

# Identificación y justificación de los *bugs*

Para la segunda entrega, y de acuerdo a lo solicitado por la rúbrica, elegimos reparar los siguientes bugs detectados:

- a. *Arreglar la funcionalidad de Exportación/Importación; y,*
- b. *Arreglar el modificar concierto para mantener a los artistas.*

Ambas tareas fueron seleccionadas debido a la importancia de acuerdo a los requerimientos del proyecto y su impacto en la funcionalidad general del sistema. La prioridad y severidad de los errores fueron consideradas para determinar la necesidad de solucionarlos de manera urgente. El criterio utilizado fue:

1. seleccionar los *bugs* mayor prioridad (determinada por el PO);
2. frente a *bugs* de la misma prioridad, los distinguimos seleccionando aquellos de mayor severidad (determinada por el tester).

En el caso de la funcionalidad de *Exportación/Importación de conciertos*, se trataba de un requerimiento solicitado por letra y era necesario de forma inmediata porque la funcionalidad no estaba correctamente implementada y no se podía utilizar. Por lo tanto, su prioridad era alta y requería una solución urgente.

En cuanto a la tarea de *arreglar el modificar concierto para mantener a los artistas*, se trataba de un error de severidad mayor y prioridad alta debido a que generaba una disfuncionalidad en el requerimiento *editar concierto* que se solicitaba en la letra, ya que por ejemplo, al editar la fecha, no se mantenían los artistas asociados al concierto. Esto afectaba la funcionalidad general del sistema y la capacidad del sistema de mantener a los artistas de manera efectiva. Por lo tanto, esta tarea también requería una solución inmediata.

Por último, queremos aclarar que en la lista de bugs encontrados e informados en la primera entrega, se incluyó el error de *El acomodador no tiene la opción de escanear ticket*, el cual se le había asignado una severidad crítica y una prioridad inmediata. En principio, nos enfocamos en solucionar este error, pero, posteriormente, descubrimos que la funcionalidad en realidad estaba ejecutándose correctamente –véase la imagen a continuación para observar la funcionalidad disponible en un perfil acomodador– y que se trató de un error al momento de realizar el *testing* y el informe de *bugs*, por lo que procedimos a trabajar en el siguiente *bug* aplicando los criterios anteriormente mencionados.

## Arenas Gestor

Home Tickets ▾ Protagonistas ▾

Horacio Acomodador Acomodador ▾

Escanear tickets

Escanear ticket

Id

Escanear

## Bug: Arreglar la funcionalidad de Exportación/Importación

### Descripción de arreglo:

Las pruebas integradas en la solución recibida no verificaban efectivamente esta funcionalidad. Todos los *test* se realizaban con *mocks* de la base de datos, de los servicios y no existía la invocación real de la lógica de importación y exportación en ellas. Debido a esto, nunca fue verificado que la lógica del negocio realmente permitiera la importación y exportación a través de las mismas reglas utilizadas para los conciertos generados localmente.

El principal problema identificado fue que el formato establecido para la importación de conciertos no especificaba algunos objetos del dominio creados específicamente en la solución recibida (no se podían derivar de la letra asociada a este trabajo), por lo tanto se establecía una asociación entre entidades mucho más restrictiva de la cual se desconocía todo detalle al momento de importar los conciertos.

En la resolución se propone la utilización de una *location* base para todos los conciertos importados (ubicación que tiene que estar generada en la base de datos antes de importar los conciertos). Entendemos que, dada las características y restricciones solicitadas en la letra anterior, utilizar la ubicación del Antel Arena para estos efectos es lo más adecuado.

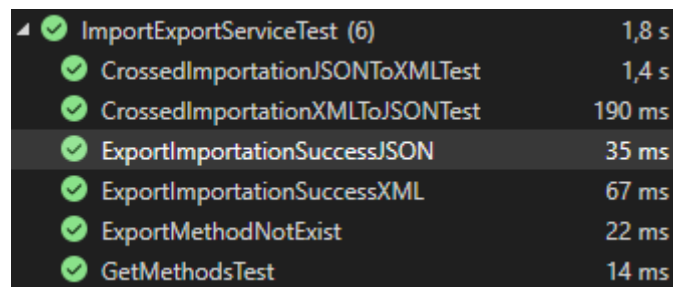
Adicionalmente, se intentaba realizar el tracking de las entidades asociadas para la inserción de un concierto importado (recordar que la lógica implementada necesita más información de la recibida en la ejecución de la importación), por lo que implementamos un método que permita utilizar la lógica de inserción de conciertos pero que en el acceso a datos esté diferenciada de la anterior.

Por último, fue necesario realizar varios ajustes al mapeo de los objetos del dominio en entidades del tipo *Data Transfer Objects* y viceversa.

En líneas generales, entendemos que la solución propuesta a este *bug* puede evolucionar en una mejor versión, pero dadas las restricciones de tiempo asociadas a la modificación de diversos aspectos de la lógica no pueden ser incluídas en el alcance de esta entrega.

### Evidencia Test:

A continuación se adjunta una imagen con los test realizados automáticamente con importación y exportación de archivos tanto en formato XML como JSON.



ImportExportServiceTest (6)	1,8 s
CrossedImportationJSONToXMLTest	1,4 s
CrossedImportationXMLToJSONTest	190 ms
ExportImportationSuccessJSON	35 ms
ExportImportationSuccessXML	67 ms
ExportMethodNotExist	22 ms
GetMethodsTest	14 ms

**Bug:** Arreglar el modificar concierto para mantener a los artistas

**Descripción de arreglo:**

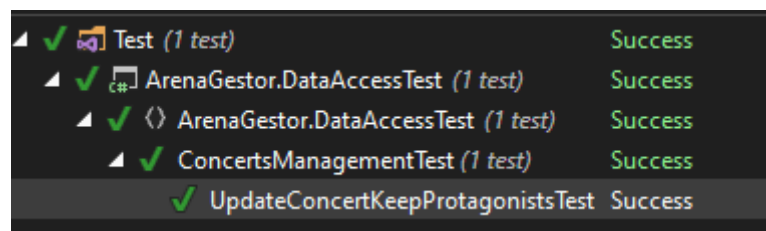
El sistema borraba los protagonistas del concierto porque al realizar el update de la entidad *Concert* en la base de datos no se hacía el respectivo attach de la entidad *ConcertProtagonist* asociada.

Se procedió a realizar el cambio correspondiente en la clase *ConcertsManagement* ubicada en el paquete de *DataAccess*, pues es quien tenía la responsabilidad correspondiente a la interacción del sistema con la base de datos respecto a la entidad conciertos.

**Evidencia Test:**

El cambio fue realizado mediante TDD, procediendo a crear un Test que probara la situación que origina el bug. En este caso la prueba consistió en hacer un update de Concierto editando su fecha y verificando que los artistas continuaban asociados al evento.

Evidencia de la etapa verde de la prueba unitaria:



Se adjunta el [enlace al video de la revisión con el PO](#).