Ficha Técnica del Proyecto Móvil

Área de Desarrollo Móvil
Centro De Electricidad Y Automatización Industrial, SENA
3067594: Análisis y Desarrollo de Software
Fabian Andrés Muñoz Alegria
1 de Junio del 2026.

Cali, Valle del Cauca

Tabla de contenido

Introducción

La presente ficha técnica tiene como propósito documentar de manera integral la estructura del proyecto TAM en el área de desarrollo móvil, proporcionando una visión clara de sus características funcionales, técnicas y arquitectónicas. Este documento sirve como referencia para comprender el funcionamiento de la aplicación, los recursos tecnológicos utilizados y los lineamientos que garantizan su correcto desarrollo e implementación.

En primer lugar, se aborda la descripción funcional del sistema, detallando los procesos, servicios y funcionalidades que ofrece la aplicación móvil a los usuarios. Esto permite identificar el alcance del proyecto y la forma en que cada componente contribuye al cumplimiento de los objetivos planteados por la organización.

Asimismo, se describen los requisitos técnicos necesarios para la operación del sistema, incluyendo los lenguajes de programación, las bases de datos, las herramientas de desarrollo y la infraestructura tecnológica empleada. De igual manera, se presenta la arquitectura de software seleccionada, explicando la organización de los componentes y la comunicación entre ellos para garantizar eficiencia, mantenibilidad y rendimiento.

Finalmente, la ficha técnica contempla las integraciones con APIs y sistemas externos, así como los mecanismos de seguridad y escalabilidad implementados. Estos aspectos son fundamentales para asegurar la protección de la información, la interoperabilidad con otros servicios y la capacidad del sistema para adaptarse al crecimiento futuro y a nuevas necesidades del negocio.

1. **Descripción funcional del sistema**
   La aplicación móvil TAM es una plataforma de comercio electrónico desarrollada para dispositivos Android, que permite a los usuarios explorar, comparar y gestionar productos tecnológicos. Las funcionalidades implementadas incluyen:

   - **Autenticación:** registro de usuarios con verificación de cuenta vía correo electrónico, inicio de sesión con control de intentos fallidos y cierre de sesión.
   - **Catálogo de productos:** visualización de productos por categorías y subcategorías.
   - **Carrito de compras:** gestión de productos (agregar, modificar cantidad, eliminar), resumen de compra con subtotal y costo de envío, y persistencia local mediante AsyncStorage.
   - **Comparador de productos:** comparación de celulares, portátiles y consolas con análisis de rendimiento por características técnicas.
   - **Perfil de usuario:** visualización y edición de datos personales, dirección y teléfono.
   - **Menú de navegación (Drawer):** acceso a categorías y subcategorías cargadas dinámicamente desde la API.

2. **Requisitos técnicos (Lenguajes, BD y infraestructura)**
   Área de desarrollo Frontend:

   1. **Lenguajes y frameworks:**
   - **TypeScript** como lenguaje principal.
   - **React Native** con **Expo** como framework de desarrollo móvil.
   - **Expo Router** para la navegación basada en archivos.
   - **NativeWind** (Tailwind CSS para React Native) para estilos.

   2. **Librerías principales:**
   - **expo-linear-gradient** → gradientes visuales.
   - e**xpo-font** → fuentes personalizadas (Poppins).
   - **expo-splash-screen** → pantalla de carga inicial.
   - **@react-native-async-storage/async-storage** → persistencia local del carrito.
   - **@react-native-community/datetimepicker** → selector de fecha de nacimiento.
   - **@expo/vector-icons** → iconografía (Ionicons).

3. **Base de Datos:**

El sistema utilizará un enfoque relacional para garantizar la integridad, consistencia y seguridad de la información.

- **Motor de Base de Datos:** MySQL.
- **Descripción:** Se emplea para el almacenamiento estructurado de datos, gestión de transacciones mediante el motor **InnoDB** (soportando claves foráneas y ACID), y optimización de consultas mediante indexación.
- **Herramientas de Gestión (Opcional en desarrollo):** MySQL Workbench, phpMyAdmin o DBeaver.

Área backend

- **Lenguajes:**
  Typescript como lenguaje principal
  Express: framework

3. **Arquitectura (monolito, microservicios)**

La aplicación móvil sigue una arquitectura **cliente-servidor** desacoplada:

- **Frontend móvil (React Native/Expo):** consume servicios REST, maneja estado local con React Context API y persistencia con AsyncStorage.
- **Backend Node.js:** expone endpoints REST para productos, usuarios, categorías, carrito y autenticación.
- **Backend PHP:** gestiona el envío de correos de verificación mediante SMTP.

La **navegación** está organizada en dos grupos principales:

- (auth) → pantallas de autenticación (login, registro, verificación)
- (stack) → pantallas protegidas (home, productos, carrito, perfil, comparador)

El **estado global** se maneja mediante tres contextos:

- **AuthContext** → sesión del usuario
- **CartContext** → estado del carrito con sincronización a BD
- **CompareContext** → productos seleccionados para comparar

4. **Integraciones (APIs, sistemas externos)**

La comunicación entre el frontend y la API en TypeScript se estructura bajo el modelo **Cliente-Servidor** mediante los siguientes pilares:

- **Protocolo REST y JSON:** El frontend realiza peticiones HTTP (GET, POST, PUT, DELETE) utilizando **Axios** o **Fetch API** hacia la URL del backend. El intercambio de datos se realiza estrictamente en formato **JSON**.
- **Contratos de Datos (TypeScript):** Las interfaces y tipos definidos en el backend se comparten o replican en el frontend. Esto garantiza que ambos lados conozcan la estructura exacta de los datos (evitando errores de propiedades indefinidas).

- **Seguridad y CORS:** El backend Express tiene habilitado el middleware cors para permitir únicamente peticiones desde el dominio del frontend. La seguridad de las rutas se maneja mediante **JWT (JSON Web Tokens)** adjuntos en la cabecera de las peticiones (Authorization: Bearer <token>).
- **Control de Errores mediante Estados HTTP:** El frontend procesa las respuestas basándose en códigos estándar: 200/201 para éxito, 400 para errores de validación, 401 para redirigir al login si el token expiró, y 500 para fallos del servidor.

5. **Seguridad y escalabilidad.**

Seguridad:

- **Autenticación segura:** Uso de **JWT** (JSON Web Tokens) transmitidos en cookies seguras (HttpOnly) para gestionar sesiones.
- **Control de accesos:** Middlewares en Express para restringir rutas según el rol del usuario (**RBAC**).
- **Protección contra Inyección SQL:** Sanitización automática de datos mediante el uso estricto de **TypeScript y un ORM**.
- **Cifrado de contraseñas:** Almacenamiento seguro en MySQL utilizando *hashing* de alta seguridad con **bcrypt**.

Escalabilidad

- **Optimización de Base de Datos:** Uso de **Pool de conexiones** para reutilizar enlaces a MySQL y diseño de **índices avanzados** para consultas rápidas.
- **Arquitectura modular:** Código estructurado en TypeScript que permite separar la base de datos en réplicas de lectura/escritura si el tráfico lo requiere.

Escalabilidad del Proyecto

La app está pensada para crecer sin problemas cuando tenga más usuarios o productos.

1. Estructura modular

- El proyecto está organizado por secciones (auth, carrito, productos, perfil).
- Esto permite agregar nuevas funciones sin dañar lo existente.

2. Separación de frontend y backend

- La app móvil (frontend) puede conectarse a diferentes servicios o APIs.
- Esto facilita escalar el servidor sin cambiar la app.

3. Manejo de datos eficiente

- Los productos, carrito y usuarios se manejan de forma dinámica.
- Se pueden conectar a bases de datos grandes sin afectar el rendimiento.

4. Uso de servicios en la nube (opcional)

- Se puede integrar con servicios como Firebase o servidores cloud.
- Esto permite soportar muchos usuarios al mismo tiempo.

5. Optimización de carga

- Las imágenes y productos se cargan sólo cuando se necesitan.

- Esto hace la app más rápida y ligera.

6. Posibilidad de crecimiento

- Se pueden agregar fácilmente nuevas funciones como:
  pagos en línea
  notificaciones push
  recomendaciones de producto

Conclusión

En conclusión, el proyecto TAM es una aplicación que cumple bien con lo que se propone, ya que permite a los usuarios interactuar de forma fácil y segura. Se tuvieron en cuenta aspectos importantes como la protección de la información y el buen funcionamiento del sistema.

Además, está pensada para crecer en el futuro, permitiendo agregar nuevas funciones o más usuarios sin que la aplicación se vuelva lenta o complicada. En general, es un proyecto sólido, útil y preparado para seguir mejorando.