UNIVERSIDAD DON BOSCO

FACULTAD DE INGENIERÍA

ESCUELA DE COMPUTACIÓN

ASIGNATURA:

DESARROLLO DE SOFTWARE PARA MÓVILES

Trabajo de investigación 2

INTEGRANTES:	**CARNÉ:**
GIOVANNY MISAEL ABREGO HERRERA	AH132090
RODRIGO ERNESTO HERNÁNDEZ RIVAS	HR132106

CATEDRÁTICO:

ING. ALEXANDER ALBERTO SIGUENZA CAMPOS

FECHA: 9 DE ABRIL DE 2022

Arquitectura CLEAN

Una arquitectura limpia es aquella que pretende conseguir unas estructuras modulares bien separadas, de fácil lectura, limpieza del código y testabilidad.

Características.

- Independientes del framework utilizado.
- Testeables.
- Independientes de la interfaz gráfica.
- Independientes de los orígenes de datos.
- Independientes de factores externos.

Elementos que conforman la arquitectura.

- Entidades.
- Casos de uso.
- Adaptadores de interfaz.
- Frameworks y Drivers.

Principios Solid

Principios para la programación orientada a objetos.

- S - Principio de responsabilidad única.

  Una clase debe encapsular una única funcionalidad

- O - Principio de ser abierto y cerrado.

  Diseñar código que esté abierto a extensiones y cerrado a modificaciones.

- L - Principio de sustitución de Liskov.

  Toda subclase que herede de una clase padre debe realizar su función sin alterar el funcionamiento de la clase padre.

- I - Principio de segregación de interfaz.

  Asegurar que las interfaces utilizadas en las clases extendidas contengan exclusivamente los métodos requeridos para el funcionamiento de la clase, si existen métodos que no se utilizan separar la interfaz en múltiples interfaces.

- D - Principio de inversión de dependencia.

  Utilizar instancias o abstracciones de forma que la funcionalidad dependa de estos objetos individuales y no de referencias anidadas entre diferentes clases.

Patrones de Diseño

Los patrones de diseño tratan de resolver los problemas relacionados con la interacción entre interfaz de usuario, lógica de negocio y los datos.

Los patrones más utilizados para la resolución de este tipo de problemas son:

- MVC (Modelo Vista Controlador)

  El usuario interactúa con la vista realizando una acción, esta acción es atendida por el controlador, seguidamente, el controlador solicita al modelo obtener o modificar los datos solicitados, en el modelo, estos datos son atendidos por el controlador, que los devuelve a la vista seleccionada para presentarlos por pantalla.

  - Modelo.

    Esta capa contiene el conjunto de clases que definen la estructura de datos con los que vamos a trabajar en el sistema.

  - Vista.

    Esta capa contiene la interfaz de usuario de nuestra aplicación.

  - Controlador.

    Esta capa es la intermediaria entre la Vista y el Modelo.

- MVP (Modelo Vista Presentador)

  En este patrón, toda la lógica de la presentación de la interfaz reside en el Presentador, de forma que este da el formato necesario a los datos y los entrega a la vista para que esta simplemente pueda mostrarlos sin realizar ninguna lógica adicional

  - Modelo.

    Es la capa encargada de gestionar los datos.

  - Vista.

    Es una interfaz de comportamiento de lo que podemos realizar con la vista, servirá de puente de comunicación entre el presentador y las Activity.

- Presentador.

  Es la capa que actúa como intermediaria entre el modelo y la vista.