Estándar de Código para el Proyecto A.V.U.

Convención de nomenclatura:

Variables:

- Camel Case: Las variables se comienzan con minúscula y las palabras subsecuentes comienzan con mayúscula. Ejemplo: nombreVariable.
- Su nombre debe ser descriptivo de su función

Funciones y Métodos:

- Camel Case: Las funciones y métodos comienzan con minúscula y las palabras subsecuentes comienzan con mayúscula.
- Su nombre debe ser descriptivo de su función

Clases:

- Camel Case con la primera letra en mayúscula: Ejemplo: ClaseEjemplo.

Constantes:

En mayúsculas con palabras separadas por guiones bajos. Ejemplo: PI = 3.14159265359.

Pantallas:

El nombre de las pantallas deben de describir que pantalla es.

Indentación y formato:

Comas:

Coloca una coma seguida de un espacio después de cada elemento en una lista, como parámetros de función o elementos de una matriz.

Paréntesis:

Deja un espacio después de los paréntesis en las declaraciones de funciones y control de flujo.

Comentarios:

- Los comentarios se usarán para explicar la lógica de funciones difíciles o de la implementación de nuevas funciones para que cada integrante este al tanto del funcionamiento de la aplicación.

Estructura del proyecto:

Por el momento el proyecto se divide en 3 archivos:

- Screens (Pantalla de usuario)
- AdminScreen(Pantalla de Admin)
- Assets (Imagenes)

Manejo de excepciones y errores:

El código estará escrito con try-catch para así capturar excepciones y errores. De esta manera evitaremos que la aplicación se caiga por completo y también podremos saber que error tiene la aplicación.

Control de Versiones

Nomenclatura de Ramas:

- Las ramas deben tener nombres descriptivos y significativos que indiquen su propósito.
- Las ramas de características pueden seguir el formato "feature/nombre-de-la-característica".
- Las ramas de corrección de errores pueden seguir el formato "bugfix/nombre-del-bug".
- Las ramas de lanzamiento pueden seguir el formato "release/version".
- Las ramas principales como main o master deben ser protegidas y solo se pueden actualizar mediante solicitudes de extracción (pull requests).

Commits:

- Los commits deben ser atómicos, es decir, cada uno debe representar un cambio lógico y completo.
- Utiliza mensajes de confirmación descriptivos y concisos que comiencen con un verbo en presente. Por ejemplo, "Agrega funcionalidad de inicio de sesión" o "Corrige error de validación en formulario".
- Divide los cambios en varios commits si es necesario para mantener la claridad y facilitar la revisión de código.

Solicitudes de Extracción (Pull Requests):

- Antes de fusionar una rama, debe crearse una solicitud de extracción.
- La solicitud de extracción debe incluir una descripción detallada de los cambios realizados.
- Deben solicitarse revisiones por parte de otros miembros del equipo.
- No se permite la fusión de una rama sin aprobación y revisión adecuada.

Fusiones (Merges):

- Los merges a la rama principal deben hacerse mediante fusiones rápidas o rebase si es posible para mantener un historial limpio y lineal.
- Resuelve cualquier conflicto de fusión antes de completar la fusión.

Etiquetas (Tags):

- Utiliza etiquetas para marcar versiones significativas del software, como "v1.0.0".
- Las etiquetas deben incluir una descripción que indique los cambios realizados en esa versión.

Colaboración:

- Los miembros del equipo deben de estar de acuerdo antes de cualquier merge a la rama principal.