**# Anexo de Desarrollo - Sistema de Planificación de Producción**

**## 1. Visión General del Desarrollo**

**### 1.1 Objetivo**

**El desarrollo de este sistema surgió de la necesidad de optimizar la planificación de producción en un entorno industrial, con el objetivo principal de minimizar roturas de stock mientras se mantiene una producción eficiente. El sistema combina predicción de demanda con optimización de recursos, considerando múltiples restricciones operativas.**

**### 1.2 Arquitectura del Sistema**

**El sistema se estructura en módulos independientes pero interconectados:**

**1. \*\*Core (main.py)\*\*:**

**- Gestión de flujo principal**

**- Validación de entradas**

**- Control de ejecución**

**2. \*\*Módulo de Productos (class Producto)\*\*:**

**- Gestión de datos de productos**

**- Cálculos de métricas**

**- Conversión de tipos**

**3. \*\*Optimizador (método Simplex)\*\*:**

**- Implementación de algoritmo**

**- Gestión de restricciones**

**- Generación de plan**

**4. \*\*Utilidades (funciones auxiliares)\*\*:**

**- Procesamiento de datos**

**- Validaciones**

**- Exportación de resultados**

**## 2. Proceso de Desarrollo**

**### 2.1 Fase Inicial**

**El desarrollo comenzó con análisis de los requerimientos y diseño inicial del sistema.**

**#### 2.1.1 Análisis de Requisitos**

**- Identificación de variables clave:**

**- Stock disponible**

**- Demanda histórica**

**- Capacidad de producción**

**- Restricciones operativas**

**- Definición de objetivos:**

**- Minimizar roturas de stock**

**- Optimizar recursos de producción**

**- Mantener niveles de stock eficientes**

**#### 2.1.2 Diseño Inicial**

**- Estructura de datos:**

**```python**

**class Producto:**

**def \_\_init\_\_(self, ...):**

**# Datos básicos**

**self.cod\_art = cod\_art**

**self.nom\_art = nom\_art**

**# Stocks**

**self.disponible = disponible**

**self.calidad = calidad**

**# Campos calculados**

**self.demanda\_media = 0**

**self.stock\_inicial = 0**

**```**

**### 2.2 Implementación**

**#### 2.2.1 Primera Fase**

**1. \*\*Lectura de Datos\*\***

**- Implementación de lectura de CSV**

**- Manejo de codificación de caracteres**

**- Validación de estructura**

**2. \*\*Cálculos Básicos\*\***

**- Demanda media**

**- Stock inicial**

**- Coberturas**

**#### 2.2.2 Segunda Fase**

**1. \*\*Implementación del Optimizador\*\***

**- Método Simplex**

**- Restricciones de producción**

**- Validaciones de resultados**

**2. \*\*Sistema de Exportación\*\***

**- Formato CSV**

**- Métricas calculadas**

**- Estados de productos**

**### 2.3 Desafíos y Soluciones**

**#### 2.3.1 Desafíos Encontrados**

**1. \*\*Codificación de Caracteres\*\***

**- Problema: Caracteres especiales mal visualizados**

**- Solución: Implementación de UTF-8 con BOM**

**```python**

**df.to\_csv(nombre\_archivo, encoding='utf-8-sig')**

**```**

**2. \*\*Validación de Entradas\*\***

**- Problema: Datos inconsistentes**

**- Solución: Sistema robusto de validación**

**```python**

**def validar\_fecha(fecha\_str, nombre\_campo):**

**try:**

**fecha\_dt = datetime.strptime(fecha\_str, '%d-%m-%Y')**

**return fecha\_dt, fecha\_str**

**except ValueError:**

**raise ValueError(f"Formato de {nombre\_campo} inválido")**

**```**

**3. \*\*Optimización de Recursos\*\***

**- Problema: Restricciones complejas**

**- Solución: Implementación Simplex con múltiples restricciones**

**## 3. Explicación Detallada del Código**

**### 3.1 Estructura Principal (main.py)**

**#### 3.1.1 Flujo Principal**

**```python**

**def main():**

**try:**

**# 1. Solicitar y validar parámetros**

**while True:**

**fecha\_dataset = input("Ingrese fecha de dataset DD-MM-YYYY: ")**

**if verificar\_dataset\_existe(construir\_nombre\_archivo(fecha\_dataset)):**

**break**

**# 2. Leer y procesar datos**

**productos = leer\_dataset(nombre\_dataset)**

**# 3. Optimizar producción**

**productos\_optimizados = aplicar\_simplex(...)**

**# 4. Exportar resultados**

**exportar\_resultados(...)**

**except Exception as e:**

**logger.error(f"Error en ejecución: {str(e)}")**

**```**

**#### 3.1.2 Sistema de Validación**

**```python**

**def validar\_fecha(fecha\_str, nombre\_campo):**

**"""**

**Valida el formato de fecha y la convierte a objeto datetime**

**Args:**

**fecha\_str (str): Fecha en formato DD-MM-YYYY**

**nombre\_campo (str): Nombre del campo para mensajes de error**

**Returns:**

**tuple: (datetime, str) Fecha convertida y string original**

**"""**

**```**

**### 3.2 Módulo de Productos**

**#### 3.2.1 Clase Producto**

**```python**

**class Producto:**

**def \_\_init\_\_(self, ...):**

**"""**

**Inicializa un producto con sus atributos básicos y calculados**

**Args:**

**cod\_art (str): Código del artículo**

**nom\_art (str): Nombre del artículo**

**...**

**"""**

**```**

**#### 3.2.2 Cálculos y Conversiones**

**```python**

**def \_convertir\_float(self, valor):**

**"""**

**Convierte valores a float manejando diferentes formatos**

**Args:**

**valor: Valor a convertir (str, int, float)**

**Returns:**

**float: Valor convertido**

**"""**

**```**

**### 3.3 Sistema de Optimización**

**#### 3.3.1 Método Simplex**

**```python**

**def aplicar\_simplex(productos\_validos, horas\_disponibles, ...):**

**"""**

**Optimiza la producción usando el método Simplex**

**Args:**

**productos\_validos (list): Lista de productos a optimizar**

**horas\_disponibles (float): Horas totales disponibles**

**Returns:**

**list: Productos con cantidades optimizadas**

**"""**

**```**

**#### 3.3.2 Restricciones**

**- Horas disponibles**

**- Stock de seguridad**

**- Lotes mínimos**

**- Capacidad de producción**

**### 3.4 Sistema de Reportes**

**#### 3.4.1 Exportación**

**```python**

**def exportar\_resultados(productos\_optimizados, productos, ...):**

**"""**

**Genera reporte CSV con resultados de la planificación**

**Args:**

**productos\_optimizados (list): Productos con planificación**

**productos (list): Todos los productos activos**

**"""**

**```**

**## 4. Resultados y Validación**

**### 4.1 Métricas de Rendimiento**

**- Tiempo de ejecución**

**- Precisión de optimización**

**- Calidad de soluciones**

**### 4.2 Casos de Prueba**

**- Escenarios típicos**

**- Casos límite**

**- Manejo de errores**

**## 5. Conclusiones y Mejoras Futuras**

**### 5.1 Logros Alcanzados**

**- Sistema funcional y robusto**

**- Optimización efectiva**

**- Manejo adecuado de errores**

**### 5.2 Mejoras Propuestas**

**1. Interfaz gráfica**

**2. Persistencia de configuraciones**

**3. Análisis predictivo avanzado**

**4. Reportes adicionales**

**### 5.3 Consideraciones de Mantenimiento**

**1. Actualización de dependencias**

**2. Monitoreo de rendimiento**

**3. Backup de configuraciones**

**4. Documentación continua**