**Cliente: FastBuy**

**Contexto:** La Empresa **FatsBuy** quiere extender sus operaciones a la Web, y entre los principales requerimientos son:

1. Una Página Web para mostrar sus productos, de fácil acceso para sus clientes y que estos realicen sus pedidos online y se contacten con los administradores para concretar los pagos y a la vez, dichos clientes puedan realizar el seguimiento de sus pedidos.
2. Una App móvil para la administración de dicha página web de acceso exclusivo para los administradores y personal de **FatsBuy**.
3. Que sus vendedores y demás trabajadores puedan incorporarse al giro comercial de la empresa en los nuevos puestos que se crearán con esta migración, como son los puestos de trabajos despachadores, vendedores a domicilios entre otros.

**Propuesta:** Consideramos luego de analizar los planteamientos de **FatsBuy,** facilitar su migración al mundo online a través de las siguientes tecnologías:

1. Creación de un Backend bajo la arquitectura REST, desarrollado con el Framework **Nestjs**, que corre sobre Nodejs, y que ofrece una arquitectura modular fácilmente escalable desde un monolito hasta microservicio conforme el E Commerce vaya creciendo. Este Backend sería el interlocutor entre el Frontend y la Base de Datos Postgres.
2. En la base de datos tendríamos en un principio las siguientes Entidades o Tablas: **Users, Categories, Products, Orders**, y luego de su lanzamiento se incorporarán **List Wish, Question, Cart, Newsletter**.
3. El Fronted sería construido con **Nextjs**, Framework construido sobre la librería de React, que permite la incorporación de técnicas de seo imprescindibles para esta clase de negocios, así como en el renderizado de páginas estáticas y renderizadas desde el servidor, además de otras optimizaciones que en su conjunto permitirán una navegación rápida entre las diferente páginas del ecommerce.

Por destacar algunas de las funcionalidades que tendrá este ecommerce, es la de mostrar los productos por categorías, por el tipo de oferta, mostrar el detalle de un producto, agregarlo a un carrito de compras que en principio se almacenaría en el Localstorage del usuario, y una vez presione el botón comprar se enviará esa información al backend para crear la orden y a su vez se redirigirá al usuario a una cuenta de whatsapp de los administradores para ultimar los detalles de pago.

Igualmente el usuario podrá incorporar a su lista de deseos diferentes productos, y realizar preguntas sobre los productos que le interesan y recibir la respectiva respuesta.

1. Respecto a la Aplicación Móvil, el Framework que utilizaremos será **Expo** que corre sobre la librería de React-Native, este nos permite con un mismo código desplegar en las store de IOS y Android, incluye una estandarización de los componentes de IOS y Android, un despliegue continuo y actualizaciones continuas, con herramientas que el mismo Framework nos aporta.

Entre las funcionalidades de esta app están: acceder a un Dashboard que muestra información de las ventas por fechas, las órdenes filtradas por estatus, número de usuarios y sus detalles, actualización de los roles de los usuarios, bloqueo de usuarios, entre otras funcionalidades administrativas.

1. Para la incorporación del personal de **FatsBuy** a esta nueva modalidad de trabajo, la propuesta sería construir un Customer Relationship Management, o Gestión de Relaciones con los Clientes conocido como CRM. Se trata de un software diseñado para ayudar a mejorar la experiencia del cliente, aumentar la productividad de los equipos de ventas y mejorar el seguimiento y el compromiso de los clientes. Este se desarrollaría con React.

Todas la tecnologías antes propuestas son open source lo que significa que son gratuitas y tienen detrás grandes empresas y miles de Desarrolladores que contribuyen con su mantenimiento, además que se está empleando el mismo lenguaje de programación para todo el desarrollo lo que facilita la contratación de Desarrolladores y mantenimiento a largo plazo.

Respecto a la seguridad la idea es tercerizar los servicios de autenticación, dividir las responsabilidades del frontend, la app administrativa y el CRM, cuyo acceso para estas últimas 2 sea media doble verificación, así como el empleo de json web token para el acceso a las mismas y a los endpoint.

En cuanto a la escalabilidad como se plantea al principio de la presente, se iniciará con una arquitectura monolítica modular en el backend, se desplegará dentro de un container de docker en algún servidor de la nube por ejemplo en C2 de Amazon Web Service y se contrataría un balanceador de carga que active nuevas instancias de acuerdo a la demanda o la desactive de ser el caso.