**Roadmap extendido para tu TFC:**

**Fase 1: Planificación y definición del alcance**

1. **Definir el objetivo de la aplicación**:
   * Juego de cartas coleccionables con apertura de sobres, gestión de colección y mercado.
2. **Funcionalidades clave**:
   * Registro e inicio de sesión (Firebase Authentication).
   * Apertura de sobres de cartas con animaciones.
   * Visualización y gestión de la colección de cartas del usuario.
   * Mercado para compra/venta de cartas entre usuarios.
   * Sistema de recompensas y moneda virtual (ej. monedas o gemas).
   * Posibilidad de añadir eventos especiales (sobres limitados, cartas raras).
3. **Establecer el stack de tecnologías**:
   * **Frontend**: Flutter
   * **Backend**: Firebase (Firestore, Authentication, Storage, Cloud Functions)
   * **Gestión de estado**: Provider o Riverpod para la gestión global.
   * **Integraciones futuras**: Posible integración con APIs de terceros (p. ej., para analizar el mercado de cartas o clasificación).

**Fase 2: Configuración inicial del proyecto**

1. **Configuración del proyecto Flutter**:
   * Crear un nuevo proyecto Flutter.
   * Configurar dependencias esenciales (firebase\_core, firebase\_auth, cloud\_firestore, firebase\_storage, provider o riverpod, animations).
   * Configuración de Firebase para autenticación y Firestore.
2. **Conexión con Firebase**:
   * Firebase Authentication para inicio de sesión (email, Google, etc.).
   * Cloud Firestore para la base de datos en tiempo real.
   * Firebase Storage para almacenar imágenes de las cartas.
   * Configuración de reglas de seguridad de Firestore y Storage.
3. **Diseño básico de la base de datos**:
   * Colección de **Usuarios**: Datos como nombre, email, monedas, colección de cartas, etc.
   * Colección de **Cartas**: Nombre, rareza, imagen, poder, etc.
   * Colección de **Mercado**: Listados de compra/venta de cartas.

**Fase 3: Implementación de autenticación y gestión de usuarios**

1. **Pantalla de registro e inicio de sesión**:
   * Diseño e implementación de la interfaz de usuario para registro/login.
   * Autenticación con email/contraseña y posiblemente con Google (usando Firebase Auth).
   * Integración de funciones para restablecer contraseña.
2. **Perfil de usuario**:
   * Crear y mostrar el perfil del usuario, con opciones para editar el perfil.
   * Almacenar monedas virtuales que el usuario usa para comprar sobres o cartas en el mercado.

**Fase 4: Sistema de cartas y apertura de sobres**

1. **Definir el conjunto de cartas**:
   * Crear un conjunto inicial de cartas con atributos como nombre, rareza, imagen y poder.
   * Almacenar las cartas en Firestore y las imágenes en Firebase Storage.
2. **Sistema de apertura de sobres**:
   * Diseño de la pantalla para abrir sobres con animaciones (usando AnimatedContainer o paquetes de animaciones como flutter\_animate o lottie).
   * Lógica para asignar cartas aleatorias de distintos niveles de rareza (p. ej., 80% comunes, 15% raras, 5% legendarias).
   * Actualización de la colección de cartas del usuario después de abrir un sobre.
3. **Almacenamiento de la colección**:
   * Guardar las cartas obtenidas en la colección del usuario en Firestore.
   * Lógica para acumular múltiples copias de la misma carta.

**Fase 5: Visualización de la colección de cartas**

1. **Pantalla de colección de cartas**:
   * Implementar una interfaz con un **GridView** que muestre las cartas del usuario.
   * Opción para filtrar las cartas por rareza, tipo, poder, etc.
   * Mostrar información detallada de una carta cuando se selecciona (nombre, imagen, estadísticas).
2. **Interacciones con las cartas**:
   * Opción de "marcar como favorita", organizar o buscar cartas en la colección.
   * Posibilidad de **destruir** cartas para obtener monedas o recursos.

**Fase 6: Mercado de cartas**

1. **Pantalla del mercado**:
   * Crear una interfaz que muestre cartas en venta por otros usuarios (usando **ListView** o **GridView**).
   * Implementar filtros (por precio, rareza, etc.).
   * Mostrar detalles de cada carta y opciones para comprar.
2. **Sistema de compra/venta**:
   * Lógica para que los usuarios puedan listar cartas a la venta: fijar un precio y añadir la carta al mercado.
   * Lógica de transacciones: cuando un usuario compra una carta, actualizar el saldo de monedas y transferir la carta al comprador.
   * Actualización de las colecciones de cartas de ambos usuarios.

**Fase 7: Sistema de economía y recompensas**

1. **Moneda virtual**:
   * Crear un sistema de monedas o gemas que los usuarios pueden usar para comprar sobres o cartas.
   * Lógica para obtener monedas a través de eventos o destruyendo cartas.
   * Mostrar el saldo de monedas del usuario en la interfaz.
2. **Sistema de recompensas diarias**:
   * Implementar un sistema de recompensas por iniciar sesión diariamente o completar desafíos.
   * Guardar el progreso en Firestore y actualizar las recompensas.

**Fase 8: Eventos especiales y sobres limitados**

1. **Sobres de edición limitada**:
   * Crear sobres especiales que solo estén disponibles por tiempo limitado (p. ej., con cartas raras exclusivas).
   * Implementar un temporizador y lógica para permitir la compra de estos sobres mientras estén disponibles.
2. **Eventos en el juego**:
   * Lógica para implementar eventos especiales como torneos o retos que ofrezcan cartas exclusivas.

**Fase 9: Implementación de Firebase Functions (Opcional)**

1. **Backend para tareas específicas**:
   * Usa Firebase Functions para manejar la lógica compleja del mercado o la validación de transacciones.
   * Lógica para eventos programados, por ejemplo, reseteo diario de recompensas o sobres.

**Fase 10: Integración de animaciones y mejora de UI/UX**

1. **Mejoras visuales**:
   * Animaciones en la apertura de sobres (posible uso de **Lottie** para animaciones).
   * Transiciones suaves entre pantallas (usando el paquete flutter\_animate o page\_transition).
2. **Optimización de la interfaz**:
   * Ajustar el diseño para que sea visualmente atractivo y fácil de usar.
   * Implementación de un **tema oscuro** opcional.

**Fase 11: Pruebas y optimización**

1. **Pruebas de rendimiento**:
   * Realizar pruebas de rendimiento en diferentes dispositivos (usando **Firebase Test Lab** o simuladores).
2. **Pruebas de usabilidad**:
   * Realizar pruebas de usuario para obtener retroalimentación sobre la interfaz y el flujo del juego.
3. **Corrección de errores**:
   * Revisar el código y corregir errores identificados durante las pruebas.

**Fase 12: Publicación y mantenimiento**

1. **Preparación para la publicación**:
   * Optimizar la app para Android (y, si es multiplaforma, para iOS).
   * Preparar los recursos gráficos para la tienda (íconos, capturas de pantalla, etc.).
   * Configuración de la app en Google Play Store (y App Store si es aplicable).
2. **Mantenimiento post-lanzamiento**:
   * Monitorear el rendimiento de la app en Firebase Analytics.
   * Recoger feedback de los usuarios y aplicar actualizaciones con mejoras o nuevas funcionalidades.