



Diplomado Quality Assurance y Software Testing

Modulo II: Web Services Testing - Mobile

Nombre: Cesar Mendoza Quisbert

2023

Contenido

-	Introduccion	3
-	Limites.....	3
-	Desarrollo y descripción de las tecnologías utilizadas	3
-	Desarrollo y descripción de los ciclos de testeo	4
-	Conclusiones y recomendaciones	5

1. Introducción

El modulo de Web Services Testing - Mobile muestra la formas de test que hay actualmente en el área de Mobile y como aplicarlas y los web services testing da paso a las formas de test en los servicios web.

2. Limites

El presente proyecto solo mostrara las partes principales del proceso de pruebas manules y automáticas en el área de mobile y servicios web.

3. Desarrollo y descripción de las tecnologías utilizadas

A continuación, se describe las herramientas que se utilizaron para el desarrollo del proceso de control de calidad, mobile y servicios web.

Testlink: Es una herramienta de software libre de gestión de pruebas de software que permite a los equipos de desarrollo de software planificar, diseñar, ejecutar y reportar pruebas de software de forma centralizada y colaborativa.

Word: Microsoft Word es un programa de procesamiento de textos desarrollado por Microsoft Corporation. Se utiliza para crear, editar y formatear documentos de texto, incluyendo cartas, ensayos, informes y otros tipos de documentos. Además, incluye herramientas de revisión y colaboración en tiempo real, así como una amplia variedad de opciones de formato y diseño.

RobotFramework: Robot Framework es un marco de pruebas de software de código abierto que se utiliza para automatizar pruebas de software y proporcionar una estructura para organizar y presentar los resultados.

Android Estudio dispositivo virtual: El dispositivo virtual de Android Studio es una herramienta de simulación que permite a los desarrolladores probar y depurar aplicaciones de Android en un entorno controlado y aislado en el equipo de desarrollo.

Appium: Es un marco de automatización de pruebas de aplicaciones móviles de código abierto que permite a los equipos de pruebas automatizar pruebas en dispositivos iOS y Android. Se integra con una amplia variedad de lenguajes de programación y herramientas de prueba, y utiliza un enfoque de "pruebas nativas" para probar aplicaciones móviles, lo que significa que la automatización se realiza en el nivel de la interfaz de usuario de la aplicación.

Appium inspector: Es una herramienta de inspección de la interfaz de usuario de la aplicación que se integra con el marco de automatización de pruebas de aplicaciones móviles Appium. Permite a los equipos de pruebas examinar y examinar la estructura de la interfaz de usuario de la aplicación y generar automáticamente scripts de prueba en tiempo real.

AppiumLibrary: es una biblioteca de Python para el marco de automatización de pruebas de aplicaciones móviles Appium. Proporciona una serie de comandos y funciones de alto nivel para automatizar pruebas de aplicaciones móviles en dispositivos iOS y Android. Los comandos incluyen acciones como tocar, deslizar, ingresar texto y verificar la presencia de elementos en la pantalla. Además, AppiumLibrary se integra con otros marcos de pruebas como Robot Framework, lo que permite a los equipos de pruebas utilizar sus scripts de prueba existentes y aprovechar la potencia de la automatización de pruebas en combinación con Appium.

Python: Python es un lenguaje de programación de alto nivel, interpretado y orientado a objetos, con un enfoque en la legibilidad y la facilidad de uso. Fue creado en el año 1989 por Guido van Rossum y es **conocido** por su sintaxis clara y concisa, lo que lo hace accesible a programadores principiantes y avanzados.

JAVA: Es un lenguaje de programación de alto nivel, orientado a objetos, que fue desarrollado por Sun Microsystems (ahora propiedad de Oracle) en el año 1995. Se caracteriza por su independencia de la plataforma, lo que significa que los programas escritos en Java pueden ejecutarse en cualquier sistema operativo que tenga una máquina virtual de Java instalada.

RequestsLibrary: Es una biblioteca de Python para el marco de automatización de pruebas Robot Framework. Proporciona una serie de comandos para realizar solicitudes HTTP y verificar las respuestas en aplicaciones web y servicios web. Los comandos incluyen funciones para realizar solicitudes GET, POST, PUT, DELETE y otros métodos HTTP, así como para verificar el estado de la respuesta HTTP, las cabeceras y el cuerpo de la respuesta.

Collections: La librería Collections de Python es un módulo que proporciona diferentes clases y funciones que facilitan el manejo de colecciones de datos en Python. Las colecciones incluyen listas, tuplas, diccionarios, conjuntos y otros tipos de estructuras de datos. La biblioteca Collections incluye clases como deque, Counter, OrderedDict, defaultdict y namedtuple, que brindan una variedad de funcionalidades para trabajar con colecciones de datos en Python.

JSONLibrary: Es una biblioteca de Python para el marco de automatización de pruebas Robot Framework. Proporciona una serie de comandos para trabajar con archivos y datos JSON (JavaScript Object Notation). JSON es un formato de intercambio de datos ligero y humanamente legible que se utiliza comúnmente en aplicaciones web y servicios web. JSONLibrary permite a los equipos de pruebas automatizar y validar la funcionalidad de las aplicaciones y servicios web que utilizan JSON como formato de intercambio de datos.

4. Desarrollo y descripción de los ciclos de testeo

4.1. Estrategia a desarrollar

Pruebas unitarias: pruebas que se realizan en componentes individuales de la aplicación para asegurarse de que funcionen de manera aislada.

Pruebas integración: pruebas que se realizan para verificar cómo los componentes de la aplicación funcionan juntos.

Pruebas de usuario: pruebas que se realizan con usuarios reales para obtener retroalimentación y verificar la usabilidad de la aplicación.

A continuación, se describe el Test plan en el área de Mobile:

4.2. Test plan Mobile:

4.2.1. Introducción

La información descrita en el siguiente documento permite al área de QA describir toda la información para ejecutar un plan de pruebas en el flujo de material móvil.

4.2.2. Alcance

Definir todos los requisitos funcionales o no funcionales de las características que se probarán y agregarán en el plan de pruebas a la versión

4.2.3. En el ámbito de aplicación

- Setup area
- Login/password
- Asset area
- Part Routes
- Route Selection
- Material pick
- Material delivery
- Logout

4.2.4. Entornos

La aplicación solo se probará en emulación de Android:

Versión de Android: 8.1

Móvil: Pixel 3- API: 27

4.2.5. Fuera del alcance

El plan de prueba no cubre las áreas/procesos:

- Área de excepción
- Escanear para seleccionar área
- Proceso de escaneo en todas las áreas
- Proceso de recogida/entrega con el proceso de excepción
- Proceso de alerta de mensajes

4.2.6. Entornos

- La aplicación no se probará en otros tipos de móviles virtuales

4.2.7. Objetivos de la prueba

El objetivo de la prueba es verificar que la funcionalidad del flujo de material móvil según el alcance se define con los siguientes criterios cuando la aplicación se somete al proceso de prueba de control de calidad:

- Los errores como showstoppers se informarán lo antes posible al área móvil de DEV.
- Si la aplicación tiene un showstopper, el error debe notificarse y volver a ejecutar el plan de prueba
- Si un error se identifica como de prioridad alta o media, la funcionalidad debe volver a probarse
- Si un error tiene un defecto de gravedad alta o media debe resolverse y verificar la corrección.
- Los errores como de baja prioridad o gravedad, el error debe ser revisado para el PM / DEV / QA
- Para ejecutar el plan de prueba, la aplicación debe tener una versión candidata

4.2.8. Funciones y responsabilidades

Descripción detallada de los roles y responsabilidades de los diferentes miembros del equipo en el proyecto:

- Gerente de Control de Calidad

- Prueba manual de control de calidad
- Prueba automatizada de control de calidad
- Administrador móvil DEV
- Diseño UI móvil
- DEV móvil

4.2.9. Metodología de prueba

4.2.10. Visión general

Scrum es un marco de gestión para el desarrollo incremental de productos utilizando uno o más multifuncionales. Proporciona una estructura de roles, reuniones, reglas y artefactos. Los equipos son responsables de crear y adaptar sus procesos dentro de este marco.

4.2.11. Niveles de prueba

Prueba funcional, para verificar las diferentes funciones trabajadas según el alcance.

Report GUI Test, para verificar la GUI de la aplicación de acuerdo con el diseño de maqueta aprobado.

4.2.12. Clasificación de errores

Los errores podrían tener el siguiente estado:

BAJO: El tipo de error bajo podría ser errores de GUI, ortografía, tipografía, navegadores de compatibilidad, iconos, algunos errores de escritura o incluso errores cosméticos. Todo lo que no se muestra correctamente según la información de las maquetas.

MEDIO: Los bugs como medio podrían ser funcionales de proceso, filtros de información, enlaces rotos, botones rotos, cualquier error en consola.

ALTO: Los errores tan altos podrían ser una característica no es utilizable como se supone que debe ser.

CRÍTICO: Esto generalmente ocurre en los casos en que se bloquea una funcionalidad completa y se detiene el proceso de prueba. O el rendimiento de la aplicación se ve afectado por fugas de memoria, el error debe solucionarse lo antes posible.

4.2.13. Criterios de suspensión y requisitos de reanudación

Cuando se encuentra un error que bloquea parte de la aplicación y las otras funciones dependen de su información, la prueba debe detenerse e informar del error.

Después de resolver el error, el plan de prueba debe comenzar desde cero, ya que la solución podría romper otras funciones ya probadas.

4.2.14. Integridad de la prueba

- 100% de cobertura de prueba según el alcance
- Todos los casos de prueba manuales y automatizados ejecutados de acuerdo con el alcance
- Todos los errores abiertos se corregirán en la próxima versión de acuerdo con la prioridad del propietario o gerente del proyecto.
- Si se lanza la aplicación y los errores continúan como abiertos, deben informarse como problemas conocidos

4.2.15. Prueba de automatización

- Debería implementarse con el nombre de marco de automatización de código abierto Robot Framework.
- Robot Framework utilizará bibliotecas Selenium y lenguaje Python.
- Appium y AppiumLibrary para Robotframework
- Administrador de dispositivos virtuales Android

4.2.16. Entregables de prueba

El Área de QA entregará la siguiente información una vez finalizado el proceso de prueba:

- Plan de pruebas
- Casos de prueba
- Prueba manual/automatizado
- Informes de errores

4.2.17. Necesidades de recursos y medio ambiente

4.2.17.1. Herramientas de prueba

Prueba de automatización:

- Selenium2Library
- Marco del robot
- AppiumLibrary
- Appium
- Appium Server GUI
- Appium Inspector

Informe de casos de prueba:

- Informe de enlace de prueba
- Informe de Robot Framework

4.2.17.2. Entorno de prueba

- Android 8.1
- Nivel de API 27
- Resolución 1080 x 2220
- Núcleo de CPU 4

4.2.18. Riesgos y contingencias

La siguiente tabla describe los riesgos y contingencias que podrían aparecer en el proceso de desarrollo/prueba

Riesgos	Contingencias
El sprint terminará con tareas/tickets abiertos	Las áreas de DEV/QA deben anotar al propietario las tareas/tickets que pasan al siguiente sprint o si se debe completar una tarea/ticket importante en el sprint.
Vacaciones en Bolivia	Si el mes tiene feriados bolivianos, se informará al propietario.
Si una persona del área de control de calidad está enferma o renuncia	Si un personal de control de calidad está fuera, algunas pruebas manuales de control de calidad personal pueden ayudar a continuar el proceso de control de calidad en el sprint.
Si el código congelado no se implementa a tiempo para ser probado para el área de control de calidad	Si el control de calidad revisa el código congelado fuera de tiempo, el proceso de prueba debe reprogramarse y anotarlo al propietario.
Si el usuario o propietario necesita información sobre una funcionalidad, instalación o preguntas sobre la aplicación en producción	El área de DEV/QA estará disponible para responder a las preguntas a través de una reunión de acuerdo con el propietario

4.3. Test Cases Mobile: Test Plan: FLOW Android mobile test plan happy path (Ingles) Test Suite: Android Device

Test Case ZMF-6362: Setup and verify the flow connection server		
<u>Author:</u>	cmendoza	
<u>Summary:</u>	To setup and verify the connection server	
<u>Preconditions:</u>	<ul style="list-style-type: none"> - To have a mobile MC33 or emulator with Android 8.1 - To have the application material flow installed in the devices or emulator - To have the FLOW server up 	
<u>#:</u>	<u>Step actions:</u>	<u>Expected Results:</u>
1	Open the mobile application material Flow	-The application should open and display the login and password options
2	Login to the application as setup user	-The application should display the configuration area with the server address, inactivity time (minutes) fields and show debug logs option
3	Changes the server address to the QA server enable and click the [i] icon to verify the connection	-If the server path is correct the application display the message:

		"Server validate successful" and the [i] icon changes to check icon -If the server path is not correct the application displays the message: "invalid server"
4	Check/uncheck the show debug logs	- If the checkbox is checked the " show debug logs" the application can save the logs of the application in the device - If the checkbox is not checked the " show debug logs" the application cannot save the logs of the application in the device
5	Click the SAVE button	-The application should save the changes and go back to the login area
<u>Execution type:</u>		Manual.automated
<u>Priority:</u>		Medium

Test Case ZMF-6363: Verify the search option in the asset area

<u>Author:</u>		cmendoza
<u>Preconditions:</u> - To have a mobile MC33 or emulator with Android 8.1 - To have the application material flow installed in the devices or emulator - To have the FLOW server up		
<u>#:</u>	<u>Step actions:</u>	<u>Expected Results:</u>
1	Open the mobile application material Flow	-The application should open and display the login and password options
2	Login to the application as admin user	-The application should display the different areas added in the mobile like: asset, part routes, route selection, scan to pick, material pick, material delivery, exceptions, logout
3	Click in the asset option	The application should display a message for scan process: "Please proceed to scan a Asset Code or close this dialog to pick one from the list.
4	Click the "OK" button in the message opened for scan process	
5	Click in the lens icon	The application should open the search area, with the fields like Tag ID#, Asset ID, Keyword and the button SEARCH
6	Filter asset by the option by Tag ID#, Asset ID, Keyword	- All the fields should be work individual or combine the fields.

		- If the search match a element the application show the "asset ID" - If the search goesn't mach any element the application display the message "Nothing found to display"
<u>Execution type:</u>	Manual/automated	
<u>Priority:</u>	Medium	

Test Case ZMF-6364: Verify the search option in part routes area

<u>Author:</u>	cmendoza	
<u>Summary:</u>		
To verify the the search functionality in the part routes		
<u>Preconditions:</u>		
<ul style="list-style-type: none">- To have a mobile MC33 or emaluator with Android 8.1- To have the application material flow installed in the devices or emulator- To have the FLOW server up		
<u>#:</u>	<u>Step actions:</u>	<u>Expected Results:</u>
1	Open the mobile application material Flow	-The application should open and display the login and password options
2	Login to the application as admin user	-The application should display the differents areas added in the mobile like: asset, part routes, route selection, scan to pick, material pick, material delivery, exceptions, logout
3	Click in the Part Route option	The application should display a message "Please proceed to scan a new part route or close this dialog to pick one from the list"
4	Click the "OK" option	The application should display a list of part routes created
5	Click in the lens icon	The aplicacion should display the fields like: <ul style="list-style-type: none">- Part# : prefix, base, suffix- tag ID #- PR#-Keyword and the button SEARCH
6	Filter a part route by prefix, base, suffix. <ul style="list-style-type: none">- Tag ID #- PR# (part route	The filters could be search a part route individually or combined the fields.

	number) - Key word	- If the search fould parts route according the filters, the application should display all the part route that have similimar element search in the fields. - if the search doesn't match any element the application displays the message "Nothing found to display"
<u>Execution type:</u>	Manual	
<u>Priority:</u>	Medium	

Test Case ZMF-6365: Verify the request option in the part routes area		
<u>Author:</u>	cmendoza	
<u>Summary:</u>	to verify if the user can do a request part from the Flow mobile	
<u>Preconditions:</u>	<ul style="list-style-type: none"> - To have a mobile MC33 or emaluator with Android 8.1 - To have the application material flow installed in the devices or emulator - To have the FLOW server up 	
<u>#:</u>	<u>Step actions:</u>	<u>Expected Results:</u>
1	Open the mobile application material Flow	he application should open and display the login and password options
2	Login to the application as admin user	The application should display the differents areas added in the mobile like: asset, part routes, route selection, scan to pick, material pick, material delivery, exceptions, logout
3	Click the Par Routes option	the application goes to Part routes and display a message for the scan process
4	Click the OK button from the message	
5	Search a part form the list and click it	<p>The application should open the part, showing its information like.</p> <ul style="list-style-type: none"> - PR# -Part# - Std Pack Name - Std PAck Qty -Description - Tag id# - Trigger Location - Instructions

6	Click the box icon in the down site to make a request	<p>The application should display the message "Request, Do you want to perform the Material Request for this Part route" with the option Cancel/OK</p> <p>If the Cancel button is press the application go back to the Part route details</p> <p>If the OK button is press the application make a request of the part route</p> <p>If the part route is not configured correctly the application display the message "There was a problem requesting Part Route, please try again or check the server address"</p> <p>If the part route is configured correctly the part route should be requested and the application display the message "Material requested successful"</p>
<u>Execution type:</u>	Manual	
<u>Priority:</u>	Medium	

Test Case ZMF-6366: Verify the assign route to the user by Route Selection		
<u>Author:</u>	cmendoza	
<u>Summary:</u>	to very the assign route to the user to pick and delivery a part	
<u>Preconditions:</u>	<ul style="list-style-type: none"> - To have a mobile MC33 or emaluator with Android 8.1 - To have the application material flow installed in the devices or emulator - To have the FLOW server up 	
<u>#:</u>	<u>Step actions:</u>	<u>Expected Results:</u>
1	Open the mobile application material Flow	-The application should open and display the login and password options
2	Login to the application as admin user	-The application should display the differents areas added in the mobile like: asset, part routes, route selection, scan to pick, material pick, material delivery, exceptions, logout
3	Select the option route selection	The application should go to the route selection area to select a route type
4	Select a route type from the dropdown "my current assigned routes" and select a route type	the application should display the routes create on this route type.

5	Select/check a route from the list and click the save button icon	<p>The application should display the messages "The record was updated successfully" , "the request has succeeded"</p> <p>The route should be assigned to the user and it should be displayed in the list of routes.</p>
<u>Execution type:</u>	Manual	
<u>Priority:</u>	Medium	

Test Case ZMF-6367: Verify the Material pick process with part routes requested		
<u>Author:</u>	cmendoza	
<u>Summary:</u>	To verify the pick process material pick process	
<u>Preconditions:</u>	<ul style="list-style-type: none"> - To have a mobile MC33 or emulator with Android 8.1 - To have the application material flow installed in the devices or emulator - To have the FLOW server up 	
<u>#:</u>	<u>Step actions:</u>	<u>Expected Results:</u>
1	Open the mobile application material Flow	-The application should open and display the login and password options
2	Login to the application as admin user	-The application should display the different areas added in the mobile like: asset, part routes, route selection, scan to pick, material pick, material delivery, exceptions, logout
3	Click the material pick option	The application should display all the parts requested according to the routes assigned in the part routes area
4	From the list select a part and click the "next request" option or the "delivery without scan"	<ul style="list-style-type: none"> - When the user clicks the "next request" option the part goes to the "Pick Request Part" area, to verify the data and scan the requirements data as: Part, Lot#, Vendor# - when the user clicks the "delivery without scan" option the application goes to the delivery area without scanning any data.
5	From the "Pick request part" area the user can: <ul style="list-style-type: none"> - CANCEL the pick part - Report the part as EXCEPTION or PARTIAL EXCEPTION - Continue with the PICK 	<p>If the user:</p> <ul style="list-style-type: none"> - CANCEL the pick part, the part goes to the pick material list - put the part as EXCEPTION the part goes to EXCEPTION area and the pick process is finished - put the part as PARTIAL EXCEPTION the part goes to exception area and the other parts

	process: pick the next part if it is not a bulk package - Scan the requemients data.	contine the pick process - Click the NEXT REQUEST button, the user can pick the next part if it is not a bulk package
6	Continue with the pick process, clicking the NEXT REQUEST button	when the route doesn't have any more part on the list the application display the message: "Attetion, No more Material Requests for Route(s). Please begin Delivery process by clicking 'begin delivery'"
7	Click the "Begin delivery" option	The parts should go to material delivery area
<u>Execution type:</u>	Manual	
<u>Priority:</u>	Medium	

Test Case ZMF-6368: Verify the delivery process of a part in material delivery area

<u>Author:</u>	cmendoza	
<u>Summary:</u>	To verify the delivery process of a part in delivery area	
<u>Preconditions:</u>	<ul style="list-style-type: none"> - To have a mobile MC33 or emaluator with Android 8.1 - To have the application material flow installed in the devices or emulator - To have the FLOW server up - To have parts in the delivery area 	
<u>#:</u>	<u>Step actions:</u>	<u>Expected Results:</u>
1	Open the mobile application material Flow	-The application should open and display the login and password options
2	Login to the application as admin user	-The application should display the differents areas added in the mobile like: asset, part routes, route selection, scan to pick, material pick, material delivery, exceptions, logout
3	Click the material delivery option	<p>The application should open the material delivery area.</p> <ul style="list-style-type: none"> - if the area doesn't have any part in delivery the application should display the message: "Nothing found to display" - if the area have part in delivery, the application should display the part in a list.
4	Select a part from the list checking the checkbox	

5	Click the button begin delivery (play icon)	<p>The application goes to the "Delivery Request Part" area with the part information like:</p> <ul style="list-style-type: none"> - part#, Description, Quantity, To location, LS#, Gen. <p>Instructions nad the option to scan a part to be validate.</p> <p>- If the part is bulk package the application display a warning message: "Please note, this part has a bulk package type".</p>
6	Click the delivery without scan option (check yellow icon)	<ul style="list-style-type: none"> - The delivery without scan the user can delivery the part with out a scan. - The part complete the pick/delivery process. - The application goes to the material delivery area.
<u>Execution type:</u>	Manual	
<u>Priority:</u>	Medium	

Test Case ZMF-6369: Logout mobile flow application		
<u>Author:</u>	cmendoza	
<u>Summary:</u>	<p>To verify the log out of a user from the application</p>	
<u>Preconditions:</u>	<ul style="list-style-type: none"> - To have a mobile MC33 or emaluator with Android 8.1 - To have the application material flow installed in the devices or emulator - To have the FLOW server up 	
<u>#:</u>	<u>Step actions:</u>	<u>Expected Results:</u>
1	Open the mobile application material Flow	-The application should open and display the login and password options
2	Login to the application as admin user	-The application should display the differents areas added in the mobile like: asset, part routes, route selection, scan to pick, material pick, material delivery, exceptions, logout
3	Select the option logout	The application should display a message," Are you sure to log out ?" with two options Cancel/ok
4	Click the YES button	The application goes back to the material flow login/password
<u>Execution type:</u>	Manual	
<u>Priority:</u>	Medium	

4.4. Test Cases Video Games Rest API :Test Suite: Android Device (Ingles)
Test Suite : Rest API Video Games

Test Case VG-1: GET video games		
<u>Author:</u>	cmendoza	
<u>Summary:</u>		
Returns all the videos games in the DB		
<u>Preconditions:</u>		
To have Video Game Database - Test Application running		
<u>#:</u>	<u>Step actions:</u>	<u>Expected Results:</u>
1	Create a session to VideoGamesDB	
2	Create a GET session to app/videogames	<p>The application should display all games in JSON format like:</p> <p>VideoGame {</p> <p>id (integer, optional),</p> <p>name (string, optional),</p> <p>releaseDate (string, optional),</p> <p>reviewScore (integer, optional),</p> <p>category (string, optional),</p> <p>rating (string, optional)</p> <p>}</p> <p>The Response code should be 200</p>
<u>Execution type:</u>	Manual/automatico	
<u>Priority:</u>	Medium	

Test Case VG-2: POST video games		
<u>Author:</u>	cmendoza	
<u>Summary:</u>		
Add a new video game to the DB		
<u>Preconditions:</u>		
To have Video Game Database - Test Application running		
<u>#:</u>	<u>Step actions:</u>	<u>Expected Results:</u>
1	Create a session to VideoGamesDB	
2	<div>Create a POST session to app/videogames/ to add a new video game with those parameters: VideoGame { id (integer, optional), name (string, optional), releaseDate (string, optional), reviewScore (integer, optional), category (string, optional), rating (string, optional) }</div>	<div>The application should add a new video game to the DB</div> <div>The Response code should be 200</div> <div>The response body should be : {"status": "Record Added Successfully"}</div> <div>If the parametres are incorrect the application display a error 400</div> <div>If the item was already added the application display error 500</div> <div>If the game id is invalid display the error 404</div>
<u>Execution type:</u>	Manual	
<u>Priority:</u>	Medium	

Test Case VG-3: DELETE video games

Author: cmendoza

Summary:

Add a new video game to the DB

Preconditions:

To have Video Game Database - Test Application running

<u>#:</u>	<u>Step actions:</u>	<u>Expected Results:</u>
1	Create a session to VideoGamesDB	
2	<p>Create a DELETE session to app/videogames/{videoGameId} to delete a video game with those parameters:</p> <p>Parameter: VideoGameId Value example: 6 Data type: integer</p>	<p>The application should add a new video game to the DB</p> <p>The Response code should be 200</p> <p>The response body should be : {"status": "Record Added Successfully"}</p> <p>If the parametres are incorrect the application display a error 400</p> <p>If the item was already added the application display error 500</p> <p>If the game id is invalid display the error 404</p>
<u>Execution type:</u>	Manual	
<u>Priority:</u>	Medium	

Test Case VG-8: PUT video game by gameId

Author: cmendoza

Summary:

Returns a video game from the DB

Preconditions:

To have Video Game Database - Test Application running

<u>#:</u>	<u>Step actions:</u>	<u>Expected Results:</u>
1	Create a session to VideoGamesDB	
2	<p>Create a PUTsession to app/videogames/{videoGameId}</p> <p>Parameters: videoGameId Value: 1 Data Type: integer</p> <p>Send the body parameters</p> <p>VideoGame { id (integer, optional), name (string, optional), releaseDate (string, optional), reviewScore (integer, optional), category (string, optional), rating (string, optional) }</p>	<p>The request should return the {videoGameId}</p> <p>Response Body</p> <p>VideoGame { id (integer, optional), name (string, optional), releaseDate (string, optional), reviewScore (integer, optional), category (string, optional), rating (string, optional) }</p> <p>The Response code should be 200</p> <p>If the game id is not in the DB the application display a error 500</p> <p>If the game id is invalid display the error 404</p>
<u>Execution type:</u>	Manual	
<u>Priority:</u>	Medium	

Test Case VG-6: GET video game by gameId

Author: cmendoza

Summary:

Returns a video game from the DB

Preconditions:

To have Video Game Database - Test Application running

<u>#:</u>	<u>Step actions:</u>	<u>Expected Results:</u>
1	Create a session to VideoGamesDB	
2	<p>Create a GET session to app/videogames/{videoGameId}</p> <p>Parameters: videoGameId Value: 1 Data Type: integer</p>	<p>The request should return the {videoGameId}</p> <p>VideoGame { id (integer, optional), name (string, optional), releaseDate (string, optional), reviewScore (integer, optional), category (string, optional), rating (string, optional) }</p> <p>The Response code should be 200</p> <p>If the game id is not in the DB the application display a error 500</p> <p>If the game id is invalid display the error 404</p>
<u>Execution type:</u>	Manual	
<u>Priority:</u>	Medium	

4.5 Test plan reports

Test Plan : FLOW Android mobile test plan happy path

Test Project : Material Flow

Overall Build Status

Overall Build Status percentages are computed using only test cases that have tester assignment on build

Results by top level Test Suites

Test Suite	Total	Not Run	[%]	Passed	[%]	Failed	[%]	Blocked	[%]	Completed [%]
AndroidDevice	8	0	0.0	8	100.0	0	0.0	0	0.0	100.0

Test results according to test priorities

Priority	Total	Not Run	[%]	Passed	[%]	Failed	[%]	Blocked	[%]	Completed [%]
Medium	8	0	0.0	8	100.0	0	0.0	0	0.0	100.0

Test Plan : Test plan report Rest API Video Games

Test Project : Material Flow

Results by top level Test Suites

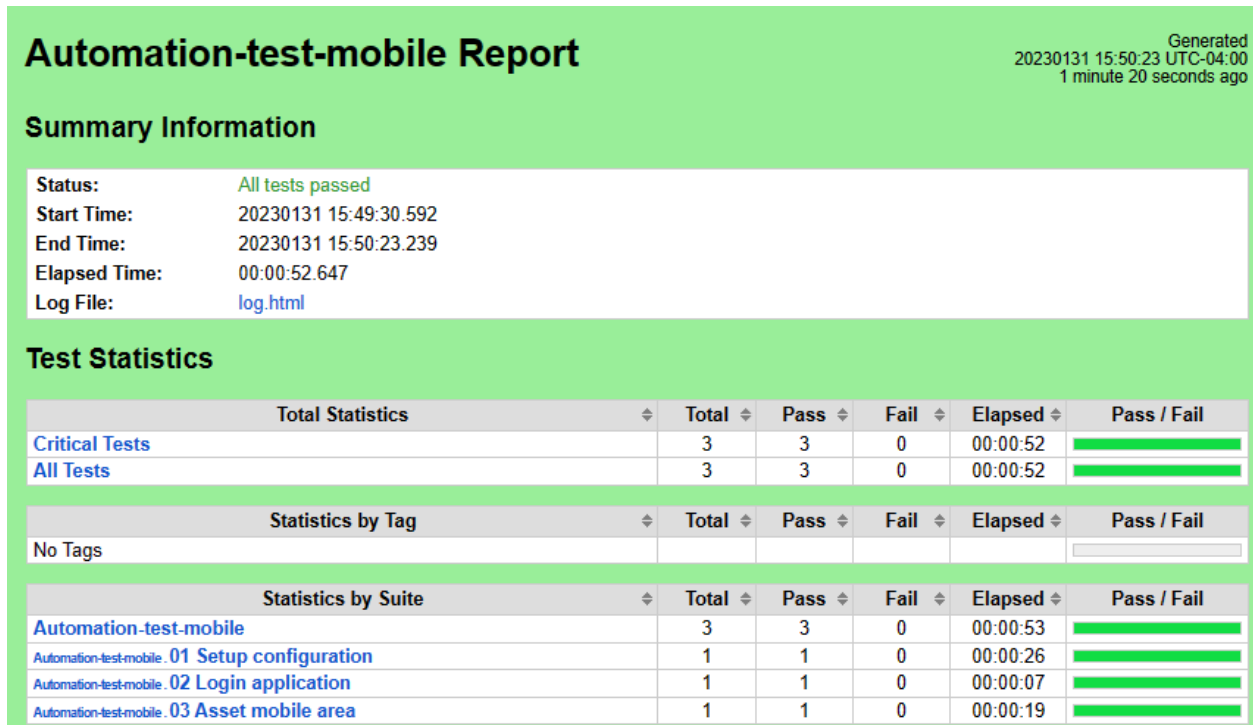
Test Suite	Total	Not Run	[%]	Passed	[%]	Failed	[%]	Blocked	[%]	Completed [%]
Rest API Video Games	5	0	0.0	5	100.0	0	0.0	0	0.0	100.0

This report shows results for each top level test suite. Results for subordinated test suites are count in for the corresponding top level test suite.

Test results according to test priorities


Priority	Total	Not Run	[%]	Passed	[%]	Failed	[%]	Blocked	[%]	Completed [%]
Medium	5	0	0.0	5	100.0	0	0.0	0	0.0	100.0

4.6 Automated test report



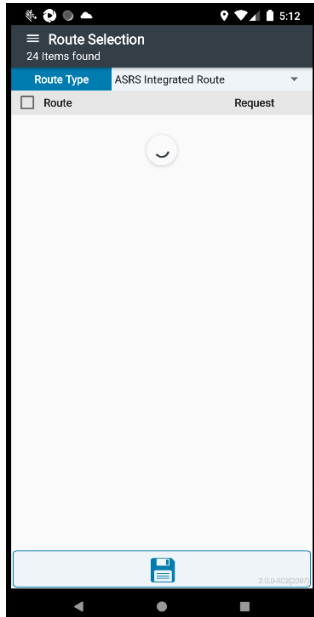
4.7 Bug Report Test plan

[AND]-BUG-ZMF-01: When the user put a wrong url and hit the save button the application save the URL and the application crashed

<u>Author:</u>	cmendoza	
<u>Preconditions:</u>		
<div>- To have a mobile MC33 or emulator with Android 8.1</div> <div>- To have the application material flow installed in the devices or emulator</div> <div>- To have the FLOW server up</div>		
<u>#:</u>	<u>Step actions:</u>	<u>Expected Results:</u>
1	Open the mobile application material Flow	-The application should open and display the login and password options
2	Login to the application as setup user	-The application should display the configuration area with the server address, inactivity time (minutes) fields and show debug logs option
3	Changes the server address adding a wrong URL	Actual Results: <div>-The application can save the url without verify the URL connection, the URL data doesn't have restriction to add different types of values like #\$\$%^&*(<></div> <div>- The application is crashed after added those type of values</div>
		Expected Results: <div>- The application should verify the URL connection when the user save the data.</div> <div></div>

[AND]-BUG-ZMF-02: The application display display a error message when the user assign a part route in the first time

<u>Author:</u>	cmendoza
----------------	----------

#:	Step actions:	Expected Results:
1	Open the mobile application material Flow	-The application should open and display the login and password options
2	Login to the application as admin user	-The application should display the different areas added in the mobile like: asset, part routes, route selection, scan to pick, material pick, material delivery, exceptions, logout
3	Select the option route selection	The application should go to the route selection area to select a route type
4	Select a route type from the dropdown "my current assigned routes" and select a route type	the application should display the routes create on this route type.
5	Select/check a route from the list and click the save button icon	<p>Actual Results: The application displays just display the loading button</p> <p>Expected Results: The application should display the part route assigned</p> 

5. Conclusiones y recomendaciones

Los bugs reportados por el desarrollo del test plan deben ser resueltos por el área de DEV Mobile y luego correr de nuevo el test plan más una regresión test con los bugs resueltos.

Seria adecuado para el test plan Mobile ampliar el número de test cases automatizados para aumentar la cobertura del test plan.

Los test automatizados para el área de Rest API puede aumentar su efectividad al cubrir los diferentes tipos de repuesta que lanza el Rest API.

La herramienta utilizada para la creación de test cases automatización como es Robot Framework ayudan al desarrollo acelerado de test cases automatizados por su simplicidad y integración con Selenium.

Appium como herramienta para el área de mobile tendría que ser evaluado para ser integrado en el proyecto si este puede ser usado para una integración continua o E2E.