Plan de pruebas #505806-Buscar Por Provincia Y Localidad[1]

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

- Pruebas de aceptación. Las pruebas de aceptación se definirán siguiendo los criterios de aceptación definidos en la historia de usuario y serán ejecutadas de forma manual en el propio dispositivo del Product Owner u otro que se le proporcione.
- Pruebas unitarias. Se utilizará técnica de prueba de métodos y de caja negra (partición equivalente y AVL) para la definición de los casos de prueba de cada método. Será necesaria la utilización de Junit y Mockito.
- Pruebas de integración. Estas pruebas verifican la interacción entre clases. Se llevarán a cabo usando el framework JUnit y las librerías Mockito
- Pruebas de interfaz. Se probará la funcionalidad de todos los componentes de la aplicación de manera conjunta. Se llevarán a cabo usando el framework Espresso sobre JUnit.

PRUEBAS DE ACEPTACIÓN

En base a los criterios de aceptación se identifican los siguientes escenarios:

A1. CA: Caso de éxito: Búsqueda exitosa con provincia y municipio.
1. El usuario pulsa el botón de filtros en la toolbar.
2. El usuario selecciona la provincia en el menú desplegable e introduce el municipio.
3. El usuario pulsa el botón “buscar”.
4. La aplicación muestra una lista con las gasolineras encontradas en el municipio seleccionado.
5. La aplicación muestra un toast con el número de gasolineras encontradas.

A2. CA: Caso de éxito: Búsqueda exitosa con provincia sin municipio.
1. El usuario pulsa el botón de filtros en la toolbar.
2. El usuario selecciona la provincia en el menú desplegable y deja vacía la localidad.
3. El usuario pulsa el botón “buscar”.
4. La aplicación muestra una lista con las gasolineras encontradas en la provincia seleccionada.
5. La aplicación muestra un toast con el número de gasolineras encontradas.

A3. CA: Caso de éxito: Búsqueda exitosa sin provincia con municipio.
1. El usuario pulsa el botón de filtros en la toolbar.
2. El usuario no selecciona ninguna provincia e introduce el municipio.
3. El usuario pulsa el botón de “buscar”.
4. La aplicación muestra una lista con todas las gasolineras encontradas en el municipio.
5. La aplicación muestra un toast con el número de gasolineras encontradas.

A4. CA: Caso de éxito: Búsqueda exitosa sin filtros aplicados.
1. El usuario pulsa el botón de filtros en la toolbar.
2. El usuario pulsa el botón de “buscar” sin introducir ninguna provincia ni municipio.
3. La aplicación muestra una lista con todas las gasolineras encontradas.
4. La aplicación muestra un toast con el número de gasolineras encontradas.

1. Aunque se llama PorProvinciaYLocalidad nos han recomendado que se busque por municipio por cómo funciona la API.
* No se define el caso de error en acceso a BD ya que no es necesario en esta historia de usuario.

A5. CA: Datos no válidos. (Difiere con lo definido en scrum)
1. El usuario pulsa el botón de filtros en la toolbar.
2. El usuario introduce un municipio inexistente o que no pertenece a la provincia seleccionada.
3. La aplicación muestra la lista que se estaba mostrando anteriormente.
4. La aplicación muestra un toast indicando el error.

A6. CA: Pérdida de conexión al iniciar la aplicación.
1. El usuario abre la aplicación.
2. Se pierde la conexión a internet mientras se inicializa.
3. La aplicación muestra un mensaje de error.
4. La aplicación muestra una lista vacía.

A7. CA: Error en acceso al servicio de datos
1. El usuario abre la aplicación.
2. Ocurre un error en el acceso al servicio de datos (API no responde o falla la solicitud) mientras se inicializa.
3. La aplicación muestra un mensaje de error.
5. La aplicación muestra una lista vacía.

A8. CA: Resultado con lista vacía
1. El usuario pulsa el botón de filtros en la toolbar.
2. El usuario selecciona una provincia y municipio donde no hay gasolineras disponibles.
3. El usuario pulsa el botón “buscar”.
4. La aplicación no encuentra gasolineras que coincidan con los filtros aplicados.
5. La aplicación muestra un mensaje indicando que no se encontraron resultados.
6. Se muestra una lista vacía de gasolineras.

A9. CA: Ausencia de datos importantes
1. El usuario pulsa el botón de filtros en la toolbar.
2. El usuario selecciona la provincia y/o municipio.
3. El usuario pulsa el botón “buscar”.
4. La aplicación encuentra gasolineras, pero algunas tienen datos incompletos (falta la provincia o el municipio).
5. La aplicación muestra un toast indicando que algunas gasolineras no se pueden mostrar debido a la falta de datos importantes.
6. La lista de gasolineras se muestra con solo aquellas que tienen datos completos.

Tabla 2. Casos de prueba de aceptación

Identificador	Entrada	Resultado
A1	(“Cantabria”, “Santander”)	Lista con 31 gasolineras
A2	(“Cantabria”, “”)	Lista con 162 gasolineras
A3	(“-”, “Santander”)	Lista con 31 gasolineras
A4	(“-”, “”)	Lista con 11.989 gasolineras
A5	(“Asturias”, “Santander”)	Lista anterior + mensaje municipio incorrecto
A6	( )	Lista vacía + mensaje error de conexión
A7	( )	Lista vacía + mensaje error servicio de datos
A8	(“Valencia”, “Emperador”)	Lista vacía + mensaje sin resultados
A9		Mensaje ausencia de datos

PRUEBAS UNITARIAS

En esta historia de usuario se probarán los métodos añadidos filtrarPorProvinciaYMunicipio de la clase Filtros y el método buscarGasolinerasConFiltros de la clase MainPresenter. Los resultados obtenidos en las pruebas se filtran en base a un fichero JSON el cuál se encuentra resumido en en Anexo 1.

Método filtrarPorProvinciaYMunicipio

List<Gasolinera> filtrarPorProvinciaYMunicipio(List<Gasolinera> gasolineras, String provincia, String municipio)

A. Provincia y municipio válidos.
B. Provincia válida y municipio vacío.
C. Provincia vacía y municipio válido.
D. Provincia válida y municipio vacío sin resultados.
E. Provincia vacía y municipio válido sin resultados.
F. Municipio no perteneciente a provincia.
G. Provincia y municipio válidos sin resultados.
H. Provincia y municipio vacíos.
I. Lista de gasolineras vacía.

Tabla 3. Casos de prueba para el método filtrarPorProvinciaYMunicipio de la clase Filtros

Identificador	Entrada	Valor esperado
UD.1A	(“Cantabria”, “Santander”)	[Cepsa, Repsol, Carrefour]
UD.1B	(“Cantabria”, “”)	[Cepsa, Repsol, Carrefour, Ballenoil, Shell]
UD.1C	(“-”, “Santander”)	[Cepsa, Repsol, Carrefour]
UD.1D	(“Madrid”, “”)	[ ]
UD.1E	(“”, “Tineo”)	[ ]
UD.1F	(“Asturias”, “Santander”)	[ ]
UD.1G	(“Asturias”, “Tineo”)	[ ]
UD.1H	(“-”, “”)	[Cepsa, Repsol, Carrefour, Ballenoil, Shell, Petronor, Avia, Cepsa]
UD.1I	-	null

Método buscarGasolinerasConFiltros

buscarGasolinerasConFiltros(String provincia, String municipio): void

A. Provincia y municipio válidos.
B. Provincia válida y municipio vacío.
C. Provincia vacía y municipio válido.
D. Provincia válida y municipio vacío sin resultados.
E. Provincia y municipio válidos sin resultados.
F. Provincia vacía y municipio válido sin resultados.
G. Municipio no perteneciente a provincia.
H. Provincia y municipio vacíos.

Tabla 4. Casos de prueba para el método buscarGasolinerasConFiltros de la clase MainPresenter

Identificador	Entrada	Valor esperado
UB.1A	(“Cantabria”, “Santander”)	Se llama al método *filtrarPorProvinciaYMunicipio* con finalProvincia = Cantabria y finalMunicipio = Santander. Además, se llama a showStations y showLoadCorrect(3).
UB.1B	(“Cantabria”, “”)	Se llama al método *filtrarPorProvinciaYMunicipio* con finalProvincia = Cantabria y finalMunicipio = null. Además, se llama a showStations y showLoadCorrect(5).
UB.1C	(“”, “Santander”)	Se llama al método *filtrarPorProvinciaYMunicipio* con finalProvincia = null y finalMunicipio = Santander. Además, se llama a showStations y showLoadCorrect(3).
UB.1D	(“Madrid”, “”)	Se llama al método *filtrarPorProvinciaYMunicipio* con finalProvincia = null y finalMunicipio = Santander. Además, se llama a showStations + mensaje sin resultados.
UB.1E	(“Asturias”, “Tineo”)	Se llama al método *filtrarPorProvinciaYMunicipio* con finalProvincia = Asturias y finalMunicipio = Tineo. Además, se llama a showStations + mensaje sin resultados.
UB.1F	(“”, “Tineo”)	Se llama al método *filtrarPorProvinciaYMunicipio* con finalProvincia = null y finalMunicipio = Tineo. Además, se llama a showStations + mensaje sin resultados.
UB.1G	(“Asturias”, “Santander”)	Lanzar error municipio incorrecto.
UB.1H	(“-”, “")	Se llama a showStations con la lista de todas la gasolineras y showLoadCorrect(8).

PRUEBAS DE INTEGRACIÓN

Se probará que el presenter funciona bien en conjunto con la view y la clase creada Filtros. Los resultados obtenidos en las pruebas se filtran en base a un fichero JSON el cuál se encuentra resumido en en Anexo 1. El orden de las pruebas y los casos de prueba a realizar serían los siguientes:

Tabla 5. Casos de prueba Integración entre la Vista, el Presenter y Filtros.

Entrada View	Acción Esperada (Presenter)	Resultado Esperado (View)
(“Cantabria”, “Santander”)	Se llama a Filtros.filtrarPorProvinciaYMunicipio con “Cantabria” y “Santander”	Mostrar lista y toast con 3 gasolineras [Cepsa, Repsol, Carrefour]
(“Cantabria”, “”)	Se llama a Filtros.filtrarPorProvinciaYMunicipio con “Cantabria” y null	Mostrar lista y toast con 5 gasolineras [Cepsa, Repsol, Carrefour, Ballenoil, Shell]
(“-”, “Santander”)	Se llama a Filtros.filtrarPorProvinciaYMunicipio con null y “Santander”	Mostrar lista y toast con 3 gasolineras [Cepsa, Repsol, Carrefour]

(“-”, “”)	Se llama a Filtros.filtrarPorProvinciaYMunicipio con null y null	Mostrar lista y toast con 8 gasolineras [Cepsa, Repsol, Carrefour, Ballenoil, Shell, Petronor, Avia, Cepsa]
(“Asturias”, “Santander”)	Lanzar error municipio	Mostrar lista anterior y toast con municipio incorrecto
(“Albacete”, “”)	Se llama Filtros.filtrarPorProvinciaYMunicipio con “Albacete” y null	Lista vacía y toast sin resultados
(“Galicia”, “”)	Se llama Filtros.filtrarPorProvinciaYMunicipio con “Galicia” y null	Mostrar lista y toast ausencia de datos [Avia]

PRUEBAS DE INTERFAZ

Las pruebas de interfaz coinciden con las pruebas de aceptación, sin embargo, se usarán los datos de las gasolineras del fichero JSON indicado en el Anexo 1.

Tabla 6. Casos de prueba UI.

ID	Entrada	Resultado
UI.1	(“Cantabria”, “Santander”)	[Cepsa, Repsol, Carrefour]
UI.2	(“Cantabria”, “”)	[Cepsa, Repsol, Carrefour, Ballenoil, Shell]
UI.3	(“-”, “Santander”)	[Cepsa, Repsol, Carrefour]
UI.4	(“-”, “”)	[Cepsa, Repsol, Carrefour, Ballenoil, Shell, Petronor, Avia, Cepsa]
UI.5	(“Asturias”, “Santander”)	Lista anterior + mensaje municipio incorrecto
UI.6	( )	[ ] + mensaje de error de conexión.
UI.7	( )	[] + mensaje de error servicio de datos.
UI.8	(“Albacete”, “”)	[ ] + mensaje no se encontraron resultados
UI.9	(“Galicia”, “”)	[Avia] + mensaje error ausencia de datos

REPORTE FINAL

ANEXO 1. FICHERO JSON UTILIZADO PARA LAS PRUEBAS

Para probar el correcto funcionamiento de la implementación desarrollada para esta historia de usuario se utilizarán los datos incluidos en el fichero JSON, los cuales se encuentran resumidos en la Tabla 1.

Tabla 1 Fichero JSON.

GASOLINERAS		
RÓTULO	PROVINCIA	LOCALIDAD
Cepsa	Cantabria	Santander
Repsol	Cantabria	Santander
Carrefour	Cantabria	Santander
Ballenoil	Cantabria	Guarnizo
Shell	Cantabria	Maliaño
Petronor	Asturias	Gijón
Avia	Coruña (A)	Carballo
Cepsa	Coruña (A)	

Pablo Canales Cobo