcentaje de stock mínimo, y otros umbrales utilizados por los algoritmos de forecast y corrección.

Configuración de secciones: Define las secciones activas del vivero y sus parámetros específicos, incluyendo los objetivos semanales de cada sección.

Configuración de email: Define los parámetros del servidor SMTP para el envío automático de emails, incluyendo el servidor, puerto, remitente y credenciales.

Configuración de corrección: Define los parámetros de la FASE 2 de corrección, incluyendo si la corrección está habilitada y las políticas de stock mínimo por categoría ABC.

4.2 Archivo encargados.json

El archivo encargados.json mapea cada sección con su encargado correspondiente, incluyendo nombre y email. Este archivo es utilizado por el sistema para enviar automáticamente los informes de clasificación ABC+D a cada encargado.

Estructura del archivo:

```
{
  "encargados": {
    "maf": {
      "nombre": "Nombre del encargado",
      "email": "email@ejemplo.com"
    },
    "interior": {
      "nombre": "Nombre del encargado",
      "email": "email@ejemplo.com"
    }
  }
}
```

4.3 Configuración de Variables de Entorno

Para el envío de emails, el sistema requiere la contraseña del remitente configurada como variable de entorno. En sistemas Windows (PowerShell), se configura de la siguiente manera:

```
$env:EMAIL_PASSWORD="tu_contraseña_aqui"
```

En sistemas Linux/macOS (bash):

```
export EMAIL_PASSWORD="tu_contraseña_aqui"
```

5. Flujo de Trabajo del ERP

5.1 Envío de Archivos para Clasificación ABC+D (4 veces al año)

El sistema ERP debe seguir el siguiente procedimiento para enviar los archivos de datos al sistema de pedidos. Este proceso debe ejecutarse al final de cada período de análisis, antes de ejecutar `clasificacionABC.py`.

Paso 1: Generación de archivos

El ERP genera los cuatro archivos necesarios con los datos del período correspondiente. Cada archivo debe tener el formato esperado por el sistema.

Paso 2: Envío a la carpeta del período

El ERP coloca los archivos en la carpeta de entrada del período correspondiente. Es importante que los archivos se envíen a la carpeta correcta según el período que se está procesando.

Ejemplo de estructura de envío:

```
Carpeta destino: data/PERIODO_1_2026-01-01_2026-02-28/input/
```

Archivos a enviar:

```
- compras.xlsx    → data/PERIODO_1_2026-01-01_2026-02-28/input/
compras.xlsx
- Ventas.xlsx     → data/PERIODO_1_2026-01-01_2026-02-28/input/
Ventas.xlsx
- Stock.xlsx      → data/PERIODO_1_2026-01-01_2026-02-28/input/
Stock.xlsx
- Coste.xlsx      → data/PERIODO_1_2026-01-01_2026-02-28/input/
Coste.xlsx
```

Paso 3: Ejecución del script

Una vez enviados los archivos, el usuario ejecuta el script de clasificación para el período correspondiente.

```
python clasificacionABC.py --periodo 1
```

Paso 4: Verificación de resultados

El script genera los archivos de clasificación en la carpeta de salida del período. El usuario debe verificar que todos los archivos se generaron correctamente.

Paso 5: Copia a la carpeta de entrada para `main.py`

Los archivos ABC+D generados deben copiarse a la carpeta data/input/ para que main.py pueda utilizarlos en la generación semanal de pedidos.

```
# Copiar todos los archivos ABC+D al directorio de entrada
cp data/PERIODO_1_*/output/CLASIFICACION_ABC+D_*.xlsx data/input/
```

5.2 Envío de Archivos Semanales

Para la generación semanal de pedidos, el ERP debe mantener actualizados los siguientes archivos en la carpeta data/input/:

Archivo Ventas.xlsx: Este archivo debe actualizarse semanalmente con las ventas de la semana actual. El sistema lo utiliza para el forecast de la próxima semana.

Archivo Stock_actual.xlsx: Este archivo debe reflejar el inventario actual del almacén. Se utiliza en la FASE 2 de corrección para ajustar los pedidos según el stock real.

Archivo Coste.xlsx: Este archivo contiene los costes unitarios de cada artículo y debe mantenerse actualizado. Se utiliza para calcular el importe de los pedidos.

Archivo compras.xlsx: Este archivo registra las compras recientes y se utiliza para evitar duplicidades en los pedidos.

Archivos de corrección opcionales: Para la FASE 2 avanzada, se pueden proporcionar archivos adicionales como Ventas_semana_XX.xlsx y Compras_semana_XX.xlsx para cada semana.

6. Algoritmos del Sistema

6.1 Clasificación ABC+D

El algoritmo de clasificación ABC+D es fundamental para la gestión eficiente del inventario. Se basa en el principio de Pareto (80/20), donde un pequeño número de artículos genera la mayor parte del beneficio.

Paso 1: Filtrado de artículos con ventas

Se separan los artículos que han tenido ventas durante el período de aquellos que no han vendido nada. Los artículos sin ventas se clasifican automáticamente como categoría D.

Paso 2: Cálculo de beneficio por artículo

Para cada artículo con ventas, se calcula el beneficio total generado durante el período. El beneficio se calcula como el importe de ventas menos el coste de las mercancías vendidas.

Paso 3: Ordenamiento por beneficio

Los artículos se ordenan de mayor a menor beneficio generado.

Paso 4: Cálculo de porcentaje acumulado

Se calcula el porcentaje de beneficio de cada artículo respecto al total, y se calcula el porcentaje acumulado.

Paso 5: Asignación de categorías

- Categoría A: Artículos con porcentaje acumulado menor o igual al 80%
- Categoría B: Artículos con porcentaje acumulado mayor al 80% y menor o igual al 95%
- Categoría C: Artículos restantes con porcentaje acumulado mayor al 95%
- Categoría D: Artículos sin ventas durante el período

6.2 Forecast de Ventas (FASE 1)

El algoritmo de forecast predice las ventas de la próxima semana basándose en datos históricos.

Factores considerados:

El algoritmo considera múltiples factores para generar el forecast. Las ventas históricas del artículo proporcionan la base para predecir las ventas futuras. La familia del artículo determina la rotación esperada (días de cobertura de stock). La categoría ABC del artículo influye en los objetivos de disponibilidad. Los días de cobertura objetivo varían según la categoría, siendo más estrictos para artículos A.

Cálculo de stock mínimo y máximo:

El sistema calcula el stock mínimo y máximo para cada artículo según su categoría y familia:

Para rotación de 7 días: Stock mínimo = ventas_día * 3.5, Stock máximo = ventas_día * 10.5

Para rotación de 15 días: Stock mínimo = ventas_día * 7.5, Stock máximo = ventas_día * 22.5

Para rotación de 30 días: Stock mínimo = ventas_día * 15, Stock máximo = ventas_día * 45

Para rotación de 60 días: Stock mínimo = ventas_día * 30, Stock máximo = ventas_día * 90

Para rotación de 90 días: Stock mínimo = ventas_día * 45, Stock máximo = ventas_día * 135

6.3 Corrección de Pedidos (FASE 2)

La FASE 2 ajusta el pedido teórico de la FASE 1 basándose en la realidad operativa.

Fórmula principal:

La corrección se aplica mediante la siguiente fórmula:

$$\text{Pedido_Corregido} = \max(0, \text{Pedido_Generado} + (\text{Stock_Mínimo} - \text{Stock_Real}))$$

Donde:

- Pedido_Generado: Resultado del algoritmo de forecast (FASE 1)
- Stock_Mínimo: Stock mínimo objetivo del artículo (calculado según su categoría ABC)
- Stock_Real: Stock físico actual en el almacén
- $\max(0, \dots)$: Función para evitar pedidos negativos

Corrección por tendencia de ventas:

Cuando las ventas reales de un artículo superan las ventas objetivo, el sistema detecta una tendencia de aumento y aplica un incremento adicional al pedido.

```
Porcentaje_Consumido = (Ventas_Reales - Ventas_Objetivo) /  
Ventas_Objetivo  
Incremento_Tendencia = Ventas_Objetivo * Porcentaje_Consumido  
Pedido_Final = Pedido_Corregido + Incremento_Tendencia
```

7. Secciones del Sistema

El sistema está diseñado para gestionar las siguientes secciones del vivero, cada una con sus propias características y encargados:

MAF (Mesas, Asientos, Fuentes): Sección dedicada a mobiliario de jardín. Los códigos de artículo comienzan con el dígito 7.

DECO_INTERIOR (Decoración Interior): Artículos de decoración para interiores. Códigos начинаются con 6.

SEMILLAS: Sección de semillas y productos relacionados con la siembra. Códigos начинаются con 5.

MASCOTAS_VIVO: Productos vivos para mascotas (animales). Códigos начинаются con 2 y son códigos de 4 dígitos específicos (2104, 2204, etc.).

MASCOTAS_MANUFACTURADO: Productos manufacturados para mascotas. Códigos начинаются con 2 pero no están en la lista de vivos.

INTERIOR: Plantas de interior. Códigos начинаются con 1.

FITOS (Fitosanitarios): Productos fitosanitarios y de protección de plantas. Códigos начинаются con 3, pero no son de tierra/áridos (31, 32).

VIVERO: Plantas de vivero y árboles. Códigos начинаются con 8.

UTILES_JARDIN (Útiles de Jardín): Herramientas y útiles de jardín. Códigos начинаются con 4.

TIERRAS_ARIDOS (Tierras y Áridos): Tierras, sustratos y áridos. Códigos начинаются con 31 o 32.

DECO_EXTERIOR (Decoración Exterior): Artículos de decoración para exteriores. Códigos начинаются con 9.

8. Troubleshooting y Mantenimiento

8.1 Problemas Comunes

Error: No se encontró el archivo

Si el sistema reporta que no encuentra un archivo, verificar que la ruta sea correcta y que el archivo exista. Para `clasificacionABC.py`, verificar que los archivos estén en la carpeta del período correcto. Para `main.py`, verificar que los archivos ABC+D estén en `data/input/`.

Error: Faltan columnas en el archivo

El sistema busca columnas específicas en cada archivo. Si falta una columna, revisar que el archivo Excel tenga las columnas esperadas. El sistema tiene lógica de normalización de columnas que es insensible a mayúsculas/minúsculas y acentos, pero requiere que las columnas existan con nombres similares.

Error: SMTP al enviar emails

Verificar que la variable de entorno `EMAIL_PASSWORD` esté configurada correctamente. Verificar la configuración del servidor SMTP en `config.json`. Usar el comando `python main.py --verificar-email` para diagnosticar problemas.

No hay datos de ventas para la semana

Esto puede ocurrir si se especifica una semana futura o si el archivo `Ventas.xlsx` no contiene datos para esa semana. Verificar que el archivo `Ventas.xlsx` tenga registros para la semana que se intenta procesar.

8.2 Backup y Recuperación

Respaldo del estado del sistema:

El archivo `state.json` en `data/COMPARTIDO/` contiene el estado actual del sistema, incluyendo el historial de ejecuciones y el stock acumulado. Se recomienda hacer backup de este archivo regularmente.

Recuperación después de un error:

Si el sistema falla durante la ejecución, verificar los logs en `data/COMPARTIDO/logs/` para identificar la causa del error. Corregir el problema y re-ejecutar el script con la opción `--semana` para forzar el reprocesamiento de la semana.

8.3 Actualización del Sistema

Actualización de código:

Si se actualiza el código del sistema, verificar que todos los scripts estén sincronizados. Las configuraciones en `config/` son compartidas, por lo que cualquier cambio afectará a todos los scripts.

Cambio de período:

Para cambiar de período de análisis, simplemente ejecutar `clasificacionABC.py` con el nuevo número de período. El sistema会自动mente utilizará los datos de la carpeta correspondiente.

9. Anexos

9.1 Formato de Archivos de Entrada

Formato de `compras.xlsx`:

Campo	Tipo	Descripción
Artículo	Texto	Código del artículo
Nombre artículo	Texto	Descripción del artículo
Fecha	Fecha	Fecha de la compra
Unidades	Número	Cantidad comprada
Precio	Número	Precio unitario

Formato de `Ventas.xlsx`:

Campo	Tipo	Descripción
Artículo	Texto	Código del artículo
Nombre artículo	Texto	Descripción del artículo
Fecha	Fecha	Fecha de la venta
Unidades	Número	Cantidad vendida
Importe	Número	Importe total de la venta
Tipo registro	Texto	Tipo de registro (usar "Detalle")

Formato de `Stock.xlsx`:

Campo	Tipo	Descripción
Artículo	Texto	Código del artículo
Nombre artículo	Texto	Descripción del artículo
Talla	Texto	Talla del artículo (si aplica)
Color	Texto	Color del artículo (si aplica)

Campo	Tipo	Descripción
Unidades	Número	Stock disponible

Formato de Coste.xlsx:

Campo	Tipo	Descripción
Artículo	Texto	Código del artículo
Talla	Texto	Talla del artículo
Color	Texto	Color del artículo
Coste	Número	Coste unitario
Últ. Compra	Fecha	Fecha de última compra

9.2 Códigos de Familia y Rotación

El sistema utiliza códigos de familia para determinar la rotación esperada de cada artículo:

Familia	Nombre	Rotación (días)
10-19	Plantas Interior	30
20-29	Mascotas	30
31-32	Tierras y Áridos	60
33-39	Fitosanitarios	90
40-49	Útiles Jardín	90
50-59	Semillas	60
60-69	Deco Interior	90
70-79	MAF	90
80-89	Vivero	60
90-99	Deco Exterior	90

9.3 Matriz de Escenarios

El sistema utiliza una matriz de escenarios para determinar las acciones sugeridas para cada artículo. Cada escenario considera tres factores: ventas reales vs objetivo, compras reales vs sugerido, y stock real vs mínimo.

10. Glosario de Términos

ABC+D: Método de clasificación de artículos basado en su importancia (80% A, 15% B, 5% C, 0% D).

Forecast: Predicción de ventas futuras basada en datos históricos.

Rotación: Número de días que tarda un artículo en venderse completamente.

Stock mínimo: Nivel de inventario mínimo recomendado para un artículo.

Stock máximo: Nivel de inventario máximo recomendado para un artículo.

Tasa de venta: Porcentaje del stock disponible que se ha vendido.

Período de análisis: Intervalo de tiempo utilizado para la clasificación ABC+D (2-3 meses).

Semana de pedido: Semana específica para la cual se genera un pedido de compra.

FASE 1: Etapa de forecast donde se genera el pedido teórico basándose en ventas históricas.

FASE 2: Etapa de corrección donde se ajusta el pedido teórico según la realidad operativa.

Corrección por tendencia: Ajuste adicional al pedido cuando las ventas reales superan las ventas objetivo.

Documento generado automáticamente por el Sistema de Pedidos Vivero V2

Para dudas o sugerencias, consultar al departamento de sistemas.