Plan de pruebas – US423825-FiltrarEventosPorTipo (David Moreno Pérez)

Los niveles de prueba que se van a aplicar son los siguientes:

- Pruebas de aceptación: las pruebas de aceptación se definirán siguiendo una estrategia basada en historias de usuario y serán ejecutadas de forma manual.
- Pruebas de integración: estas pruebas verifican la interacción entre clases. Se llevarán a cabo usando el framework Espresso y JUnit.
- Pruebas de interfaz: se pruebas todos los elementos desde la interfaz, comprobando que la funcionalidad es correcta y que las salidas son las esperadas respecto a las entradas que se han introducido. Para ello, se ha usado el framework Espresso, que nos permite automatizar las pruebas.

PRUEBAS DE ACEPTACIÓN

Se muestran las pruebas en base a los criterios de aceptación acordados con el *Product Owner*.

Prueba 00: Éxito

1. El usuario selecciona filtrar.
2. El sistema crea una ventana emergente con los tipos de evento.
3. El usuario ajusta los filtros.
4. El usuario selecciona aplicar.
5. Se cierra la ventana emergente
6. El sistema muestra la lista de eventos con el filtro aplicado

Prueba 01: Cancelar

1. El usuario selecciona filtrar.
2. El sistema crea una ventana emergente con los tipos de evento.
3. El usuario ajusta los filtros.
4. El usuario selecciona cancelar.
5. Se cierra la ventana emergente
6. El sistema muestra la lista de eventos previa a la selección del botón filtrar.

Prueba 02: El sistema pierde la conexión a internet

1. El usuario selecciona filtrar por un tipo de evento.
2. El sistema crea una ventana emergente con los tipos de evento.
3. El usuario ajusta los filtros.
4. El usuario selecciona aplicar.
5. Se cierra la ventana emergente.
6. El sistema muestra la lista de eventos previa a la selección del botón filtrar con los filtros aplicados.

Prueba 03: El sistema no puede acceder al servicio de datos

1. El usuario selecciona filtrar por un tipo de evento.
2. El sistema crea una ventana emergente con los tipos de evento.
3. El usuario ajusta los filtros.
4. El usuario selecciona aplicar.
5. Se cierra la ventana emergente.
6. El sistema muestra la lista de eventos previa a la selección del botón filtrar con los filtros aplicados.

Prueba 04: El sistema muestra resultados anómalos

1. El usuario selecciona filtrar por un tipo de evento.
2. El sistema crea una ventana emergente con los tipos de evento.
3. El usuario ajusta los filtros
4. El usuario selecciona aplicar.
5. Se cierra la ventana emergente
5. El sistema detecta que no hay eventos de ese tipo.
6. El sistema muestra un pop-up indicándole al usuario que no hay ningún evento del tipo especificado.
7. El sistema muestra la lista de eventos previa a la selección del botón filtrar.

Prueba 05: Situación anómala (evento sin tipo)

1. El usuario selecciona filtrar por un tipo de evento.
2. El sistema crea una ventana emergente con los tipos de evento.
3. El usuario ajusta los filtros.
4. El usuario selecciona aplicar.
5. Se cierra la ventana emergente.
6. El sistema detecta que hay eventos sin tipo.
7. El sistema obvia estos eventos a la hora de filtrar.
8. El sistema muestra la lista de eventos con el filtro aplicado.

PRUEBAS DE INTEGRACIÓN

Se aplica prueba de métodos, siendo los casos de prueba definidos para cada método los siguientes:

Método **onFiltrarClicked(List<String> checkboxSeleccionados)** de la clase **EventsPresenter.java**

Casos de prueba válidos:

Identificador	**Entrada**	**Resultado**
IT. 1A	Lista con el tipo de evento Música.	Se actualiza la lista de eventos conteniendo los de tipo Música.
IT. 1B	Lista vacía.	Se actualiza la lista de eventos conteniendo la lista original de eventos.

IT. 1C	Lista con todos los tipos de evento existentes.	Se actualiza la lista de eventos con todos los tipos de eventos exceptuando los eventos sin tipo.

Casos de prueba no válidos:

Identificador	**Entrada**	**Resultado**
IT. 1D	Lista que contiene un tipo de evento no existente.	Se actualiza la lista de eventos conteniendo la lista original de eventos.
IT. 1E	Lista con elemento null.	nullPointerException.

Método **loadData()** de la clase **EventsPresenter.java**

Casos de prueba válidos:

Identificador	**Entrada**	**Resultado**
IT. 1A	*	Se obtiene la lista de eventos correctamente.

Casos de prueba no válidos:

Identificador	**Entrada**	**Resultado**
IT. 1B	*	No se obtiene la lista de eventos.

Nota*: en este caso el método loadData() no recibe ningún parámetro, simplemente carga a la lista "cachedEvents" los eventos si todo ha ido bien. Por tanto, se comprobará, cambiando la URL de la fuente de datos.

PRUEBAS DE INTERFAZ DE USUARIO

Se realizarán tres pruebas:

A. No se selecciona tipo de evento.

1. Se abrirá la interfaz pulsando el botón "Filtrar".
2. Se comprobará que no hay ningún tipo de evento seleccionado.
3. El usuario pulsará el botón "Aplicar" de la ventana flotante sin seleccionar ningún tipo de evento.
4. Se comprobará que se muestra toda la lista de eventos sin aplicar ningún filtro.

B. Seleccionar un tipo de evento.

1. Se abrirá la interfaz pulsando el botón “Filtrar”.
2. Se comprobará que no hay ningún tipo de evento seleccionado.
3. Se seleccionará un tipo de evento.
4. El usuario pulsará el botón “Aplicar” de la ventana flotante.
5. Se comprobará que los eventos mostrados se corresponden al filtro seleccionado previamente.

C. Seleccionar todos tipos de eventos.

1. Se abrirá la interfaz pulsando el botón “Filtrar”.
2. Se comprobará que no hay ningún tipo de evento seleccionado.
3. Se seleccionarán todos los tipos de eventos.
4. El usuario pulsará el botón “Aplicar” de la ventana flotante.
5. Se comprobará que los eventos mostrados se corresponden a los filtros seleccionados previamente.