

INTRODUCCIÓN

El objetivo de esta prueba es la creación de una aplicación web enfocada en la visualización, búsqueda y gestión de un catálogo de teléfonos móviles. La aplicación debe permitir a los usuarios consultar detalles específicos de cada dispositivo, así como gestionar un carrito de compras de manera eficiente.

FUNCIONALIDADES Y ESTRUCTURA

La aplicación deberá contener tres vistas principales:

1.Vista Listado de Teléfonos

Requerimientos:

- Implementar una cuadrícula con tarjetas que muestren los primeros 20 teléfonos provenientes de la API.
 - Cada tarjeta deberá incluir imagen, nombre, marca y precio base.
- Implementar un buscador en tiempo real que filtre los teléfonos por nombre o marca (Usar filtrado por API).
- El buscador debe incluir un indicador con el número de resultados encontrados.
- Implementar una barra de navegación que contenga:
 - Un icono con un enlace al panel de inicio.
 - Un icono que muestre la cantidad de teléfonos en el carrito.
 - El carrito debe de ser persistente, se puede manejar su estado haciendo uso de localStorage.
- Al hacer clic en un teléfono, deberá redirigir a la vista de detalle del mismo.

2.Vista Detalle de Teléfono

Requerimientos:

Mostrar detalles del teléfono seleccionado, incluyendo:

- Nombre y marca del dispositivo.
- Imagen grande del móvil, con capacidad de cambiar dinámicamente según el color seleccionado.
- Selectores para almacenamiento y color, con actualización en tiempo real del precio.
- Especificaciones técnicas detalladas, precio base y variaciones según almacenamiento.
- Un botón "Añadir al carrito" que solo se activará cuando se hayan seleccionado color y almacenamiento.
- Una sección de "Productos similares" en la parte inferior.

3.Vista de Carrito

Requerimientos:

Mostrar los teléfonos añadidos al carrito, con:

- Imagen, nombre, especificaciones seleccionadas (almacenamiento / color) y precio individual.
- Implementar un botón para eliminar productos individuales del carrito.
- Mostrar el precio total de la compra.
- Un botón de "Continuar comprando" que redirija a la vista principal.

DISEÑO

El diseño de las vistas deberá ser responsive y ajustarse a los diseños definidos en Figma

- [Diseños de figma](#)
- [Prototipo de figma](#)
- FUENTES: Para las fuentes de la aplicación se deberá utilizar: **font-family: Helvetica, Arial, sans-serif;**

MODO DESARROLLO Y MODO PRODUCCIÓN

La aplicación deberá incluir dos modos:

- **Modo Desarrollo:** Servir los assets sin minimizar.
- **Modo Producción:** Servir los assets concatenados y minimizados.

PRESENTACIÓN

El objetivo final es presentar un repositorio de código público (en plataformas como GitHub o Bitbucket) con la solución desarrollada. El repositorio debe incluir un archivo README con las instrucciones para ejecutar la aplicación, una explicación de la arquitectura y estructura del proyecto, y cualquier información relevante.

DOCUMENTACIÓN Y UTILIDADES

API REST:

Puedes consultar los detalles completos de la API en el siguiente [enlace](#).

Autenticación de la API REST:

Todas las solicitudes a la API deben estar autenticadas. Para ello, es necesario incluir en el encabezado "x-api-key" el valor: **87909682e6cd74208f41a6ef39fe4191** en cada llamada.

STACK TECNOLÓGICO

- **Frontend:** React >= 17, CSS, SASS o StyledComponents.
- **Backend:** Node 18.
- **Gestión de Estado:** React Context API.
- **Autenticación:** Manejo del parámetro **x-api-key** en las peticiones.

ZARA CHALLENGE

REQUISITOS:

- Implementación de pruebas (testing).
- La aplicación debe ser responsive.
- Correcta accesibilidad.
- Uso de linters y formatters.
- La consola del navegador debe estar libre de errores y advertencias.
- Incluir un README detallado.

OPCIONAL:

- Despliegue de la aplicación.
- Uso de SSR (Server Side Rendering) con Next.js.
- Uso de variables CSS.

CONTACTO

Si tienes dudas o necesitas aclaraciones, no dudes en contactarnos a través del correo: ddfrontendzara.com@inditex.com.