

FO-X | Versión: 2 | Fecha: 29/04/2025 | Página 1 de 5

Introducción

Este documento describe un ejercicio práctico para el desarrollo de componentes en React. Se trabajará con un conjunto de datos simulados que representan la proyección diaria de productos. El objetivo es construir dos componentes principales: un Grid de Proyección y un Resumen de datos.

Datos Proporcionados

Se proporcionará un listado de datos con la siguiente estructura JSON:

```
Unset

{
    "CenterCode": "BCV",
    "Reference": "210001000004R20",
    "VisibleForecastedDate": "2025-03-21T00:00:00",
    "NetFlow": 1513.0,
    "GreenZone": 121.768285,
    "YellowZone": 121.768285,
    "RedZone": 194.829256,
    "MakeToOrder": 0
}
```

Cada objeto representa la información simulada de un producto en un día específico.

Campos de Datos

- CenterCode: Código del centro donde se encuentra el producto.
- Reference: Código identificador del producto (Referencia).
- VisibleForecastedDate: Fecha simulada.
- NetFlow: Valor de inventario neto por día.
- GreenZone: Rango para pintar de color verde.
- YellowZone: Rango para pintar de color amarillo.
- RedZone: Rango para pintar de color rojo.
- MakeToOrder: Pedido de Abastecimiento por día.



FO-X | Versión: 2 | Fecha: 29/04/2025 | Página 1 de 5

Ejercicio

Componentes a Desarrollar

1. Grid de Proyección

Construir un componente tipo tabla que:

- Muestre los datos organizados por producto y fecha.
- Cada celda muestra el valor de `MakeToOrder`.
- El color de fondo de cada celda se determina según el cálculo `NetFlow +
 MakeToOrder`, comparado con los rangos `RedZone`, `YellowZone` y
 `GreenZone`.
 - Rojo: Si `1 <= (NetFlow + MakeToOrder) <= RedZone`.
 - Amarillo: Si `RedZone < (NetFlow + MakeToOrder) <= RedZone + YellowZone`.
 - Verde: Si `RedZone + YellowZone < (NetFlow + MakeToOrder) <= RedZone + YellowZone + GreenZone`.
 - Negro: Si `(NetFlow + MakeToOrder) == 0`.
 - Azul: Si `(NetFlow + MakeToOrder) > RedZone + YellowZone + GreenZone`.
- Al editar el valor de `MakeToOrder` de una celda:
 - Se recalcule únicamente el color de esa celda.
 - No se modifiquen ni recalculen las celdas siguientes en la misma fila.
 - El nuevo color se determina en tiempo real basado en el nuevo valor ingresado y el `NetFlow` correspondiente.
 - El Componente 2 (Resumen) se recalcule tomando en cuenta el nuevo valor de la celda.

Ejemplo:

Si `NetFlow = 100`, `MakeToOrder = 23`, `RedZone = 50`, `YellowZone = 55`, `GreenZone = 55`, el color de la celda es **verde**.

Importante: Cada celda del grid debe tener un 'backColor' según la regla anterior. El dato que se muestra es 'MakeToOrder', pero el que define el color es 'NetFlow + MakeToOrder'.



FO-X | Versión: 2 | Fecha: 29/04/2025 | Página 1 de 5



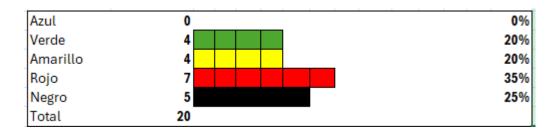
2. Resumen

Crear un componente de resumen que muestre:

- Cantidad de filas por cada color en la columna seleccionada.
- Presentación en porcentaje de cada color.

Funcionalidades:

- Calcular la cantidad de filas por cada color y su porcentaje perteneciente a la columna de la celda seleccionada.
- Actualizarse la información al seleccionar una celda en una columna diferente de fecha en el grid de proyección.
- Solo cargar información si la columna seleccionada es una de las columnas de proyección (con fechas).





FO-X | Versión: 2 | Fecha: 29/04/2025 | Página 1 de 5

Resultado Final

Ambos componentes deben estar montados en una página y contar con las funcionalidades descritas para considerar el ejercicio exitoso.



Entrega

Al finalizar, enviar un archivo comprimido (`.zip`) al correo electrónico desde el cual se recibió este documento con el asunto "Prueba Desarrollo WA 2025". El archivo debe contener:

- El proyecto React completo.
- `package.json`.
- Código fuente.
- Un `README` básico con instrucciones para ejecutar el proyecto (`npm install + npm start`).

Criterios de Aceptación y Evaluación

- 1. Diseño y Creatividad de la Interfaz Gráfica (35%)
 - Diseño limpio y profesional.
 - Buena UX y disposición.
 - Uso adecuado de colores, espaciado y componentes visuales.
 - Se permite el uso de librerías de terceros (ej., Material UI, Ant Design).
- 2. Funcionalidad y Exactitud (25%)
 - Lógica de color correcta y conforme a las reglas.
 - Edición funcional sin errores.
 - Resumen reflejando correctamente los datos.



FO-X | Versión: 2 | Fecha: 29/04/2025 | Página 1 de 5

3. Rendimiento y Optimización (25%)

- Fluidez en la navegación.
- Uso eficiente del renderizado.
- Buen manejo del estado

4. Código y Buenas Prácticas (15%)

- Código modular y limpio.
- Buena organización del proyecto.
- Comentarios útiles y estructura clara.

Para ser considerado exitoso, el proyecto debe cumplir al menos el 80% de los requisitos funcionales.

¡Muchos éxitos!