## Plan de pruebas de la aplicación UCPark

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

* **Pruebas unitarias**. Se utilizará la técnica de prueba de métodos, usando técnicas de caja negra (partición equivalente y AVL) para la definición de los casos de prueba de cada método de cada clase o componente. Será necesaria la utilización de JUnit, Mockito y FEST.
* **Pruebas de integración**. La estrategia para la definición del orden de las pruebas de integración será jerárquica. Se probará:
  + La integración entre la capa de negocio y la de persistencia. En este caso, para la definición de los casos de prueba se utilizarán técnica de métodos y caja negra y se utilizará JUnit.
  + La integración entre las tres capas. En este caso, para la definición de los casos de prueba se utilizarán técnica de casos de uso y se utilizarán Junit y FEST.A continuación, se muestra una especificación detallada de los casos de prueba a aplicar en cada nivel mencionado anteriormente.
* **Pruebas de aceptación**. Las pruebas de aceptación se definirán siguiendo una estrategia basada en casos de uso y se ejecutarán de forma manual.

**PRUEBAS DE ACEPTACIÓN**

En base a los casos de uso se identifican los siguientes escenarios:

A1. CU: Registrarse

1. Registro válido (nuevo usuario)
2. Registro no válido (usuario ya existe)
3. Registro no válido (usuario nulo)

A2. CU: Registrar vehículo

1. Registro válido (vehículo no añadido aún)
2. Registro no válido (vehículo ya existente)
3. Registro no válido (vehículo nulo)

A3. CU: Eliminar vehículo

1. Eliminación válida
2. Eliminación no válida (el usuario no tiene vehículos)

A4. CU: Consultar denuncias

1. Consulta válida

A5. CU: Consultar estacionamientos en vigor

1. Consulta válida

A6. CU: Consultar histórico de estacionamientos

1. Consulta válida

A7. CU: Nuevo estacionamiento

1. Registro válido (el vehículo no tenía ningún estacionamiento activo)
2. Registro no válido (el vehículo ya tiene un estacionamiento activo)
3. Registro no válido (minutos mayores que 120)
4. Registro no válido (minutos menores o iguales a 0)
5. Registro no válido (error en cobro)

A8. CU: Ampliar tiempo de estacionamiento

1. Ampliación válida
2. Ampliación no válida (total de minutos excede el máximo global)
3. Ampliación no válida (minutos negativos)
4. Ampliación no válida (error en el cobro)

A9. CU: Finalizar estacionamiento

1. Finalización válida

A10. CU: Comprobar estacionamiento

1. Comprobación válida (el vehículo tiene un estacionamiento activo)
2. Comprobación válida (el vehículo no tiene un estacionamiento activo)
3. Comprobación no válida (el vehículo no existe en el sistema)

A11. CU: Denunciar estacionamiento

1. Denuncia válida (el vehículo no tiene un estacionamiento activo)
2. Denuncia no válida (el vehículo tiene un estacionamiento activo)
3. Denuncia no válida (el vehículo no existe en el sistema)

Los casos de prueba definidos solo en el caso del escenario *Nuevo estacionamiento* son los que se muestran en la Tabla 1, partiendo de una base de datos vacía.

Tabla 1. Casos de prueba de aceptación

|  |  |  |
| --- | --- | --- |
| **Identificador** | **Entrada** | **Resultado** |
| A7.a | <”1111-AAA”, “Peugeot”, “205”>, 30 | Registro válido (el vehículo no tenía ningún estacionamiento activo) |
| A7.b | <”1111-AAA”, “Peugeot”, “205”>, 60 | Registro no válido (el vehículo ya tiene un estacionamiento activo) |
| A7.c | <”2222-BBB”, “Ford”, “Focus”>, 140 | Registro no válido (minutos mayores que 120) |
| A7.d | <”3333-CCC”, “Citröen”, “Xsara”>, -40 | Registro no válido (minutos menores o iguales a 0) |
| A7.e | <”4444-DDD”, “Mini”, “Cooper”>, 90 | Registro no válido (error en cobro) |

**PRUEBAS DE INTEGRACIÓN**

El orden de las pruebas y los casos de prueba a realizar serían los siguientes:

1. GestionUsuarios con UsuariosDAO.
2. GestionEstacionamientos con EstacionamientosDAO y VehiculosDAO. Se usarían los mismos casos de prueba definidos como EGE.x en la sección de pruebas unitarias, aquí renombrados como IGE.x.
3. GestionVehiculos con UsuariosDAO con VehiculosDAO.
4. GestionDenuncias con VehiculosDAO y DenunciasDAO.
5. VistaUsuarioAnonimo con GestionUsuarios.
6. VistaUsuarioLogeado con GestionUsuarios, GestionEstacionamientos, GestionVehiculos y GestionDenuncias.
7. VistaAgente con GestionEstacionamientos y GestionDenuncias.

Las pruebas de estos tres últimos escenarios coincidirían con sus hipotéticas correspondientes pruebas de aceptación, aunque en este caso se automatizarían utilizando la librería FEST. Sin embargo, sólo se definirán las pruebas para el apartado 2.

**PRUEBAS UNITARIAS**

**Pruebas unitarias de la capa de persistencia**

Se aplica prueba de métodos, siendo los casos de prueba definidos para cada método los que se exponen a continuación. Los casos expuestos para cada método suponen como punto de partida una base de datos con los siguientes valores:

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Minutos** | **Fecha de inicio** | **Vehículo** |
| 1 | 60 | 01/04/2022, 11:58 | <”1111-AAA”, “Peugeot”, “205”> |
| 2 | 85 | 24/04/2022, 00:49 | <”1111-AAA”, “Peugeot”, “205”> |
| 3 | 90 | 04/07/2022, 15:51 | <”1111-AAA”, “Peugeot”, “205”> |
| 4 | 20 | 06/07/2022, 14:03 | <”2222-BBB”, “Ford”, “Focus”> |
| 5 | 120 | 19/07/2022, 14:10 | <”2222-BBB”, “Ford”, “Focus”> |
| 6 | 10 | 05/09/2022, 14:24 | <”3333-CCC”, “Citröen”, “Xsara”> |
| 7 | 24 | 03/10/2022, 08:35 | <”3333-CCC”, “Citröen”, “Xsara”> |
| 8 | 5 | 31/10/2022, 16:37 | <”3333-CCC”, “Citröen”, “Xsara”> |
| 9 | 45 | 24/12/2022, 19:26 | <”4444-DDD”, “Mini”, “Cooper”> |
| 10 | 33 | 01/01/2023, 17:32 | <”4444-DDD”, “Mini”, “Cooper”> |

* Método getParkingList()

|  |  |  |
| --- | --- | --- |
| **Identificador** | **Entrada** | **Valor esperado** |
| **UCD.1a** |  | Listado con diez estacionamientos |
| **UCD.1b** |  | Lista vacía (sería necesario probar con una base de datos vacía) |

* Método getParking(long)

|  |  |  |
| --- | --- | --- |
| **Identificador** | **Entrada** | **Valor esperado** |
| **UCD.2a** | 1 (estacionamiento existe) | <1, 01/04/2022 11:58, <”1111-AAA”, “Peugeot”, “205”>> |
| **UCD.2b** | 11 (estacionamiento no existe) | null |

* Método addParking(Parking)

|  |  |  |
| --- | --- | --- |
| **Identificador** | **Entrada** | **Valor esperado** |
| **UCD.3a** | < id = 1, 60, 02/04/2022 11:58, <”1111-AAA”, “Peugeot”, “205”>> (estacionamiento no existe) | true |
| **UCD.3b** | < id = 1, 60, 02/04/2022 11:58, <”1111-AAA”, “Peugeot”, “205”>> (estacionamiento ya existe) | false |

* Método modifyParking(Parking)

|  |  |  |
| --- | --- | --- |
| **Identificador** | **Entrada** | **Valor esperado** |
| **UCD.4a** | < id = 1, 85, 01/04/2022 11:58, <”1111-AAA”, “Peugeot”, “205”>> (estacionamiento existe) | < id = 1, 85, 02/04/2022 11:58, <”1111-AAA”, “Peugeot”, “205”>> |
| **UCD.4b** | <id = 11, 85, 01/04/2022 11:58, <”1111-AAA”, “Peugeot”, “205”>> (estacionamiento no existe) | null |

* Método deleteParking(long)

|  |  |  |
| --- | --- | --- |
| **Identificador** | **Entrada** | **Valor esperado** |
| **UCD.5a** | 1 (estacionamiento existe) | true |
| **UCD.5b** | 11 (estacionamiento no existe) | false |

**Pruebas unitarias de la capa de negocio**

Para poder llevar a cabo estas pruebas, será necesario el uso de objetos Mock para la interfaz IUsuariosDAO. Se aplica prueba de métodos, siendo los casos de prueba definidos para cada método los siguientes:

* Método consultParking: conceptualmente se trata de los mismos casos identificados para el caso de uso 10.

|  |  |  |
| --- | --- | --- |
| **Identificador** | **Entrada** | **Valor esperado** |
| **UGE.1a** | “1111-AAA” | < id = 1, 85, 02/04/2022 11:58, <”1111-AAA”, “Peugeot”, “205”>> |
| **UGE.1b** | “2222-BBB” | null |
| **UGE.1c** | “5555-EEE” | OperacionNoValida |

* Método registerParking: conceptualmente se trata de los mismos casos identificados para el caso de uso 7.

|  |  |  |
| --- | --- | --- |
| **Identificador** | **Entrada** | **Valor esperado** |
| **UGE.2a** | <”1111-AAA”, “Peugeot”, “205”>, 30 | El vehículo ahora tiene un estacionamiento registrado |
| **UGE.2b** | <”1111-AAA”, “Peugeot”, “205”>, 60 | OperacionNoValida |
| **UGE.2c** | <”2222-BBB”, “Ford”, “Focus”>, 140 | OperacionNoValida |
| **UGE.2d** | <”2222-BBB”, “Ford”, “Focus”>, -50 | OperacionNoValida |
| **UGE.2e** | <”2222-BBB”, “Ford”, “Focus”>, 90 | OperacionNoValida |

* Método extendParkingTime: conceptualmente se trata de los mismos casos identificados para el caso de uso 8.

|  |  |  |
| --- | --- | --- |
| **Identificador** | **Entrada** | **Valor esperado** |
| **UGE.3a** | <”1111-AAA”, “Peugeot”, “205”>, 30 | < id = 1, 60, 01/04/2022 11:58, <”1111-AAA”, “Peugeot”, “205”>> |
| **UGE.3b** | <”1111-AAA”, “Peugeot”, “205”>, 91 | OperacionNoValida |
| **UGE.3c** | <”1111-AAA”, “Peugeot”, “205”>, -20 | OperacionNoValida |
| **UGE.3d** | <”1111-AAA”, “Peugeot”, “205”>, 30 | OperacionNoValida |

A9. CU: Finalizar estacionamiento

1. Finalización válida

* Método finishParking: conceptualmente se trata de los mismos casos identificados para el caso de uso 9.

|  |  |  |
| --- | --- | --- |
| **Identificador** | **Entrada** | **Valor esperado** |
| **UGE.4a** | 1 | El vehículo ya no tiene un estacionamiento activo |

*Patricia López Martínez y Héctor G. Iglesias*