**CUADRO COMPARATIVO: Caso 4 vs. Caso 5 con Recomendación para App Móvil**

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspecto** | **Caso 4: Flask + MySQL (Backend Web)** | **Caso 5: Kivy + SQLite + ML (App Local)** | **¿Cuál es mejor para una app móvil?** |
| **Tipo de Base de Datos** | MySQL (servidor externo) | SQLite (archivo local) | ✅ **Caso 4**, ya que es más escalable y permite sincronización en red |
| **Librería de Conexión a BD** | SQLAlchemy + PyMySQL | Use la librería sqlite3, pero si quisiera manejar los datos desde un servidor, debo isa SQLAlchemy + PyMySQL | ✅ **Caso 4**, más profesional y adaptable a cambios |
| **Dónde se ejecuta** | Servidor (accesible desde cualquier cliente móvil) | App local (en el mismo dispositivo del usuario) | ✅ **Caso 4**, se adapta mejor a apps móviles modernas |
| **Interfaz de usuario** | No tiene (solo API REST) | Tiene interfaz gráfica (Kivy) | ⚠️ **Caso 5**, si tu app es muy simple y no necesita conectarse a internet |
| **Conexión a servidor externo** | Sí (ideal para apps móviles que sincronizan datos) | No | ✅ **Caso 4**, esencial para apps colaborativas o multiusuario |
| **Acceso vía API REST** | Sí, por métodos HTTP (POST, GET, etc.) | No, solo acceso local a archivos | ✅ **Caso 4**, ya que permite integrar fácilmente con apps móviles |
| **Machine Learning incluido** | ❌ No | ✅ Sí | ⚠️ **Caso 5**, si el análisis local ML es clave, aunque puede integrarse en ambos |
| **Tecnologías adicionales** | Flask, JSON, SQLAlchemy | Kivy, Pandas, scikit-learn | ⚠️ Depende del caso: **Caso 4** es más común en producción |
| **Escalabilidad / Producción** | Alta. Ideal para producción real | Limitada a dispositivos individuales | ✅ **Caso 4**, sin dudas |
| **Modo de integración con App móvil** | Ideal como **backend RESTful** para apps móviles | Funciona como app **autónoma**, pero no se comunica con otras | ✅ **Caso 4**, es la arquitectura estándar en apps móviles |
| **Requiere conexión a internet** | Sí (aunque puedes trabajar en local con localhost) | No | ⚠️ Depende del uso. **Caso 5** si solo usas funciones locales sin sincronización |
| **Seguridad y control de acceso** | Sí, puede usar autenticación, tokens, roles, etc. | No tiene autenticación integrada | ✅ **Caso 4**, esencial en apps reales |
| **Ideal para...** | Apps móviles modernas con base de datos centralizada y usuarios múltiples | Prototipos, apps personales o educativas sin sincronización | ✅ **Caso 4**, más flexible y profesional |

**Conclusión: ¿Cuál usar para diseñar una app móvil real?**

| **Escenario** | **Elección recomendada** |
| --- | --- |
| App profesional, con múltiples usuarios, login, sincronización, backend, analíticas, escalabilidad | ✅ **Caso 4: Flask + MySQL** como base del backend, conectada a una app móvil (Flutter, React Native, etc.) |
| App personal, sin login, sin conexión, todo local en el teléfono/tableta (tipo agenda o calculadora) | ⚠️ **Caso 5: Kivy + SQLite**, aunque muy limitada y no apta para escalar o publicar |