# informe de Progreso y Configuración del Proyecto LoanApp

El presente documento detalla el proceso de configuración inicial del proyecto LoanApp, con un enfoque particular en la resolución de desafíos relacionados con las dependencias y la habilitación de la generación de código para la persistencia de datos mediante la biblioteca Hive.

## 1. Contexto Preliminar: Establecimiento del Proyecto y Dependencias Fundamentales

En la fase inicial de desarrollo del proyecto LoanApp en el entorno Flutter, se procedió a la definición de las dependencias esenciales dentro del archivo pubspec.yaml. Estas dependencias fueron seleccionadas para habilitar funcionalidades críticas, tales como la gestión de bases de datos (Hive), la generación de identificadores únicos (uuid) y la localización de formatos (intl).

La configuración inicial de las dependencias en pubspec.yaml se presentaba de la siguiente manera.

-----------PRIMERA PARTE: LUNES 21 DE JULIO 2025---------

## 2. Desafío Principal: Interrupción en la Generación de Código (client\_model.g.dart)

El obstáculo más significativo encontrado fue la falla recurrente en la ejecución del comando dart run build\_runner build --delete-conflicting-outputs. Este comando es indispensable para la creación del archivo client\_model.g.dart, el cual es fundamental para la correcta adaptación de los modelos de datos de Hive.

Los mensajes de error iniciales indicaban una incompatibilidad de versiones entre el SDK de Dart y el paquete analyzer, una dependencia crítica de build\_runner y hive\_generator. Un ejemplo de dicho mensaje era: W SDK language version 3.8.0 is newer than analyzer language version 3.7.0.

Posteriormente, los intentos de forzar versiones específicas mediante anulaciones de dependencias resultaron en errores más complejos, tales como: Error: Error when reading '../../.pub-cache/hosted/pub.dev/build-2.4.1/lib/src/internal.dart': No such file or directory Error: Type 'ExtensionTypeDeclaration' not found. Error: The getter 'libraryImports' isn't defined for the class 'CompilationUnitElement'.

Estos errores denotaban una **desalineación sustancial en las interfaces de programación de aplicaciones (APIs) internas** entre los componentes del ecosistema build\_runner (build\_runner\_core, build, build\_resolvers, dart\_style, analyzer).

## 3. Metodología de Resolución de Problemas (Enfoque Iterativo)

Se adoptó una metodología iterativa para la resolución de los problemas, que comprendió las siguientes acciones:

* **Ajustes de Versión en** pubspec.yaml**:**
  + Se realizaron intentos de actualizar las versiones de flutter\_lints, intl, hive\_generator y build\_runner a sus últimas versiones estables.
  + Se experimentó con diferentes rangos de versión para el sdk en la sección environment.
* **Implementación de** dependency\_overrides**:**
  + Se introdujo una sección dependency\_overrides con el fin de forzar versiones específicas de analyzer (por ejemplo, ^6.2.0 y posteriormente ^5.0.0), así como de otros paquetes internos (build, build\_resolvers, source\_gen, dart\_style).
  + **Hallazgo Crítico:** Se identificó que una ubicación incorrecta de la sección dependency\_overrides (tratándola como una dependencia regular en lugar de una sección raíz) provocó un error inicial. Una vez corregido este aspecto, las anulaciones continuaron generando conflictos de API entre las versiones forzadas y las esperadas por otros paquetes. Este hecho subrayó que la imposición de versiones internas de esta manera puede comprometer la compatibilidad del ecosistema build\_runner.
* **Procedimientos de Limpieza Exhaustiva:**
  + Se llevaron a cabo operaciones de limpieza profunda de la caché de paquetes y del proyecto para eliminar artefactos persistentes que pudieran causar conflictos:
    - flutter pub cache clean --force
    - flutter clean
    - rm pubspec.lock (o su equivalente para sistemas operativos Windows).
* **Actualizaciones y Reversiones de Versión de Flutter:**
  + Se ejecutó flutter upgrade para garantizar la utilización de la última versión estable del SDK de Flutter y Dart.
  + Se realizaron intentos de flutter downgrade a versiones específicas (por ejemplo, 3.13.9) con el propósito de determinar si una versión anterior de Flutter/Dart ofrecía mayor compatibilidad con las versiones disponibles de hive\_generator y build\_runner.

## 4. Solución Definitiva: Armonización del Ecosistema build\_runner

La estrategia que resultó ser más efectiva y estable consistió en **simplificar la configuración del** pubspec.yaml **y delegar al sistema de gestión de paquetes de Dart (**pub**) la responsabilidad de resolver automáticamente las dependencias de** build\_runner, absteniéndose de interferencias mediante dependency\_overrides para sus componentes internos.

El pubspec.yaml final que permitió la exitosa generación de código es el siguiente.

Tras la implementación de esta configuración y una limpieza profunda, el comando flutter pub get logró resolver las dependencias sin errores de conflicto significativos, y la ejecución de dart run build\_runner build --delete-conflicting-outputs se realizó con éxito.

**Advertencia Persistente:** Se observó una advertencia informativa recurrente: W SDK language version 3.8.0 is newer than analyzer language version 3.7.0. Run flutter packages upgrade. Esta advertencia indica una ligera desincronización entre la versión del SDK de Dart y la versión del paquete analyzer utilizada por build\_runner debido a las restricciones de dependencia. Sin embargo, se ha determinado que esta advertencia no impide la funcionalidad de generación de código.

## 5. Configuración del Punto de Entrada de la Aplicación (main.dart)

Finalmente, se procedió a la configuración del archivo lib/main.dart, el cual sirve como el punto de entrada principal de la aplicación LoanApp. Esta configuración incluyó la inicialización de la base de datos de la aplicación y el establecimiento de la pantalla de configuración de PIN como la interfaz de usuario inicial.

El archivo main.dart ahora realiza las siguientes funciones:

* Inicialización de los bindings de Flutter.
* Inicialización de la base de datos de la aplicación (asumiendo la configuración de Hive o un sistema similar).
* Lanzamiento de la aplicación MyApp.
* Establecimiento de PinSetupScreen() como la pantalla inicial de la aplicación.

## 6. Estado Actual del Proyecto y Próximas Fases

El proyecto LoanApp se encuentra actualmente en un estado operativo, lo que facilita el avance en su desarrollo.

* **Generación de Código:** Las herramientas build\_runner y hive\_generator operan eficientemente, permitiendo la creación y actualización de los archivos .g.dart requeridos por Hive.
* **Resolución de Dependencias:** Las dependencias del proyecto han sido resueltas en una configuración compatible, y las advertencias sobre versiones más recientes disponibles no comprometen la funcionalidad principal.
* **Estructura Inicial:** El archivo main.dart ha sido configurado adecuadamente para la lógica de la aplicación, estableciendo una base sólida para la implementación de funcionalidades adicionales.

La siguiente fase del proyecto implica la continuación del desarrollo de las características fundamentales de la aplicación, tales como la gestión de clientes, la administración de préstamos y el procesamiento de pagos, haciendo uso de la infraestructura de base de datos Hive ya establecida.