PLAN DE PRUEBAS#506510-Busacar a partir de coordenadas

1. Pruebas de aceptación.

En base a los criterios de aceptación negociados con el *product owner*, se identifican los siguientes escenarios:

A1. CA : Caso de éxito: Se muestra una lista de gasolineras filtradas bajo las coordenadas y radio introducidos.

1. El usuario introduce dos coordenadas válidas (latitud y longitud con precisión de cuatro cifras).
2. El usuario ajusta el deslizador de radio en un valor entre 0 y 100km.
3. El usuario pulsa el botón “Aceptar”.
4. La aplicación muestra una lista de gasolineras encontrada dentro del radio especificado.
5. La aplicación muestra un *toast* con el número de gasolineras encontradas.

A2. CA : Caso de éxito: Se muestra una lista vacía de gasolineras bajo las coordenadas y radio introducidos.

1. El usuario introduce dos coordenadas válidas (latitud y longitud con precisión de cuatro cifras).
2. El usuario ajusta el deslizador de radio en un valor entre 0 y 100km.
3. El usuario pulsa el botón “Aceptar”.
4. La aplicación muestra una lista vacía ya que no hay gasolineras dentro del radio especificado.
5. La aplicación muestra un *toast* indicando 0 gasolineras cargadas.

A3. CA : Caso de éxito: Búsqueda exitosa con filtro previo aplicado y radio y coordenadas validas.

1. El usuario aplica uno o varios filtros a la lista de gasolineras inicial (ordenación, filtro por localidad, etc.).
2. El usuario introduce dos coordenadas válidas (latitud y longitud con precisión de cuatro cifras).
3. El usuario ajusta el deslizador de radio en un valor entre 0 y 100km.
4. El usuario pulsa el botón “Aceptar”.
5. La aplicación muestra una lista de gasolineras encontradas dentro del radio especificado y que siguen los filtros previos aplicados.
6. La aplicación muestra un toast con el numero d gasolineras encontradas.

A4. CA : Caso de error: Búsqueda con coordenadas incompletas.

1. El usuario deja el campo de coordenadas vacío.
2. El usuario ajusta el deslizador de radio en un valor entre 1 y 100km.
3. El usuario pulsa el botón “Aceptar”.
4. La aplicación no realiza ninguna búsqueda y mantiene el estado de la lista previa al intento de búsqueda invalido.

A5. CA : Caso de error: Búsqueda con coordenadas incorrectas.

1. El usuario introduce coordenadas fuera del rango permitido.
2. El usuario ajusta el deslizador de radio en un valor entre 0 y 100km.
3. El usuario pulsa el botón “Aceptar”.
4. La aplicación muestra una lista vacía ya que no hay gasolineras dentro del radio para las coordenadas especificadas.
5. La aplicación muestra un *toast* indicando 0 gasolineras cargadas.

A6. CA : Caso de error: Error pérdida de conexión.

1. El usuario introduce dos coordenadas válidas (latitud y longitud con precisión de cuatro cifras).
2. El usuario ajusta el deslizador en un valor entre 0 y 100km.
3. El usuario pulsa el botón “Aceptar”.
4. La aplicación intenta realizar la búsqueda, pero no logra obtener los datos debido a u problema de conectividad.
5. La aplicación muestra un mensaje de error informando de la perdida de conexión.
6. La aplicación mantiene el estado de la lista previa al intento de búsqueda erróneo.

A7. CA : Error en el acceso al servicio de datos.

1. El usuario introduce dos coordenadas válidas (latitud y longitud con precisión de cuatro cifras).
2. El usuario ajusta el deslizador en un valor entre 0 y 100km.
3. El usuario pulsa el botón “Aceptar”.
4. La aplicación intenta acceder al servicio de datos para realizar la búsqueda, pero el servidor de datos de gasolineras no responde o esta inactivo.
5. La aplicación muestra un mensaje de error del error al acceder al servicio de datos.
6. La aplicación mantiene el estado de la lista previa al intento de búsqueda erróneo.

2. Pruebas unitarias.

A continuación, se listan los métodos modificados o creados que son candidatos a ser probados:

- `searchWithCoordinates(Double longitud, Double latitud, int distancia): void` *(MainPresenter.java)*
- `estaEnCoordenadas(Double longitudSelec, Double latitudSelec, int distancia, Double longitudGasolinera, Double latitudGasolinera): Boolean` *(MainPresenter.java)*

Método: searchWithCoordinates

void searchWithCoordinates(Double longitud, Double latitud, int distancia)

Para probar este método se codificarán los siguientes casos (estado inicial, Anexo):

A. Búsqueda con distancia 0.
B. Búsqueda con distancia 100.
C. Búsqueda sin estaciones en el rango.
D. Búsqueda con distancia media (múltiples fuera y dentro del rango).
E. Búsqueda con coordenadas 0.
F. Búsqueda con coordenadas límite de ser validas.
G. Búsqueda con estación en el limite de rango de distancia.
H. Coordenadas fuera del rango valido.

Las coordenadas no pueden ser nulas ni mayores de dos cifras enteras y cuatro decimales. *(Restricción MainView.java)*

Tabla 1. Casos de prueba unitarios para el método searchWithCoordinates de la clase MainPresenter

Identificador	Entrada	Valor esperado
UT.1A	(-4.03, 43.20, 0)	No se devuelven estaciones
UT.1B	(-4.03, 43.20, 100)	[RESPOL, CARREFOUR, BALLENOIL, SHELL, PETRONOR, AVIA, CEPSA]
UT.1C	(-4.03, 43.20, 5)	No se devuelven estaciones
UT.1D	(-4.03, 43.20, 10)	[RESPOL, CARREFOUR, BALLENOIL, SHELL]
UT.1E	(0.00, 0.00 , 50)	No se devuelven estaciones
UT.1F	(-99.99, 89.99, 100)	No se devuelven estaciones
UT.1G	(-4.03, 43.20, 29)	[RESPOL, CARREFOUR, BALLENOIL]
UT.1H	(-99.99, 99.99, 50)	No se devuelven estaciones

Método: estaEnCoordenadas

boolean estaEnCoordenadas(Double longitudSelec, Double latitudSelec, int distancia, Double longitudGasolinera, Double latitudGasolinera)

Para probar este método se codificarán los siguientes casos:

A. Búsqueda con coordinadas coincidentes y distancia 0.
B. Búsqueda con coordenadas coincidentes y distancia distinta de 0.
C. Búsqueda con gasolinera muy cercana a coordenadas.

D. Búsqueda con gasolinera fuera de rango de coordenadas.
E. Búsqueda con gasolinera en el límite de rango.
F. Búsqueda con coordenadas inválidas.

Las coordenadas no pueden ser nulas ni mayores de dos cifras enteras y cuatro decimales. *(Restricción MainView.java)*

Tabla 2 Casos de prueba unitarios para el método estaEnCoordenadas de la clase MainPresenter

Identificador	Entrada	Valor esperado
UT.2A	(10.0, 20.0, 0, 10.0, 20.0)	true
UT.2B	(10.0, 20.0, 10, 10.0, 20.0)	true
UT.2C	(10.0, 20.0, 5, 10.002, 20.002)	true
UT.2D	(10.0, 20.0, 5, 10.03, 20.04)	false
UT.2E	(10.0, 20.0, 77, 10.5, 20.5)	true
UT.2F	(-99.99, 99.99, 50)	false

3. Pruebas de interfaz.

Las pruebas de interfaz a realizar se alinean con las pruebas de aceptación. A continuación, se detallan los casos de prueba a codificar (estado inicial, Anexo).

Tabla 3 Casos de prueba de interfaz

Identificador	Entrada	Valor esperado
AT.1A	Coordenadas (-4.03, 43.20) Deslizador de distancia (100)	[RESPOL, CARREFOUR, BALLENOIL, SHELL, PETRONOR, AVIA, CEPSA], toast 7 gasolineras
AT.1B	Coordenadas (-4.03, 43.20) Deslizador de distancia (5)	[], toast 0 gasolineras
AT.1C	Filtro por Municipio "Santander" Coordenadas (-4.03, 43.20) Deslizador de distancia (10)	[RESPOL, CARREFOUR], toast 4 gasolineras
AT.1D	Coordenadas (,) Deslizador de distancia (50)	[RESPOL, CARREFOUR, BALLENOIL, SHELL, PETRONOR, AVIA, CEPSA]
AT.1E	Coordenadas (200, 200) Deslizador de distancia (50)	[], toast 0 gasolineras
AT.1F	Coordenadas (-4.03, 43.20) Deslizador de distancia (100)	Mensaje error, perdida de conexión
AT.1G	Coordenadas (-4.03, 43.20) Deslizador de distancia (100)	Mensaje de error, fallo al acceder al servicio de datos

4. Anexo.

Tabla 4 Estado inicial.

Gasolinera	Longitud	Latitud	Municipio
REPSOL	-3.86	43.40	Santander
CARREFOUR	-4.07	43.32	Santander
BALLENOIL	-3.76	43.36	Guarnizo
SHELL	-4.60	43.17	Maliaño
PETRONOR	-3.03	43.33	Gijón
AVIA	-3.09	43.36	Carballo
CEPSA	-3.33	43.11	Ferrol

5. Reporte final.

Autor del plan de pruebas: Javier Arce del Campo.
Autor de las pruebas unitarias (diseño, codificación y ejecución): Javier Arce del Campo.
Autor de las pruebas de interfaz (diseño, codificación y ejecución): Javier Arce del Campo.

Se ha llevado a cabo el proceso de pruebas según lo esperado. Todas las pruebas descritas en el plan se han ejecutado, con resultados satisfactorios que confirman la correcta implementación de los métodos.

Los métodos *searchWithCoordinates* y *estaEnCoordenadas* fueron probados con los casos descritos, verificando su funcionalidad en diversos escenarios. Los resultados obtenidos coincidieron con los esperados.

Se han ejecutado pruebas de interfaz alineadas con las pruebas de aceptación. Solo se codificó y ejecutó el primer caso de prueba, obteniendo los resultados esperados.

> P. D.: Test de Aimar pendientes de revisar.