DP II - 2022-2023

***TESTING - INDIVIDUAL - REPORT***



Repositorio: [***https://github.com/marolmmar1/C2.X02.git***](https://github.com/marolmmar1/C2.X02.git)

Tutor: RAFAEL CORCHUELO GIL

Alumno: MARCOS OLMEDO MARíN

Grupo - C2.X02

# **Índice**

[Resumen Ejecutivo 2](#_heading=h.1fob9te)

[Tabla de Versionado 2](#_heading=h.k7rnd2bgq242)

[Introducción 2](#_heading=h.6ruaw3qhvhwb)

[Contenido 3](#_heading=h.aj9xrh3cv09a)

[About Testing 3](#_heading=h.ahcfyj8qenb9)

[Performance Requests Reports 8](#_heading=h.6n908yt3v045)

[Performance Tests Reports 9](#_heading=h.a0t4rsjlg31t)

[Z-Test 11](#_heading=h.su2ap5tq9k9)

[Lessons Learnt About Testing 12](#_heading=h.6n4odws6wxkd)

[Conclusiones 12](#_heading=h.1lhqo0ap1xzs)

[Bibliografía 12](#_heading=h.5bmrxveswtdo)

# **Resumen Ejecutivo**

A continuación la finalidad del presente documento es la de recopilar y documentar el análisis de todos los tests realizados para el “Student #2” los cuáles se han realizado durante el Sprint final (Sprint 4) proporcionando resultados de que las *features* realizadas cumplen con la expectativas y no presentan ningún bugs que implique a nuestro cliente no estar satisfecho con nuestro trabajo.

Además en el siguiente documento se ha analizado el rendimiento de su ejecución de la aplicación comparándolo con diferentes dispositivos (siendo PC1 y PC2).

# **Tabla de Versionado**

|  |  |  |
| --- | --- | --- |
| **Versión** | **Fecha** | **Descripción** |
| **1.0** | 09/07/2023 | Primera versión del documento |

# 

# **Introducción**

En la siguiente sección se procederá a explicar cómo se han realizado las pruebas funcionales del requisito solicitado así como realizar la comparativa entre el mejor y peor dispositivo con los reportes aportados: ***tester-request-performance.csv*** y ***tester-test-performance.csv***

# **Contenido**

# **About Testing**

**Pruebas Funcionales > List Tutorial> Test Positivo**

**Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente con confianza media**

En este test en primer lugar, lo que hacemos es iniciar sesión como ***assistant4***. Una vez iniciada sesión, accederemos al menú de asistente y haremos click en “Tutorial List”, con lo que se nos mostrará la lista de forma ascendente de los tutoriales de este asistente.

Tras ordenar el listado, se irán comprobando los valores presentes en la aplicación con los esperados del csv.

Si todo se realiza de forma correcta cerrará sesión y el test habrá concluido con éxito.

Este tipo de prueba nos permite asegurar que los *tutoriales* asociados a ese usuario son correctos y que no aparecen otros pertenecientes a otros usuarios.

**Pruebas Funcionales > List Tutorial >Test Hacking**

**Texto

Descripción generada automáticamente**

En este test, lo que hacemos es ir probando que ninguno de los demás roles, exceptuando el nuestro de *assistant*, puedan acceder a los listado de los *tutorials* y en ese caso, de intentar acceder, se devuelve un *error 500* indicando que no estamos autorizados.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**Pruebas Funcionales > Show Tutorial> Show Positive**

Tras iniciar sesión con *assistant4*, accedemos al listado de tutoriales. Iteramos uno por uno comprobando que los datos existentes son coherentes con los datos esperados.

.

**Pruebas Funcionales > Tutorial> Show Tutorial> Test Hacking**

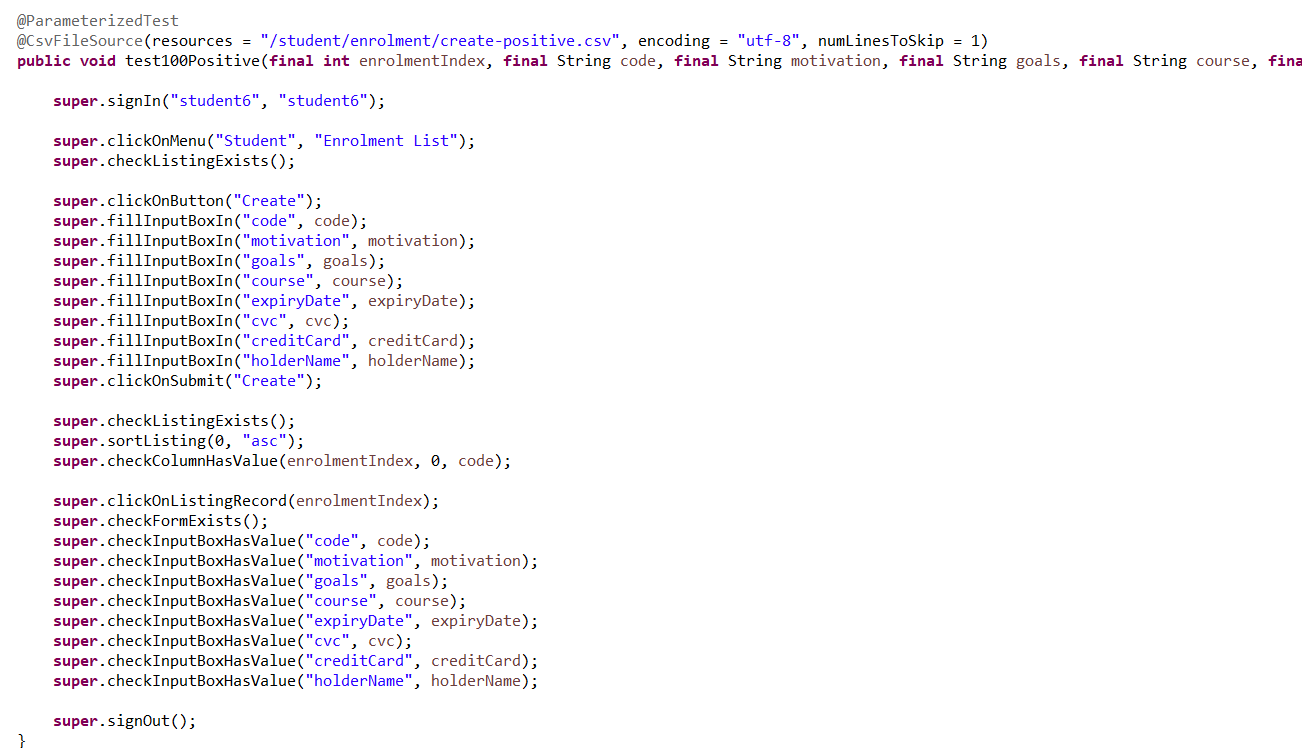
Texto

Descripción generada automáticamente con confianza baja

Tras listar el conjunto de tutoriales del asisitente 3, intentamos acceder aquellos tutoriales que siguen estando en modo borrador con otros perfiles.

Esto se hace de esta manera para reutilizar la función en tests como update o publish, para asegurarnos que la prohibición surge de la autorización del perfil que está intentando acceder a la url y no porque el tutorial no sea editable.

**Pruebas Funcionales > Assistant> Create Tutorial> Test Positive**

****

En este test, al igual que en el List Positivo, lo que hacemos es iniciar sesión con *student6*, esto hará que nos redirija al menú de *student*, se nos mostrará la lista de forma ascendente de los *enrolments* asociados y se comprobará que los valores de las columnas coincide con los que le hemos metido como parámetro. Si todo se realiza de forma correcta cerrará sesión y el test estará correcto.

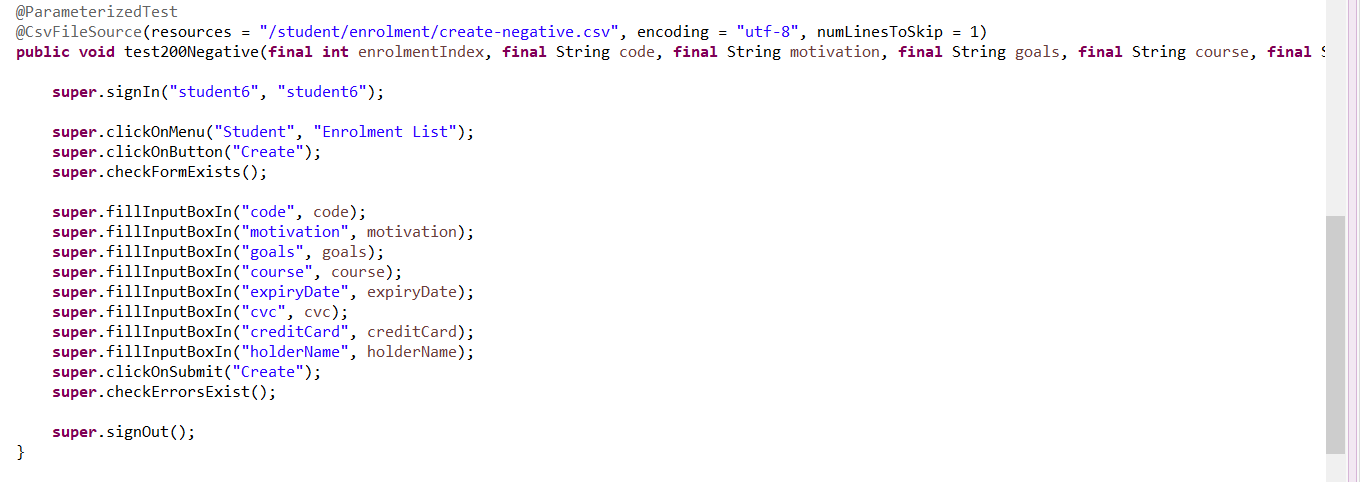
Una vez que se ha creado correctamente comprobamos que existe la lista de *enrolments* asociado a ese rol. Comprobamos que la lista muestra correctamente los datos del *enrolments* que hemos creado y que al pulsar sobre él y redirigir al formulario de show se comprueba que los datos de cada campo sean los que hemos añadido.

Por último se comprueba que hay un botón denominado *Create* y que al pulsar sobre él da lugar a una lista de *enrolments.*

***El ejecutar los test de esta manera nos permite detectar que los valores metidos en el formulario cumple con las reglas de negocio impuestas y no saltan errores, y por lo tanto nos permite crear el formulario correctamente.***

***Además se comprueba que los datos metidos en cada campo, al listar y mostrar en el show, son los que hemos metido.***

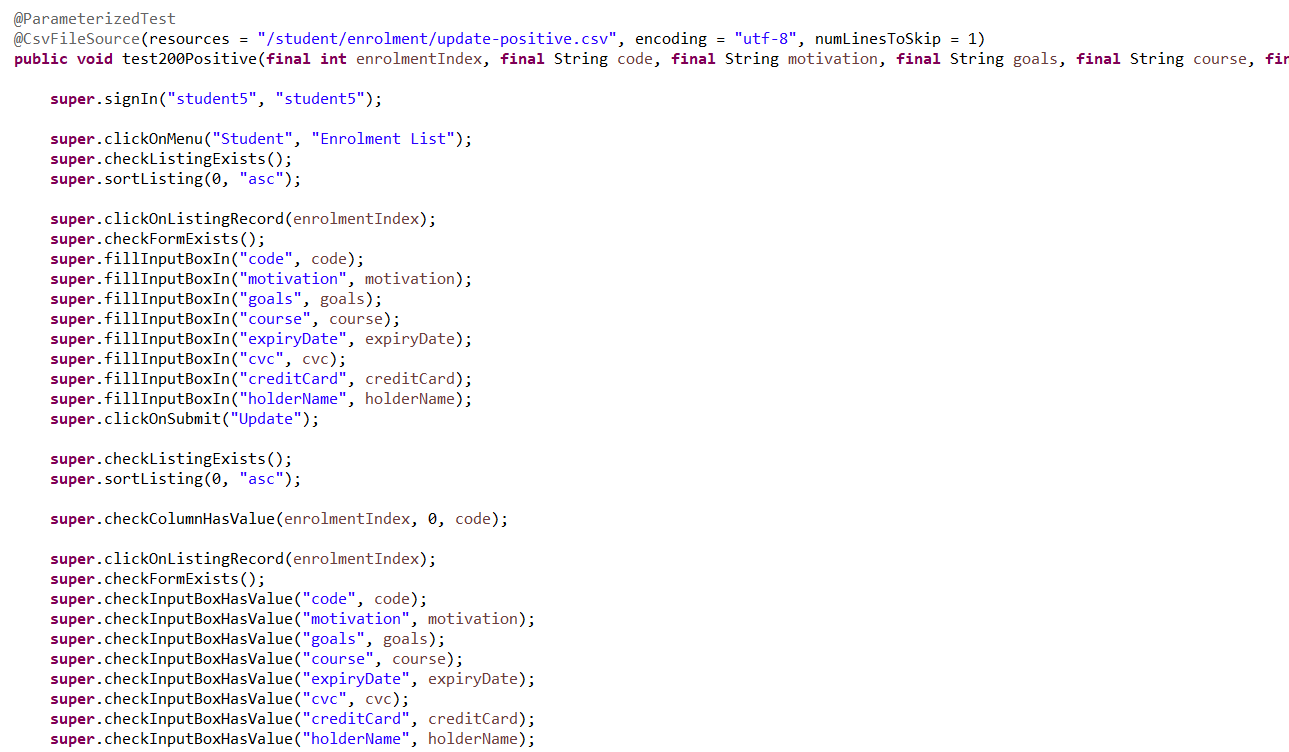
**Pruebas Funcionales > Student> Create Enrolment> Test Negative**



De igual manera, lo que hacemos es iniciar sesión con *student 6* esto hará que nos redirija al menú de *enrolments*, pulsamos el botón de create y comprobamos que sale el formulario de creación, se añadirá valores negativos en uno de los campo por cada iteración para así comprobar tras pulsar el botón create que en ese campo muestra un mensaje de error.

***Al hacer esto comprobamos que al introducir valores que no cumplen con las reglas de negocio, salta un error y así podemos comprobar que no hay error en la implementación de los mismos.***

**Pruebas Funcionales > Student> Update Enrolment> Test Positive**

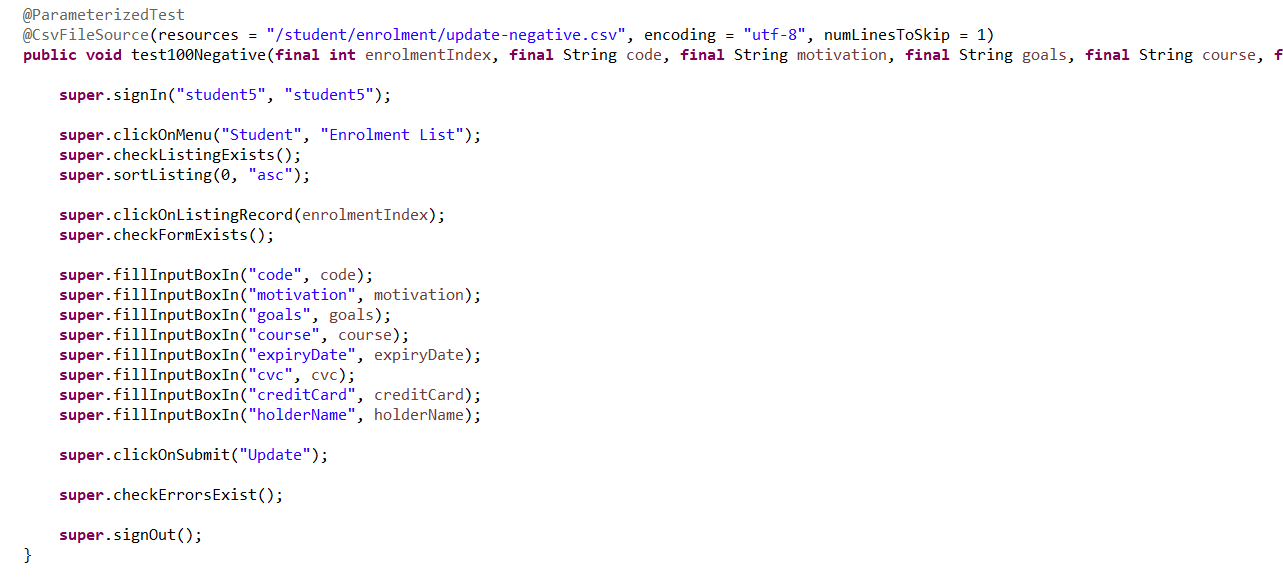


En este test, lo que hacemos es iniciar sesión con un *student5*, esto hará que nos redirija al menú de *student*, se nos mostrará la lista de *enrolments* de forma ascendente, pulsamos en cada *enrolment* asociados a ese student y nos aparecerá el formulario con todos los datos, que se actualizará con valores positivos en cada campo y tras esto se pulsará el botón de *Update*.

Una vez que se ha creado correctamente comprobamos que existe la lista de *enrolments* asociado a ese *student*, comprobamos que la lista muestra correctamente los datos del *enrolment* que hemos actualizado y que al pulsar sobre él y redirigir al formulario de show se comprueba que los datos de cada campo sean los que hemos añadidos.

***El ejecutar los test de esta manera nos permite detectar que los valores metidos en el formulario cumple con las reglas de negocio impuestas y no saltan errores y permite crear el formulario correctamente.***

**Pruebas Funcionales > Student> Update Enrolment> Test Negative**

******

De igual manera que se ha mencionado anteriormente, se inicia sesión como *student5*. En este caso, se actualizará el listado de *enrolments* con valores negativos los cuales no cumplen las reglas de negocio en cada campo y tras esto se pulsará el botón de Update.

***El ejecutar los test de esta manera nos permite detectar que los valores metidos en el formulario no cumple con las reglas de negocio impuestas, saltan excepciones y no permite crear el formulario correctamente.***

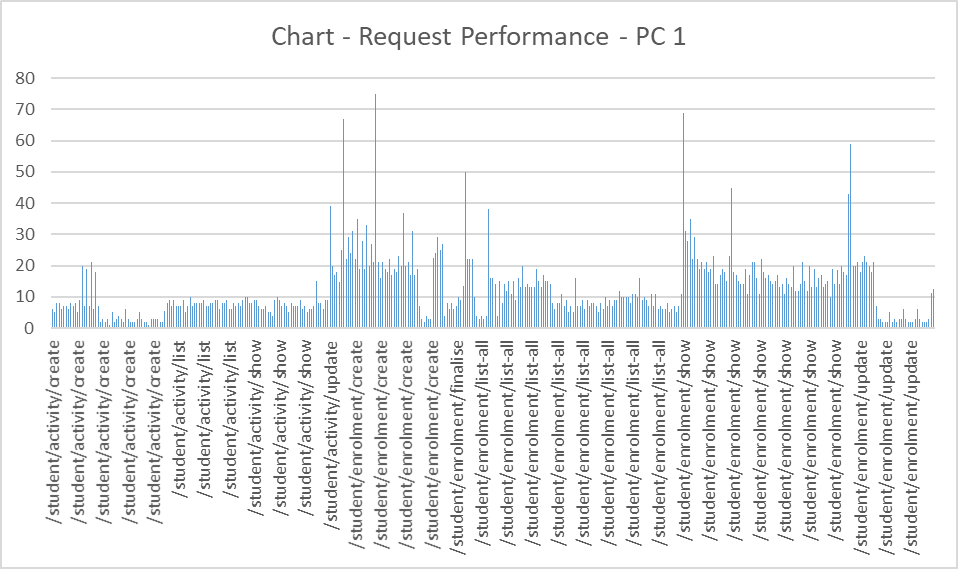
***….. (de igual manera con activity…).....***

# **Performance Requests Reports**

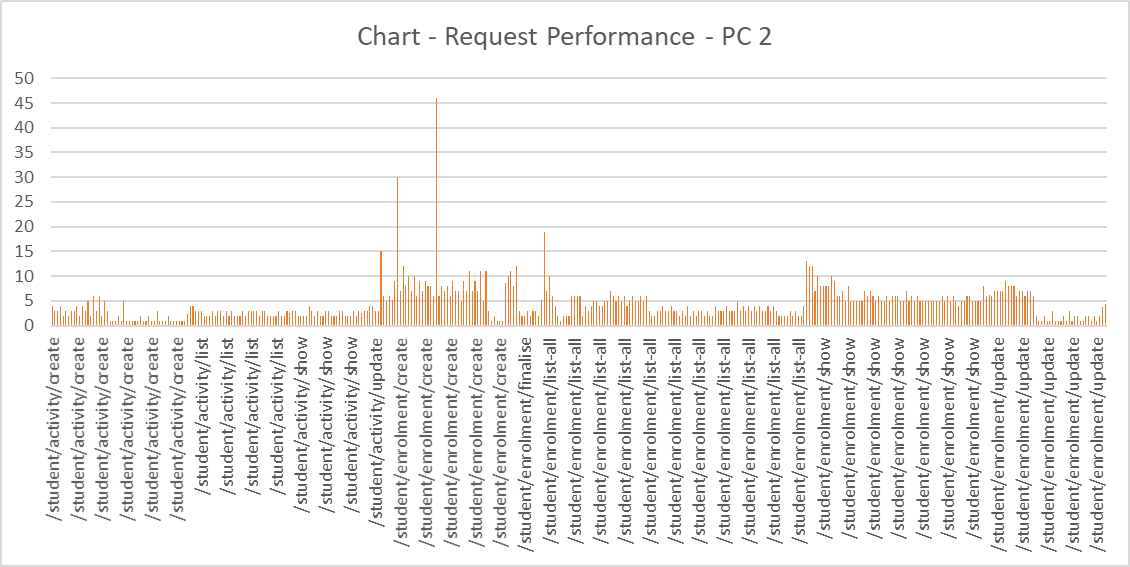
A continuación se han generado los reportes de los **performance request** y **performance testing** con el mejor dispositivo y el peor dispositivo del equipo para tener un estudio aproximado de cómo funciona el software en el mejor y en el peor de los casos; ambos obtuvieron muy buenos tiempos. Es por ello que en este documento hemos reflejado los gráficos de uno de ellos para mostrar estos resultados.

* Gráficos

Para la realización de este apartado se ha agrupado en una hoja de excel los tiempos recogidos en los reports performance requests y se han agrupado por su simple-path para calcular el promedio de tiempo invertido en estos mismos.

**

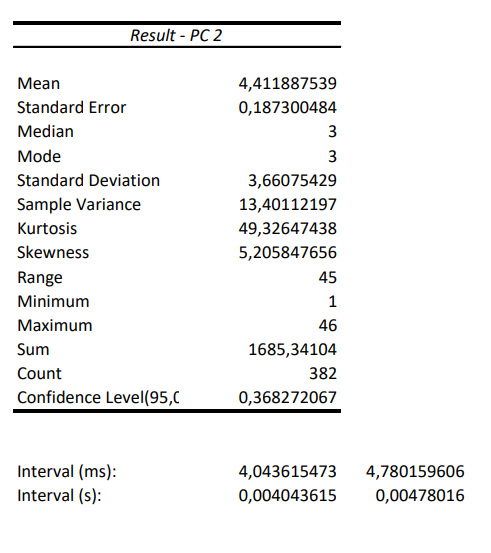
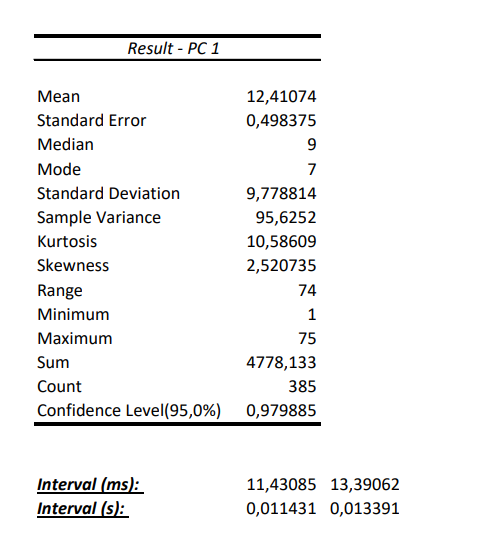
*Gráfico del promedio del tiempo de las solicitudes - PC 1*

**

*Gráfico del promedio del tiempo de las solicitudes -* ***PC 2***

* Análisis de datos

Para la realización de este apartado se ha llevado a cabo un análisis estadístico de los tiempos obtenidos en los reportes, haciendo una comparativa de los resultados obtenidos entre dos dispositivos del grupo, para posteriormente analizar los tiempos en cada dispositivo.

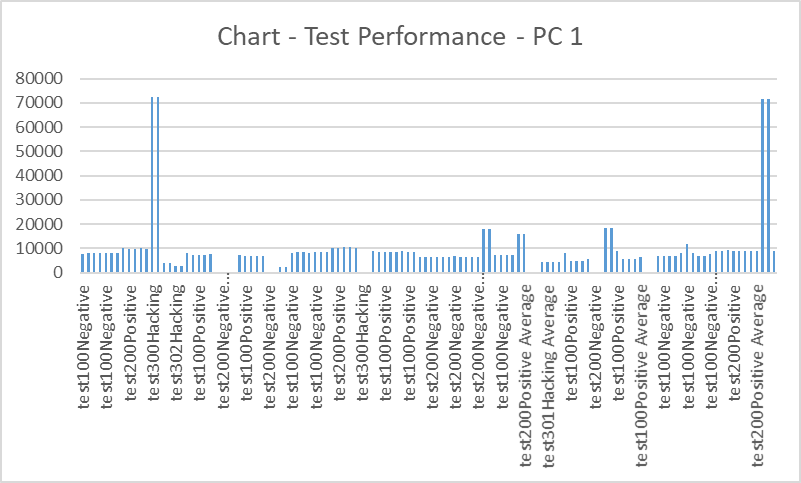
**

# **Performance Tests Reports**

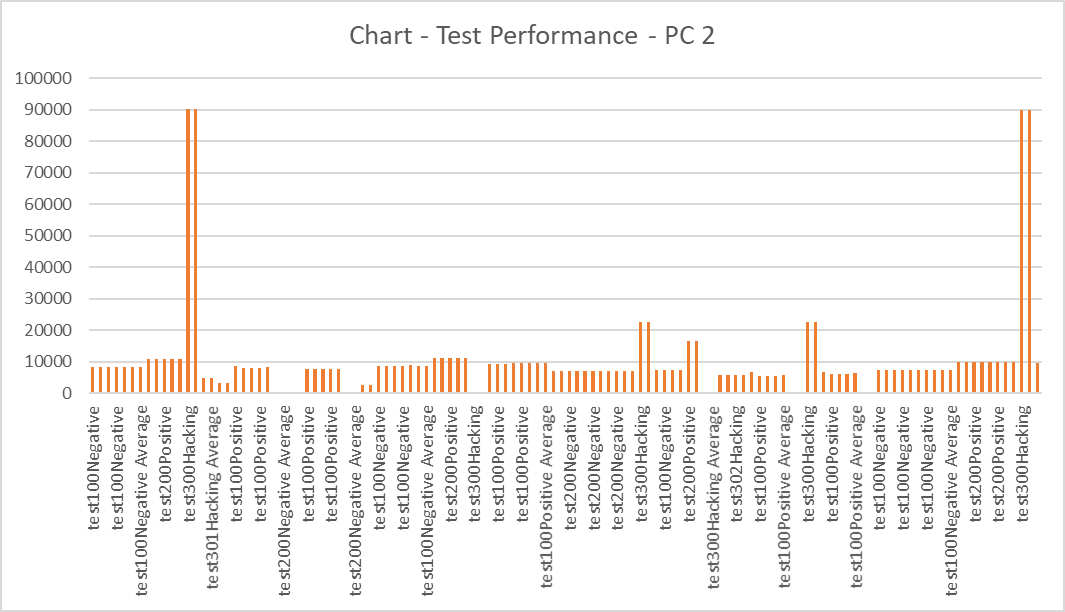
A continuación se han generado los reportes de los **performance request** y **performance testing** con el mejor dispositivo y el peor dispositivo del equipo para tener un estudio aproximado de cómo funciona el software en el mejor y en el peor de los casos; ambos obtuvieron muy buenos tiempos. Es por ello que en este documento hemos reflejado los gráficos de uno de ellos para mostrar estos resultados.

* Gráficos

Para la realización de este apartado se ha agrupado en una hoja de excel los tiempos recogidos en los reports performance tests y se han agrupado por su test-class y por su test-method después, pudiendo así calcular el promedio de tiempo invertido en cada método de cada clase.

**

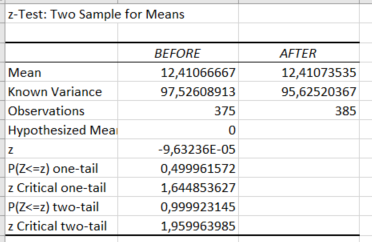
*Gráfico del tiempo promedio de ejecución de los tests (de 0 a algo más de 70 segundos) - PC1*

**

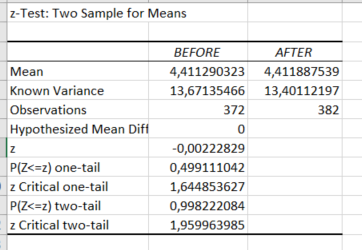
*Gráfico del tiempo promedio de ejecución de los tests (de 0 a 90 segundos) PC1*

# **Z-Test**

* **Z-Test (Before-After) con PC 1:**



* **Z-Test (Before-After) con PC 2:**



*[ ⍺ = 1- confidence level ]*

*PC 1: ⍺ = 1 - 0,979885 = 0,020115*

*PC 2 : ⍺ = 1- 0,368272067 = 0,631727933*

# |————|——————————————|——|

*0,00* *⍺ = 0,020115* *P(Z<=z) two tail = 0,99 1,00*

*⍺ = 0,631727933*

*Como observamos el P-Value (* *P(Z<=z) two tail ) se encuentra entre ⍺ y 1.00 , es decir* ***(⍺ ,1.00]*** *esto quiere decir que los cambios no han supuesto ninguna mejora significativa, es decir, los tiempos de muestreo son diferentes, pero globalmente son los mismos. Por lo que podemos concluir con que no podemos asegurar cual de los dos PC (PC 1 o PC 2 ) es más rápido.*

# **Lessons Learnt About Testing**

A continuación se enumeran las lecciones aprendidas durante este proyecto en grupo acerca del “testeo” llevado a cabo durante este proyecto. Podemos destacar las siguientes lecciones aprendidas como ventajas del testing:

* Permite una organización de una forma más óptima los tests.
* Mejora la gestión del alcance de los test, aportando una mejor optimización en cuanto al tiempo.
* Podemos ver que sin duda el testeo influye muy notable y positivamente respecto a la calidad de código, puesto que podemos detectar errores en una etapa más temprana de desarrollo y de forma más rápida.
* Observando también cómo favorece al trabajo generalizado, pues permite trabajar de manera más ágil.
* Sobre todo cara a los costes de mantenimiento del proyecto, esta es una buena manera de sobrellevarlos.

# **Conclusiones**

Intencionalmente en blanco.

# **Bibliografía**

Intencionalmente en blanco.