**PRUEBAS DE CALIDAD DE SOFTWARE:**

**¿Por qué hacer pruebas de calidad?**

Por qué nos permiten verificar y revelar la calidad de un producto software. Son utilizadas para identificar posibles fallos de implementación, calidad, o usabilidad de un programa, la cual, se integran dentro de las diferentes fases del ciclo del software dentro de la Ingeniería de software. Así se ejecuta un programa y mediante técnicas experimentales se trata de descubrir que errores tiene.

**Objetivos de las Pruebas de Software**

* Probar si el software no hace lo que debe.
* Probar si el software hace lo que no debe, es decir, si provoca efectos secundarios adversos.
* Descubrir un error que aún no ha sido descubierto.
* Encontrar el mayor número de errores con la menor cantidad de tiempo y esfuerzo posibles.
* Mostrar hasta qué punto las funciones del software operan de acuerdo con las especificaciones y requisitos del cliente.

**Tipo de Pruebas a Utilizarse.**

**Prueba Unitaria:** Este tipo de prueba nos ayudara a nivel de código para saber que cada una de nuestras clases realice su tarea asignada, también que reciba los parámetros adecuados, para asegurar que los resultados obtenidos sea los esperados, es decir, aquí se aplicara en cada uno de los módulos como las pruebas para iniciar sesión, para registrar un usuario, para comentar, para enviar mensajes y también para que se publiquen de manera correcta. Esto es fundamental para obtener información de la calidad del producto. En la prueba unitaria se prueban las funcionalidades sin interaccionar con otros sistemas.

**Prueba de Integridad:** Este tipo de prueba, es muy importante dado a que estamos desarrollando una aplicación móvil, entonces lo que se hará es verificar la robustez de la compatibilidad con la API de datos para obtener información del servidor, y la calidad, disponibilidad e integridad de la información ofrecida por el API, de esta forma asegurar que nuestra aplicación no ocurra inconvenientes al usuario, es decir, que encapsula los elementos de comunicación.

**Las pruebas funcionales:** En este tipo de prueba, la aplicación debe ser capaz de guardar, tanto de forma efectiva como selectiva, la información, dejando en el servidor aquellos datos que precisen de recursos adicionales. También, es necesario observar el comportamiento y fallos de la aplicación cuando pueda verse afectada por una situación de batería baja, entrada de llamadas o mensajes mientras realizamos una operación con la misma.

**Pruebas de carga:** En este tipo de pruebas, lo que haremos es someter la aplicación móvil a diferentes dispositivos dado a que el mercado de hoy en día existen cientos de dispositivos con diferentes especificaciones tanto de hardware como de software. No se puede determinar con certeza si nuestras pruebas a van a cumplir para todos los casos, pero se puede asegurar que cumplan para la mayoría, las mejores pruebas con las que simulan el comportamiento del usuario final, ya que esto va a determinar los problemas de UI.

Además de las anteriores mencionadas también se toma en cuenta la **Prueba de Caja Blanca:** En éste tipo de prueba, se encuentran cuestiones relacionadas a: fallas lógicas, caminos mal estructurados en el código, el flujo de entradas específicas a través del código, funcionalidad de ciclos y condiciones, hoyos de seguridad interna y permite probar cada objeto y función de manera individual. También la **Prueba de Caja Negra:** Con la versión descargable de manera pública de la aplicación. Se verifica la parte IU de la aplicación simulando su comportamiento en diferentes entornos o diferentes situaciones donde el usuario trata de inserta datos erróneos, y demás característica de la manera que encuentre errores en la parte gráfica.

**Herramientas para las pruebas:**

Las pruebas unitarias se realizan mediante el uso de Android Junit Test una extensión del conocido framework Junit adaptado a Android. El código fuente se programa de forma manual. Las subclases principales sobre las que se suelen programar las pruebas unitarias son ActivityInstrumentationTestCase2, diseñada para realizar pruebas funcionales, y ActivityUnitTestCase, diseñada para realizar test de activities Android de forma aislada.

Android Lint es una herramienta de análisis de código estático que comprueba los archivos de código fuente un proyecto Android ante posibles errores y mejoras de optimización para la corrección, la seguridad, el rendimiento, la usabilidad, la accesibilidad y la internacionalización del mismo.

Herramientas de Prueba de Caja Negra: En esta ocasión utilizaremos una herramienta llamada Selenium WebDriver para ejecutar las pruebas. Este herramienta genera un entorno de automatización de pruebas que operan en las principales navegadores, pero en nuestro caso también es útil para dispositivos móviles por ejemplo iPhone y Android, que utiliza lenguajes como Python, Ruby, Java y C#, además trabaja bajo licencia de Apache 2.0, por lo que es Open Source.