MINTIC

PLAN DE PRUEBAS
YO CUIDO LO PÚBLICO - MÓVIL
Dirección de Gobierno Digital

Bogotá, D.C, 09 de octubre de 2017

TABLA CONTENIDO

Lista de Tablas

DERECHOS DE AUTOR

INTRODUCCIÓN

El presente documento presenta el plan de pruebas con la finalidad de satisfacer los requisitos de calidad del proyecto asegurando la satisfacción de la Dirección de Gobierno Digital.

El propósito del Plan de Pruebas es proveer un artefacto central que gobierne la planeación y control del esfuerzo de pruebas. Este define el enfoque general que será empleado para probar la aplicación y para evaluar los resultados de esas pruebas y es el plan de más alto nivel que será usado por los administradores para guiar y dirigir el trabajo de pruebas en detalle.

Provee visibilidad a los interesados en el esfuerzo de pruebas que han tenido las consideraciones adecuadas para varios aspectos que orientan el esfuerzo de pruebas, y dónde es apropiado que los interesados aprueben el plan.

1. ALCANCE

Este documento detalla las actividades necesarias para ejecutar los tipos de pruebas que se realizarán en la aplicación Yo Cuido Lo Público. El alcance de las pruebas está enfocado en probar todas las historias de usuario y requerimientos no funcionales desarrollados para la solución.

2. PLAN DE PRUEBAS DETALLADO

2.1 METODOLOGÍA DE PRUEBAS

A continuación se presenta la metodología de pruebas por cada uno de los ambientes de prueba que se tienen definidos:

- **Ambiente local (ambiente de desarrollo)**
  - Pruebas unitarias para Android

    El desarrollo de las pruebas unitarias en Android se realizará de la siguiente manera:
    Los desarrolladores conectarán el dispositivo móvil al ambiente de desarrollo de la aplicación es decir sobre la arquitectura real para garantizar la veracidad y efectividad de los resultados de la prueba y no sobre simulador donde la arquitectura de pruebas no es la misma que la de producción.

  - Pruebas Unitarias para Web Services: estas pruebas se realizarán para garantizar que los Web Services funcionen correctamente.

- **Ambiente de pruebas preproducción**

Cuando la solución ha terminado el último sprint de construcción se inician la prueba en el ambiente de preproducción, en la cual se realizan la fase donde el proveedor verifica que la versión se encuentra correctamente instalada y notifica a Interventoría/Entidad/GD que se puede dar inicio a las pruebas que ellos realizan, teniendo en cuenta lo que se indica a continuación para cada una de las fases.

En esta fase se realizan pruebas por parte de la Interventoría/Entidad/GD una vez que haya verificado que la versión se encuentra correctamente instalada.

  - Verifica que la versión corresponde con la versión que debe probar la Interventoría/Entidad/GD.

- Notifica por medio de comunicado a Interventoría/Entidad/GD que la versión se encuentra instalada en Preproducción para pruebas por parte de Interventoría/Entidad/GD.

- **Ambiente de Producción:**

El equipo de pruebas realiza una verificación rápida a la versión que está lista para publicar en la tienda Google Play (Android) asegurando que ésta se encuentra actualizada antes de iniciar las pruebas que se realizarán en este ambiente.

La verificación en este ambiente por parte de Entidad/GD y Ciudadanos se iniciarán una vez se confirme que la versión esta lista para ser publicada en las tiendas.

2.2 TIPOS DE PRUEBAS

A continuación se describen las pruebas funcionales y no funcionales que aplican y no aplican para la solución. Las entidades participantes son responsables de entregar a los datos para realizar las pruebas y garantizar que estos consistentes.

2.2.1 PRUEBAS FUNCIONALES

Pretende validar los requisitos funcionales, incluyendo la navegación dentro de la aplicación, entrada de datos, procesamiento y obtención de resultados.

- **Pruebas unitarias:** Se verifica la funcionalidad de cada módulo.

- **Pruebas de funcionalidad:** Se verifica que la funcionalidad de la aplicación satisface los requerimientos funcionales solicitados.

- **Pruebas de interfaz de usuario:** Se verifica que los componentes de la interfaz gráfica de usuario sea consistente (elementos de navegación, presentación de iconos, nombre de los elementos, ortografía de los elementos, etc.)

- **Pruebas de ciclo de negocio:** Se verifica el sistema a lo largo de todo un ciclo completo de negocio.

- **Pruebas de bases de datos e integridad de datos:** No aplica porque las bases de datos u orígenes de datos[1] del aplicativo Yo Cuido Lo Público, son de

[1]El Origen de datos representa la fuente de información expuesta y disponible para ser consumida por los servicios web para cada una de las aplicaciones. Dicha fuente de información debe ser una representación coherente y alineada con los parámetros de entrada de los servicios

responsabilidad única y exclusiva de las entidades, por lo tanto, son también su desempeño, integridad y disponibilidad.

2.2.2 PRUEBAS FUNCIONALES A EJECUTAR

Tabla 1. Pruebas Funcionales a ejecutar

TIPO DE PRUEBA	¿SE REALIZA? SI /NO	AMBIENTE EN EL QUE SE REALIZA	OBSERVACIONES
Pruebas unitarias	SI	• Ambiente local (ambiente de desarrollo)	
Pruebas de funcionalidad	SI	• Ambiente prueba interno (pruebas UT) • Ambiente de pruebas preproducción(fase 1) • Ambiente de Producción	
Pruebas de interfaz de usuario	SI	• Ambiente pruebas interno (pruebas UT) • Ambiente de pruebas preproducción (fase 1) • Ambiente de Producción	
Pruebas de ciclo de negocio	SI	• Ambiente pruebas interno (pruebas UT) • Ambiente de pruebas preproducción (fase 1) • Ambiente de Producción	
Pruebas de bases de datos e integridad de datos	NO		Las bases de datos u orígenes de datos en el en el proyecto son de responsabilidad única y exclusiva de las entidades, por lo tanto lo son también su desempeño, integridad y disponibilidad.

definidos en el capítulo de integración del documento de Arquitectura de la solución. Se debe tener en cuenta que cada entidad suministrará dicha fuente de información por tanto también serán responsables de las operaciones o mecanismos necesarios para obtenerla y exponerla.

2.2.3 PRUEBAS NO FUNCIONALES

Pretenden medir el desempeño, rendimiento y disponibilidad ante fallos de los sistemas:

- **Pruebas de carga:** Se verifica que los servicios web funcionen adecuadamente cuando llega al límite esperado. Estas pruebas corresponden con el "Plan de pruebas de estrés y carga" de los pliegos.

- **Pruebas de stress:** Se establecen cuáles son los umbrales extremos de funcionamiento de los servicios web que consume cada aplicación una vez se supera el límite de carga esperado. Estas pruebas corresponden con el "Plan de pruebas de estrés y carga" de los pliegos.

- **Pruebas de desempeño:** Verificar los tiempos de respuesta esperados de la interfaz a las acciones del usuario y de los servicios web.

- **Pruebas de recuperación a fallas:** Se verifica que al forzar el fallo del software de diferentes maneras la recuperación se lleva a cabo apropiadamente.

- **Pruebas de configuración:** Se verifica que la solución se comporte de la manera esperada cuando se encuentra instalada en la versión de Android en el que serán ejecutado.

- **Pruebas de seguridad y control de acceso:** Se revisan cuáles son los riesgos de vulnerabilidad de la aplicación Yo Cuido Lo Público. Se realizan pruebas para verificar que los mecanismos de control de acceso al sistema funcionen como fueron definidos.

- **Pruebas de los servicios de interoperabilidad:** Se verifica que los diferentes servicios web involucrados en la aplicación Yo Cuido Lo Público se relacionan adecuadamente entre sí intercambiando información y utilizando la información intercambiada.

- **Pruebas de concurrencia:** Verificar la funcionalidad de los servicios web al recibir una cantidad estimada de peticiones para un mismo recurso en un mismo instante de tiempo.

- **Pruebas de Ethical Hacking:** Se verifica que al realizar ataques de vulnerabilidad controlados, la aplicación no sea vulnerada.

2.2.4 PRUEBAS NO FUNCIONALES A EJECUTAR

Tabla 2. Pruebas No Funcionales a ejecutar

TIPO DE PRUEBA	¿SE REALIZA? SI / NO	AMBIENTE EN EL QUE SE REALIZA	OBSERVACIONES
Pruebas de carga	SI	• Ambiente de pruebas preproducción en versión inicial • Ambiente de pruebas preproducción	
Pruebas de stress	SI	• Ambiente de pruebas preproducción en versión inicial • Ambiente de pruebas preproducción	
Pruebas de desempeño	SI	• Ambiente de pruebas interno UT	
Pruebas de recuperación a fallas	SI	• Ambiente de pruebas interno UT	
Pruebas de configuración	SI	• Ambiente de pruebas interno UT • Ambiente de pruebas preproducción • Ambiente de Producción	
Pruebas de seguridad y control de acceso	SI	• Ambiente de pruebas interno UT • Ambiente de pruebas preproducción • Ambiente de Producción	
Pruebas de los servicios de interoperabilidad	SI	• Ambiente de pruebas interno UT • Ambiente de pruebas preproducción	
Pruebas de concurrencia	SI	• Ambiente de pruebas interno UT • Ambiente de pruebas preproducción	

TIPO DE PRUEBA	¿SE REALIZA? SI / NO	AMBIENTE EN EL QUE SE REALIZA	OBSERVACIONES
		• Ambiente de Producción	
Pruebas de Ethical Hacking	SI	• Ambiente de pruebas interno UT • Ambiente de pruebas preproducción • Ambiente de Producción	

2.2.5 METODOLOGÍA UTILIZADA PARA LA ELABORACIÓN DE LAS PRUEBAS DE CARGA, STRESS, DESEMPEÑO Y CONCURRENCIA

Debido a la arquitectura planteada para la solución, las pruebas de carga, estrés, desempeño y concurrencia serán realizadas únicamente sobre los servicios, ya que la interfaz con el usuario se realiza sobre un dispositivo móvil y éste sólo tiene abierta una instancia de la aplicación a la vez. Adicionalmente de acuerdo a la arquitectura planteada los servicios estarán sobre arquitectura REST y los datos viajarán en formato JSON.

Para las pruebas de carga y estrés de los Web Services se usará la herramienta JMeter la cual permite realizar pruebas de rendimiento de recursos estáticos y dinámicos. Esta herramienta está ideada para simular una fuerte carga usando múltiples hilos para realizar peticiones y permitiendo el uso de diferentes esquemas para mostrar los resultados (tablas, gráficas, etc.).

Para un servicio web se deben realizar los pasos que se indican a continuación:

- **PASO 1**. Identificar los criterios de aceptación de desempeño.

  Estos criterios están provistos por los requerimientos no funcionales que se evidencian en el plan de proyecto de la solución y se detallan en el documento de historias de usuario del proyecto. De acuerdo a esto se resaltan los siguientes criterios de aceptación:

  - El número de usuarios concurrentes que debe soportar los servicios y el tiempo de respuesta están identificados en el documento de plan de administración de la capacidad.

- **PASÓ 2**. Identificar los escenarios clave.

  Los escenarios clave son las operaciones cuya funcionalidad es relevante para el negocio y para los cuales se tienen objetivos de desempeño específicos.

Son considerados de tener alto riesgo, aquellos que son más comúnmente usados, o que tienen un impacto significativo en desempeño. Los pasos básicos para identificar los escenarios clave son:

- Identificar las operaciones del servicio web que sean relevantes para cada aplicación. Como los servicios están en REST se deben identificar los parámetros y la forma en que estos serán enviados al servicio.

- Identificar las operaciones que son las más comúnmente ejecutadas o más intensivas en el uso de recursos; estos serán los escenarios claves para usar en las pruebas de carga.

- **PASÓ 3**. Identificar los niveles de carga objetivo.

Identificar los niveles de carga que se aplican a la distribución de carga de trabajo identificado(s) durante la etapa anterior. El propósito de la identificación de los niveles deseados de carga es asegurar que las pruebas se pueden utilizar para predecir o comparar una variedad de condiciones de carga de producción. Los siguientes son prerrequisitos comunes que se utilizan para determinar los niveles de carga de destino:

- Volumen del negocio (actual proyectado)

- Escenarios clave

- Distribución del trabajo

- Características de sesión (autenticación)

Combinando los puntos anteriores, se pueden determinar los detalles faltantes necesarios para implementar el modelo de carga de trabajo bajo un objetivo de carga particular.

Por ejemplo, una aplicación de consulta de estadísticas puede estar proyectada para mantener un máximo de treinta (30) usuarios concurrentes. Por lo tanto se puede modelar una prueba de carga en la que se inicia con cinco (5) usuarios y se va incrementando paulatinamente la carga hasta que se cumple con el tiempo de la prueba o el número de pruebas de rendimiento que se desean ejecutar.

- **PASO 4.** Identificar las métricas.

El objetivo es establecer cuáles son los indicadores clave que van a ser usados para obtener la información más efectiva que permita evaluar el rendimiento y desempeño de cada una de las aplicaciones que integran la solución, con el fin de determinar si existen cuellos de botella en la aplicación y si es necesario

realizar ajustes sobre la aplicación. Por ejemplo las métricas más significativas para un sistema pueden ser:

- Tiempo promedio por petición
- Peticiones por segundo
- Número máximo de peticiones concurrentes
- Errores por segundo
- Infracciones del umbral por segundo
- Porcentaje de tiempo de procesador
- Uso de Memoria
- Errores HTTP
- Número de pruebas realizadas.
- Tiempo total de prueba.
- Tiempo duración por prueba.

- **PASÓ 5**. Diseñar las pruebas específicas.

  Una vez realizados los pasos anteriores es posible diseñar pruebas específicas a ser ejecutadas. Cada prueba generalmente tiene un propósito diferente, recolectar diferentes datos, incluir diferentes escenarios, y tener diferentes niveles de carga objetivo. La clave es diseñar pruebas que ayuden al equipo a recopilar la información que necesita con el fin de entender, evaluar y poner a punto la aplicación.

- **PASÓ 6**. Ejecutar las pruebas.

  Ejecutar las pruebas con la herramienta definida para tal fin y recolectar los datos obtenidos para cada prueba diseñada.

- **PASÓ 7**. Analizar los resultados.

  Analizar los datos recolectados y comparar los resultados contra los niveles identificados para determinar el desempeño de la aplicación.

Todos los pasos indicados en este numeral deben quedar reflejados en el informe de pruebas de carga y stress que se presente.

2.2.6 METODOLOGÍA UTILIZADA PARA PRUEBAS DE SEGURIDAD (VULNERABILIDAD) Y CONTROL DE ACCESO

Por la naturaleza misma de las aplicaciones, las pruebas realizadas en el marco de este proyecto buscan exponer la efectividad de los controles implementados en las aplicaciones para mitigar los riesgos identificados en las mismas, mostrando las vulnerabilidades que existen y que pueden permitir que los riesgos generen un impacto.

- **PASO1**. Montaje de las aplicaciones

  En este primer paso se procede a configurar el ambiente de pruebas de la aplicación Yo Cuido Lo Público, el cual permite acceder a las versiones para pruebas de cada una de las aplicaciones, verificando la funcionalidad y utilizando diferentes herramientas de seguridad para realizar un análisis manual de las funcionalidades, comunicación y desempeño de la misma.

- **PASÓ 2**. Análisis Estático

  Las pruebas del análisis estático se realizan sobre el código de la aplicación. La revisión se realiza sin generar una ejecución del código.

  Las técnicas usadas en el análisis estático buscan determinar diferentes problemas de programación dentro del código como son; fugas de memoria, variables no inicializadas, código muerto, errores de uso de variables, buffer overflow, entre otras. El análisis de código se realiza en todas las líneas de código buscando posibles errores lógicos.

- **PASÓ 3**. Análisis Dinámico

  El análisis dinámico se refiere a realizar un acceso a las aplicaciones durante su ejecución, en búsqueda de posibles vulnerabilidades que se puedan identificar a nivel de transporte, almacenamiento o procesamiento de la información contenida en ellas y que pueda afectar la confidencialidad, integridad o disponibilidad de la información.

  Las pruebas dinámicas se enfocan en la realización de pruebas a los siguientes niveles:

  - Monitoreo de actividad de archivos (si aplica)
  - Monitoreo de Memoria
  - Monitoreo de Procesos
  - Monitoreo de Red

- **PASÓ 4**. Análisis del Servicio

Adicional a la revisión propia de la aplicación, se realizara adicionalmente la revisión al servicio en el marco de las pruebas realizadas frente al código. La revisión de servicio incluirá revisión de las respuestas del mismo frente a diferentes peticiones normales y especialmente estructuradas, desde la plataforma de pruebas, con el fin de definir el comportamiento del servicio, frente a las peticiones generadas en el mismo.

Las pruebas realizadas frente al análisis del servicio, se encuentran dentro del marco del análisis dinámico, pero enmarcadas como un componente adicional, enfocado a revisar los controles de seguridad establecidos para estos, dentro del marco del proceso determinado.

Las revisiones realizadas en este apartado, tendrán en cuenta la comprobación de métodos de autenticación, controles de cifrado a nivel de transporte, esquemas de autorización, manejo de sesiones y revisión de funcionalidad no deseada del servicio.

Cada una de las pruebas se encuentra basada en la metodología propuesta por el proyecto Open Web Application Security Project (OWASP)[2].

- **PASÓ 5**. Realizar análisis de impacto

  Realizar un análisis del posible impacto que se puede generar por la explotación de las aplicaciones en la plataforma y se realizan las recomendaciones específicas para la mitigación o eliminación de las vulnerabilidades identificadas.

- **PASÓ 6**. Realizar pruebas de revisión

  Dentro de estas pruebas, se realiza una revisión de la mitigación de vulnerabilidades encontradas en el código, la aplicación, el almacenamiento, los servicios y la capa de transporte ya detectadas dentro de la identificación de vulnerabilidades. Se hace una revisión completa de estas y se revisan posibles vulnerabilidades asociadas a la remediación de las que se encontraron inicialmente.

- **PASÓ 7**. Conclusiones

  Se documentan las recomendaciones y conclusiones según los hallazgos y mitigaciones realizadas.

2.2.7 METODOLOGÍA UTILIZADA PARA PRUEBAS DE ETHICAL HACKING

[2] OWASP Mobile Security Project - Disponible en: https://www.owasp.org/index.php/OWASP_Mobile_Security_Project

Dentro de las pruebas de Ethical Hacking, adicional a la identificación de vulnerabilidades realizada en las pruebas de seguridad descritas en la sección anterior, se realizan pruebas de explotación de las vulnerabilidades identificadas, las cuales permiten medir, basado en el impacto generado, la problemática real de seguridad a la que se encuentran enfrentados las aplicaciones y los servicios.

La ejecución de las pruebas de Ethical Hacking, permiten generar una medición real de las consecuencias de las vulnerabilidades identificadas en el paso anterior, estableciendo recomendaciones específicas para la generación de controles con el fin de mitigar los aspectos identificados en estas pruebas. Igualmente la explotación de vulnerabilidades, puede permitir la identificación de errores o problemas adicionales de seguridad en las aplicaciones y servicios analizados.

2.3 ENTREGABLES DE LA EJECUCIÓN DE PRUEBAS

A continuación se presentan los entregables de acuerdo con las pruebas que se realicen en cada ambiente:

- **Ambiente local (ambiente de desarrollo)**

  Las pruebas unitarias se realizaran con el objetivo de garantizar la calidad de la solución sin embargo no se realizara el registro en el repositorio.

- **Ambiente de pruebas preproducción**

  Fase 1

  El resultado obtenido en la actividad de “Revisión Final de Calidad” será incluido en el informe de pruebas que se presenta como entregable en el ambiente de pruebas funcionales (pruebas UT).

  Fase 2

  Los entregables de las pruebas realizadas en esta fase son los registros de gestión de incidencias y el informe de pruebas de la Solución. Todos estos según las plantillas definidas para tal fin y lo reportado en JIRA por la Interventoría/Entidad/GD.

- **Ambiente de Producción**

  Los entregables de las pruebas realizadas en este ambiente son:

  - Informe de pruebas de vulnerabilidad.

- Informe del Ethical Hacking.
- Informe de pruebas de estrés y carga.

2.4 AMBIENTE DE PRUEBAS

2.4.1 HARDWARE

El hardware que se indica a continuación corresponde con lo requerido para los ambientes:

Tabla 3. Hardware requisito

RECURSO	CANTIDAD	NOMBRE / TIPO DE RECURSO
Computador analista senior/junior Ambiente local	3	• Configuración: ▪ Procesador Intel Dual Core 2.66 GHz. ▪ 3 Gb RAM. ▪ 120 Gb D.D. ▪ Red 10/100. ▪ Mac requerido para desarrollo IOS Dispositivos Android
Computador analista junior pruebas Ambiente pruebas preproducción	1	• Configuración: ▪ Procesador Intel Dual Core 2.66 GHz. ▪ 3 Gb RAM. ▪ 120 Gb D.D. ▪ Red 10/100. • Dispositivos móviles, con sistema operativo android Se debe contar con conexión de plan de datos y wifi para realizar las pruebas y poder validar el rendimiento de las aplicaciones bajo condiciones de navegación limitadas.

2.4.2 SOFTWARE

El software que se indica a continuación corresponde con lo requerido para los ambientes:

Tabla 4. Software requisito

RECURSO	CANTIDAD	NOMBRE / TIPO DE RECURSO
Computador analista senior/junior Ambiente local	3	• Sistema Operativo: Windows 7, para desarrollo Android;

RECURSO	CANTIDAD	NOMBRE / TIPO DE RECURSO
Computador analista junior de Pruebas Ambiente preproducción	1	• Sistema Operativo: Windows 7

2.5 DESPLIEGUE DE VERSIÓN PARA PRUEBAS

Apoyándose en la herramienta Subversión, los desarrolladores tendrán la responsabilidad del manejo de las fuentes a su cargo. La persona del equipo por demanda encargada de realizar el despliegue de las versiones se encargará de verificar que antes de pasar una versión para pruebas en cualquiera de los ambientes indicados en este documento, el código fuente, se encuentra actualizado en el respectivo repositorio y deberá crear la respectiva línea de base con la herramienta Subversión, teniendo en cuenta el estándar de nombramiento indicado en el plan de configuraciones que se encuentra en el repositorio la ruta MODULO DOCUMENTOS / Administración / 2. Planeación / Gestión de la Calidad / Plan de Configuraciones.

2.6 PROCESO DE REPORTE Y MANEJO DE INCIDENCIAS

El proceso para el manejo de incidencias se realizará de acuerdo con lo indicado en el documento GLFS2-GB-OT-MetodologiaGestionDefectos, el cual se encuentra en la ruta MODULO DOCUMENTOS / Administración / 2. Planeación / Proceso Gestión de Defectos

3. TERMINOLOGÍA

Android[3]: Es un sistema operativo basado en Linux, diseñado principalmente para móviles con pantalla táctil como teléfonos inteligentes o tabletas inicialmente desarrollados por Android, Inc.

Casos de prueba: Son un conjunto de condiciones o variables, bajo las cuáles el analista determinará si el requisito de una aplicación es parcial o completamente satisfactorio.

Criterio de aceptación: Norma que establece la aprobación de un entregable.

Dual Core: Es un procesador que ofrece un desempeño de alto valor para ejecutar multitareas.

El Proveedor: Se refiere a la casa desarrolladora de software que realiza la implementación de la solución.

Equipo por demanda: Es el equipo de apoyo para los proyectos.

Fábrica de software: Línea de producción de software.

Hibrido[4]: Es cuando una aplicación nativa con una capa intermedia de herramientas nos permiten, por citar un ejemplo, usar una aplicación jQuery Mobile en su interior.

Historia de Usuario: Es una representación de un requisito de software escrito en una o dos frases utilizando el lenguaje común del usuario.

[3]Disponible en: http://es.wikipedia.org/wiki/Android.
[4]Disponible en: http://www.franciscojavierpulido.com/2011/12/phonegap-introduccion-los-sistemas.html#.UWRAbFdi2a4.