Manual Técnico: Aplicación de Reto Matematico

1. Arquitectura de la Aplicación:
La aplicación está desarrollada utilizando el framework Flutter, que permite crear aplicaciones nativas para iOS y Android desde un solo código base.
La arquitectura de la aplicación sigue el patrón de diseño MVC (Modelo-Vista-Controlador) para separar la lógica de negocio, la interfaz de usuario y el manejo de eventos.

2. Tecnologías Utilizadas:
Flutter: Framework de desarrollo de aplicaciones móviles multiplataforma.
Dart: Lenguaje de programación utilizado para escribir el código de la aplicación.
Widgets: Componentes de la interfaz de usuario utilizados para construir la interfaz de usuario y gestionar la interacción del usuario.
Temporizadores: Se utilizan temporizadores para controlar el tiempo de visualización de los números aleatorios y el tiempo para que el usuario responda.
Generación de Números Aleatorios: Se utiliza la librería dart:math para generar números aleatorios en el rango del 1 al 20.

3. Estructura del Código:
Modelo: Contiene la lógica de negocio de la aplicación, incluyendo la generación de números aleatorios y la verificación de las respuestas del usuario.
Vista: Contiene los widgets y la interfaz de usuario de la aplicación.

4. Flujo de la Aplicación:
El usuario inicia la aplicación y accede a la pantalla principal.
Al presionar el botón "Iniciar Juego", se inicia el temporizador y se generan números aleatorios durante 5 segundos.
Después de 5 segundos, los números aleatorios desaparecen, y el usuario debe ingresar la suma en el campo de entrada.
El controlador verifica la respuesta del usuario y decide si se pasa al siguiente nivel o se muestra un mensaje de error.
Este proceso se repite para cada nivel, aumentando la dificultad con sumas más difíciles.

5. Gestión del Tiempo:
Se utiliza un temporizador para controlar el tiempo de visualización de los números aleatorios.
El temporizador se inicia cuando el usuario presiona el botón "Iniciar Juego" y se detiene después de 5 segundos.

6. Pruebas Unitarias y de Integración:
Se han implementado pruebas unitarias para verificar la funcionalidad de las diferentes partes de la aplicación, incluyendo la generación de números aleatorios y la verificación de las respuestas del usuario.
Se han realizado pruebas de integración para garantizar que los componentes de la aplicación funcionen correctamente juntos.

7. Mejoras Futuras:
Se pueden agregar características adicionales, como ajustes de dificultad, opciones de personalización.
Se pueden realizar mejoras en el rendimiento y la eficiencia del código para garantizar una experiencia de usuario fluida.