**7. Vista de Desarrollo**

La vista de desarrollo proporciona una perspectiva técnica de la organización interna del sistema desde el punto de vista de los desarrolladores. Esta sección detalla cómo se estructuran los paquetes de código y la organización de archivos, especificando las herramientas utilizadas en el proceso de desarrollo y los estándares de codificación aplicados para asegurar consistencia y calidad en el código.

* **Diagrama de organización del código:**  
  (Incluir un diagrama de paquetes o módulos según el lenguaje y las herramientas utilizadas, por ejemplo, estructura de paquetes en Java, módulos en Node.js, etc.)

**Estructura de Carpetas y Paquetes**

La estructura de carpetas está organizada para facilitar la modularidad, la claridad y la mantenibilidad del código. La división de archivos sigue una estructura lógica que separa el frontend y el backend, asegurando que el código sea fácil de entender y extender.

1. **Frontend (Aplicación Móvil):**
   * **/src:** Contiene el código fuente de la aplicación móvil, incluyendo los componentes de interfaz de usuario y la lógica de presentación.
   * **/assets:** Almacena los recursos estáticos, como imágenes, íconos y estilos.
   * **/services:** Define los servicios que interactúan con el backend, como solicitudes de API y control de respuestas.
   * **/models:** Incluye las clases o interfaces que representan los datos de la aplicación.
   * **/utils:** Funciones de utilidad para reutilización en varias partes del frontend.
2. **Backend:**
   * **/controllers:** Define la lógica de control para gestionar las solicitudes entrantes y coordinar las respuestas.
   * **/models:** Contiene los modelos de datos, que definen la estructura de las entidades en la base de datos.
   * **/routes:** Gestiona las rutas de API y define los puntos de acceso para las interacciones del frontend.
   * **/services:** Módulos que encapsulan la lógica de negocio y los procesos del backend.
   * **/middleware:** Define el middleware para autenticación, autorización y manejo de errores.
   * **/config:** Archivos de configuración, incluyendo credenciales y configuraciones de entorno.
   * **/tests:** Almacena las pruebas unitarias y de integración.

**Herramientas de Desarrollo**

* **Entornos de Desarrollo:**
  + **Frontend:** Android Studio para el desarrollo móvil nativo o herramientas de desarrollo multiplataforma como Flutter o React Native.
  + **Backend:** Visual Studio Code o cualquier otro editor de texto avanzado compatible con Node.js, Express, y el framework utilizado en el servidor.
* **Control de Versiones:**  
  Git y GitHub para la gestión de versiones y el control de cambios. Se utiliza un flujo de trabajo de ramas (branches) para mantener el código organizado y facilitar el desarrollo colaborativo.

**Guía de Estándares de Codificación y Convenciones**

Para asegurar consistencia en el código, se aplican estándares de codificación y convenciones de nomenclatura:

1. **Convenciones de Nombres:**
   * Uso de camelCase para variables y funciones.
   * PascalCase para nombres de clases y componentes.
   * snake\_case para archivos y carpetas.
2. **Estandarización del Formato de Código:**
   * Uso de linters como ESLint o Prettier para mantener un formato de código uniforme y minimizar errores.
   * Documentación de funciones y métodos clave para facilitar la comprensión y el mantenimiento.
3. **Comentarios y Documentación:**
   * Se documenta el código siguiendo la convención de JSDoc (o similar), que facilita la generación automática de documentación técnica.
   * Uso de comentarios para clarificar la lógica en partes complejas del código, sin sobrecargar el texto.
4. **Pruebas de Código:**
   * El sistema de pruebas incluye pruebas unitarias y de integración en el backend para asegurar que cada módulo funcione correctamente.
   * Herramientas como Jest o Mocha/Chai para pruebas automatizadas en JavaScript, y suites de pruebas específicas para Android o iOS en el frontend.

La vista de desarrollo proporciona a los desarrolladores una estructura clara y consistente, garantizando un entorno de trabajo organizado y eficiente para el desarrollo y mantenimiento del sistema. Esta organización facilita tanto la colaboración entre equipos como la escalabilidad futura del proyecto.