📘 Guía de Lanzamiento - App Streamlit de Gestión Airbnb

# 1. Enfoque general del proyecto

Esta aplicación ha sido desarrollada para gestionar y analizar financieramente apartamentos turísticos estilo Airbnb. Su objetivo es centralizar reservas, ingresos, gastos e inversión, brindando al propietario una visión clara de la rentabilidad y métricas clave.

Público objetivo: propietarios o gestores de propiedades.

Valor diferencial: Automatización de gastos, análisis ROI, visualización clara y editable.

# 2. Estructura de la app y funcionalidades

Cada archivo .py representa una pestaña dentro del dashboard Streamlit:

* 📊 General.py: Vista rápida de ingresos, beneficios y ocupación actual.
* 📅 Calendario.py: Muestra calendario de reservas por propiedad/habitación.
* 📝 Reservas.py: Formulario para añadir, editar y listar reservas.
* 💸 Gastos.py: Registro de gastos fijos y gastos variables automáticos por reserva.
* 🏘️ Listings.py: Tarjetas visuales con fotos y resumen de propiedades activas.
* 📉 Inversion.py: Registro de inversión inicial + análisis ROI + métricas clave + editor.
* 📐 Simluacion.py: Modo expansión (simular compra de nuevos listings y su impacto).

# 3. Librerías requeridas

Incluidas en requirements.txt:

* streamlit
* firebase-admin
* pandas
* python-decouple
* datetime, os, etc.

# 4. Conexión con Firebase

Se requiere el archivo `secrets.toml` con credenciales de Firebase. Las principales colecciones en uso son:

* bookings
* gastos\_fijos
* inversiones
* habitaciones
* imagenes\_listings

# 5. Imágenes en tarjetas Listings

Puedes usar URLs externas (como ImgBB) o subir imágenes y enlazarlas desde Firebase Storage si se desea mayor privacidad.

# 6. Métricas clave calculadas

* ROI total y por mes
* Meses para recuperar la inversión (break-even)
* Ingresos acumulados
* Ingreso mensual promedio
* Gastos mensuales y fijos

# 7. Formulario editable de inversión

La sección inferior permite editar manualmente los gastos fijos o inversiones registradas. Esto asegura control absoluto del ROI visualizado.

# 8. Flujo de desarrollo (modo local)

1. Iniciar entorno virtual y ejecutar `streamlit run home.py`.
2. Limpiar caché con `streamlit cache clear` si los cambios no se reflejan.
3. Cada módulo .py está aislado para facilitar el mantenimiento.

# 9. Despliegue en Streamlit Cloud

Pasos:

1. Subir el código a GitHub.
2. Conectar tu repo en Streamlit Cloud.
3. Añadir `secrets.toml` desde la interfaz web con las claves de Firebase.
4. Seleccionar `home.py` como archivo principal.

# 10. Mantenimiento y versión

Para actualizar:

* Editar el código localmente y hacer push a GitHub.
* Mantener control de versiones con comentarios claros en commits.
* Actualizar `requirements.txt` si añades nuevas librerías.