**Plan de pruebas US** **500955-Filtrar por tipo de combustible.**

**PRUEBAS DE UI:**

Son las mismas pruebas que las pruebas de aceptación, renombradas como “UI.x” con excepción de que los resultados obtenidos se filtran en base a un fichero JSON llamado **gasolineras\_filtro\_tipo\_test** y se automatizan a través de Espresso, para comprobar que la interfaz de usuario funciona correctamente.

Una tabla resumen del fichero JSON mencionado anteriormente se muestra a continuación:

|  |  |  |
| --- | --- | --- |
| **Rótulo (Identifica un objeto de gasolinera)** | **Precio Gasolina 95 E5** | **Precio Gasóleo A** |
| CEPSA | 1,679 | 1,509 |
| REPSOL | 1,669 | 1,569 |
| PETRONOR | 1,639 | 1,525 |
| PETRONOR V2 | 1,639 | 1,525 |
| REDETRANS | - | 1.299 |
| GALP | 1,639 | - |

TABLA 2: JSON

|  |  |  |
| --- | --- | --- |
| **Rótulo (Identifica un objeto de gasolinera)** | **Precio Gasolina 95 E5** | **Precio Gasóleo A** |
| CEPSA | - | 1,509 |
| REPSOL | - | 1,569 |
| PETRONOR | - | 1,525 |

TABLA 3: JSON PARA UI.7

En los casos definidos a continuación se tiene en cuenta que los únicos tipos definidos dentro de la aplicación son: Gasolina 95 E5 y Gasóleo A. Por ende, solo podremos filtrar por esos dos. Dentro del JSON tenemos 6 gasolineras. En el resultado se retornarían los objetos gasolinera, para identificarlos, lo haremos a través de la marca, ya que hemos definido marcas diferentes.

|  |  |  |
| --- | --- | --- |
| **Identificador** | **Entrada** | **Resultado** |
| UI.1 | Gasolina 95 E5. | { CEPSA, REPSOL, PETRONOR, PETRONOR V2, GALP} y toast con **5** gasolineras. |
| UI.2 | Gasolina 95 E5, Gasóleo A. | { CEPSA, REPSOL, PETRONOR, PETRONOR V2} y toast con **4** gasolineras. |
| UI.3 | UI.3.1: FiltroMarca = PETRONOR && Gasolina 95 E5.  UI.3.2: FiltroMarca: CEPSA && Gasoleo A. | UI.3.1: {PETRONOR, PETRONOR V2}  UI.3.2: {CEPSA} |
| UI.4 | Restaurar orden. | Lista con todas las gasolineras del JSON. |
| UI.5 | **Asumiendo el estado anterior de tener el filtro de Gasolina 95E5.** | { CEPSA, REPSOL, PETRONOR, PETRONOR V2, GALP}. |
| UI.6 |  | { CEPSA, REPSOL, PETRONOR, PETRONOR V2} y toast con **4** gasolineras. |
| UI.7 | Gasolina 95 E5. | Lista vacía y toast con 0 gasolineras. |
| UI.8 | UI.8.1: FiltroPrecioMax = 1.2 && Gasolina 95 E5.  UI.8.2: FiltroMarca = GALP && Gasoleo A.  UI.8.3: FiltroMarca = CEPSA && FiltroPrecioMax = 1.6 && Gasolina 95 E5. | UI.8.X: Lista vacía y toast con 0 gasolineras. |
| UI.9 | TipoCombustibleOrden = Gasolina 95 E5 && Orden = ASC && TipoCombustible = Gasoleo A | { **PETRONOR, PETRONOR V2, GALP**, REPSOL, CEPSA, REDETRANS } Las 3 seleccionadas en negrita, pueden ir ordenadas bajo cualquier combinación entre ellas.  Toast: “Se ha reestablecido el filtro debido a una colisión del tipo con la ordenación” |

TABLA 4: Casos de UI

**PRUEBAS UNITARIAS**

En esta historia de usuario no se modifica nada respecto al acceso y persistencia de datos, por lo tanto, las pruebas unitarias consistirán en pruebas en las clases de dominio, negocio y de presentación.

**Pruebas unitarias de las clases de dominio**

Deberían probarse los métodos de la clase Filter.

* Método setfuelTypes(List<FuelType> ) : IFilter
* Método getFuelTypes(): List<FuelType>
* Método typeFilter( g : Gasolinera ) : Boolean
* Método toFilter( g :List <Gasolinera> ) : List<Gasolinera>
* Método toCopy(): IFilter
* Método clear() : void

Se va a implementar la prueba unitaria del método **toFilter( g :List <Gasolinera> ) : List<Gasolinera>**

|  |  |  |
| --- | --- | --- |
| Identificador | Entrada | Valor esperado |
| UD4.a | List<Gasolinera> = [CEPSA, REPSOL, PETRONOR, PETRONOR V2, REDETRANS, GALP] ; List<FuelTypeEnum> fuelTypes = [Gasolina95E5] | List<Gasolinera> = [ CEPSA, REPSOL, PETRONOR, PETRONOR V2, GALP] |
| UD4.b | List<Gasolinera> = [CEPSA, REPSOL, PETRONOR, PETRONOR V2, REDETRANS, GALP] ; List<FuelTypeEnum> fuelTypes = [GasoleoA] | List<Gasolinera> = [ CEPSA, REPSOL, PETRONOR, PETRONOR V2, REDETRANS] |
| UD4. c | List<Gasolinera> = [CEPSA, REPSOL, PETRONOR, PETRONOR V2, REDETRANS, GALP] ;  List<FuelTypeEnum> fuelTypes = [Gasolina95E5, GasoleoA] | List<Gasolinera> = [ CEPSA, REPSOL, PETRONOR, PETRONOR V2] |
| UD4.d | List<Gasolinera> = [] ;  List<FuelTypeEnum> fuelTypes = [Gasolina95E5] | List<Gasolinera> = []; |

TABLA 5: Unitarias Dominio *método toFilter( g :List <Gasolinera> )*

**Pruebas unitarias de la capa de negocio**

Deben probarse los métodos añadidos a la clase MainPresenter, mediante el uso de mocks de IFilter, IGasolinerasRepository, ICallback e IMainContract#View.

* Método onFiltersClicked(): void
* Método onFiltersPopUpFuelTypesSelected(): void
* Método onFiltersPopUpFuelTypesOneSelected(int index, boolean value): void
* Método onFiltersPopUpFuelTypesAccepted(): void
* Método onFiltersPopUpCancelClicked(): void
* Método onFiltersPopUpAcceptClicked(): void
* Método onFiltersPopUpClearFiltersClicked(): void

Como aclaración, los métodos a probar de onFiltersPopUpCancelClicked(), onFiltersPopUpAcceptClicked() y onFiltersPopUpClearFiltersClicked() y onFiltersClicked(), estarán en los planes de pruebas definidos para las historias de usuario relacionadas con el filtrado, ya que afectan a cada una de las funcionalidades de filtrado implementadas.

Se va a implementar la prueba unitaria del método **onFiltersPopUpClearFiltersClicked(): void**

|  |  |  |
| --- | --- | --- |
| Identificador | Entrada | Valor esperado |
| UB7.a | TempFilter = {Gasolina95E5} | TempFilter = {Gasolina95E5, GasoleoA} |
| UB7.b | TempFilter = {GasoleoA} | TempFilter = {Gasolina95E5, GasoleoA} |

TABLA 6: Unitarias Negocio *método onFiltersPopUpClearFiltersClicked()*

**PRUEBAS DE INTEGRACIÓN**

En lo referente a las pruebas de integración, solo se ha añadido la interfaz IFilter que contiene los métodos necesarios para llevar a cabo el filtrado por tipo. Deberemos probar, por ende, la integración entre el Presenter y la clase de dominio Filter. Se usarán mocks para IMainContract#View, probando así la interacción entre el Presenter y el Filter.

Bajo el JSON definido arriba, podemos prototipar una serie de casos de prueba para este tipo de pruebas.

Se probará el funcionamiento de:

* Método onFiltersClicked(): void
* Método onFiltersPopUpFuelTypesSelected(): void
* Método onFiltersPopUpFuelTypesOneSelected(int index, boolean value): void
* Método onFiltersPopUpFuelTypesAccepted(): void
* Método onFiltersPopUpCancelClicked(): void
* Método onFiltersPopUpAcceptClicked(): void
* Método onFiltersPopUpClearFiltersClicked(): void

Se va a implementar la prueba de integración del **onFiltersPopUpAcceptClicked(): void**

|  |  |  |
| --- | --- | --- |
| Identificador | Entrada | Valor esperado |
| IB6.a | TempFilter = {Gasolina95E5} | Filter = {Gasolina95E5}  TempFilter = null  **Se muestran las estaciones filtradas que deberían mostrarse (se llama desde la vista y filtros a los métodos necesarios).**  {CEPSA, REPSOL, PETRONOR, PETRONOR V2, GALP} |
| IB6.b | TempFilter = {GasoleoA} | Filter = {GasoleoA}  TempFilter = null  **Se muestran las estaciones filtradas que deberían mostrarse (se llama desde la vista y filtros a los métodos necesarios).**  {CEPSA, REPSOL, PETRONOR, PETRONOR V2, GALP} |
| IB6.c | TempFilter = {Gasolina95E5, GasoleoA } | Filter = {Gasolina95E5, GasoleoA}  TempFilter = null  **Se muestran las estaciones filtradas que deberían mostrarse (se llama desde la vista y filtros a los métodos necesarios).**  {CEPSA, REPSOL, PETRONOR, PETRONOR V2, GALP} |

TABLA 7: Unitarias Negocio *método onFiltersPopUpAcceptClicked()*

**REPORTE FINAL:**

Comentarios respecto a la codificación y ejecución de dos pruebas, una unitaria y otra de integración.

**Prueba unitaria**

* En la clase **Filter**, el método **toFilter( g :List <Gasolinera> ) : List<Gasolinera>**

La ejecución de las pruebas descritas en este documento ha pasado en su conjunto sin detectar fallos para la implementación realizada.

**Prueba de integración**

* En la clase **MainPresenter**, el método **onFiltersPopUpAcceptClicked(): void**

La ejecución de las pruebas descritas en este documento ha pasado en su conjunto sin detectar fallos para la implementación realizada.

**Autoría**

El plan de pruebas ha sido realizado por:

* Fernández Mancebo, Lucía

El apartado de reporte final, la codificación y ejecución de las pruebas ha sido realizada por:

* Del Río Nieto, Adrián