En este documento vamos a redactar algunos fallos que hemos encontrado con la ayuda de los tests. Hemos comentado solamente aquellos que pensamos que son diferentes, ya que esta idea se nos propuso el día anterior a la entrega y no hemos tenido tiempo para redactar todos los errores encontrados.

Cambio en AbstractTest.java

En el primer cuatrimestre, implementamos un nuevo método en LoginService.java que se encarga de devolver verdadero o falso según haya o no un usuario autenticado en ese momento. Dicho método se llama: isAuthenticated().

Sin embargo, al ejecutar los test, y simular el proceso de login para poder comprobar que estamos autenticados desde JUnit producía un error, en el nuevo método. Para solucionarlo cambiamos el valor de *autenticationToken* para que en vez de ser nulo si el nombre de usuario que le pasaban era nulo, se inicializara, con todos sus parámetros a nulo:

```
authenticationToken = new TestingAuthenticationToken(null, null);
```

Fallo Hacking

En algunos métodos del servicio los cuales no se reconstruyen y a su vez se utilizan para cambiar algún objeto del dominio, como podría ser algunos métodos de borrado. Al intentar borrarlo con ataque post, lo podías borrar sin ser tuyo. Vamos a explicar esto último con un ejemplo:

Si tenemos un Service que tiene un Manager distinto al que se encuentra autenticado, si al borrar este Service se intenta cambiar el Manager, poniendo aquel que esta autenticado, como las restricciones se comprobaban sobre el propio objeto que le llega al servidor, cualquier usuario podría borrar cualquier Service.

Nos dimos cuenta al hacer los test de Service y la solución consiste en obtener de la base de datos el Service que se ha solicitado borrar y a partir de ahí realizar las comprobaciones oportunas.

Creación RSVP

Al crear un RSVP el orden de las preguntas cambiaba, de esta forma, si el usuario dejaba alguna sin rellenar en algunas ocasiones, al mostrar que el formulario tenía fallos, cambiaba el orden de las preguntas y respuestas. El problema era que usábamos HashMap, el cual no mantenía el orden.

Nos dimos cuenta con el test de crear RSVP y comenzamos a usar LinkedMap.

Category

A la hora de reorganizar las categorías, no controlábamos que una categoría pudiera ponerse como padre a sí misma. Esto, al ejecutar el test y ver sus categorías hijas después de reorganizar provocaba un bucle infinito. La solución fue colocar esta comprobación en el servicio de categoría:

```
if (newFather != null)
```

```
Assert.isTrue(!(category.equals(newFather)));
```

Nuevo rol

Tenemos ciertos métodos que comprueban si el usuario que está accediendo a algo relacionado con una cita es menor de edad, para impedir que estos usuarios vean alguna entidad que tenga contenido mayor de edad.

Estos métodos usaban comparaciones que miraban si el actor autenticado era un usuario. En un principio se nos olvidó añadir el manager a estas comprobaciones, con los test de Rendezvous nos dimos cuenta y solventamos este problema.