

Plan de Pruebas Caso de Uso "Nuevo Estacionamiento"

Este documento define las especificaciones de pruebas para el caso de uso "Nuevo Estacionamiento" del sistema UCPark, abarcando pruebas de aceptación y pruebas unitarias seleccionadas.

1. Especificación de Pruebas de Aceptación

Caso de Uso: Nuevo Estacionamiento **Actor:** Usuario Registrado **Precondiciones:** El usuario debe estar registrado y tener al menos un vehículo asociado.

ID Prueba	Escenario	Pasos	Datos de Entrada	Resultado Esperado
ACP-01	Estacionamiento Exitoso	1. El usuario selecciona la opción "Nuevo Estacionamiento". 2. El sistema solicita seleccionar un vehículo. 3. El usuario selecciona un vehículo válido. 4. El sistema registra el inicio del estacionamiento.	Usuario: usuario@test.com Vehículo: ABC-1234	- Se crea un registro de estacionamiento. - El sistema muestra confirmación de inicio. - La hora de inicio es la actual.
ACP-02	Vehículo no seleccionado	1. El usuario intenta iniciar estacionamiento sin seleccionar vehículo.	Usuario: usuario@test.com Vehículo: null	- El sistema muestra un mensaje de error indicando que debe seleccionar un vehículo.
ACP-03	Usuario no válido (Sesión expirada)	1. El usuario intenta iniciar estacionamiento con una sesión inválida.	Usuario: null Vehículo: ABC-1234	- El sistema redirige al login o muestra error de autenticación.
ACP-04	Estacionamiento Duplicado (Vehículo Activo)		Usuario: usuario@test.com Vehículo: ABC-1234 (ya activo)	- El sistema muestra un mensaje de error. - El mensaje indica que el vehículo ya tiene un estacionamiento activo. - No se crea un nuevo registro.
ACP-05	Vehículo No Pertenece al Usuario		Usuario: usuario@test.com Vehículo: XYZ-789 (de otro usuario)	- El sistema rechaza la operación. - Se muestra error de autorización/validación. - No se crea registro de estacionamiento.
ACP-06	Usuario No Tiene Vehículos Registrados		Usuario: usuario-sin-vehiculos@test.com Vehículo: N/A	- El sistema muestra mensaje informativo indicando que debe registrar un vehículo primero.

ID Prueba	Escenario	Pasos	Datos de Entrada	Resultado Esperado
				- Opcionalmente, redirige a registro de vehículos.

2. Especificación de Pruebas Unitarias

A. Capa de Negocio

Componente:

CarManagement **Método a Probar:**

newParking(User user, Vehicle vehicle)

Casos de Prueba:

ID	Descripción	Entradas	Comportamiento Esperado (Asserts)
UT-BUS-01	Inicialización Correcta de Estacionamiento	<code>user</code> : Instancia válida <code>vehicle</code> : Instancia válida (<code>plate</code> : "XYZ-999")	1. Se debe invocar <code>vehicleDao.createParking()</code> 2. El objeto Parking pasado al DAO debe tener: - <code>vehicle</code> igual al vehículo de entrada. - <code>minutes</code> igual a <code>0</code> . - <code>amount</code> igual a <code>0.0</code> . - <code>startTime</code> no nulo (aproximadamente <code>now</code>) 3. El método debe retornar <code>true</code> .
UT-BUS-02	Fallo por Entradas Nulas	<code>user</code> : <code>null</code> O <code>vehicle</code> : <code>null</code>	1. El método debe retornar <code>false</code> . 2. No se debe invocar <code>vehicleDao.createParking()</code>
UT-BUS-03	Fallo en Persistencia DAO	<code>user</code> : Válido <code>vehicle</code> : Válido <code>vehicleDao.createParking()</code> : retorna <code>null</code> o lanza excepción (mock)	1. El método debe retornar <code>false</code> . 2. Verificar que se maneje adecuadamente el error en el DAO. 3. No se debe propagar excepción no controlada
UT-BUS-04	Verificación de Timestamp Correcto	<code>user</code> : Válido <code>vehicle</code> : Válido	1. Capturar el objeto Parking antes de <code>createParking()</code> 2. Verificar: <code>assertNotNull(parking.getStartTime())</code> 3. Verificar: <code>parking.getStartTime()</code> esté dentro de ± 1 segundo del momento actual.
UT-BUS-05	Validación de Vehículo Ya Estacionado	<code>vehicle</code> : Vehículo con estacionamiento activo (mock)	1. El método debe verificar si existe estacionamiento activo. 2. Si existe, retornar <code>false</code> . 3. No invocar <code>vehicleDao.createParking()</code>

B. Capa DAO

Componente:

VehicleDao (Anteriormente `ParkingDao` , refactorizado) **Método a Probar:**

createParking(Parking parking)

Casos de Prueba:

ID	Descripción	Entradas	Comportamiento Esperado (Asserts)
UT-DAO-01	Persistencia Exitosa	<code>parking</code> : Objeto Parking válido (con vehículo y hora definidos)	1. Se debe invocar <code>EntityManager.persist(parking)</code> . 2. El método debe retornar el mismo objeto Parking persistido.
UT-DAO-02	Persistencia Fallida (Duplicado/Error)	<code>parking</code> : Objeto Parking que viola restricciones (ej. ID existente si aplica)	1. <code>EntityManager.persist</code> lanza excepción (simulado en mock o prueba de integración). 2. Verificar manejo de excepción si existe, o fallo esperado.
UT-DAO-03	Verificar Transaccionalidad	<code>parking</code> : Válido Simular fallo después de <code>persist()</code> (mock <code>EntityManager</code>)	1. Verificar que si <code>persist()</code> falla, no se commitea la transacción. 2. Verificar rollback automático o manejo de excepción.
UT-DAO-04	Parking con Datos Nulos Parciales	<code>parking</code> : Objeto con <code>vehicle = null</code> pero otros campos válidos	1. Verificar que <code>persist()</code> lance excepción o retorne error. 2. O bien, validar que exista pre-validación antes de persistir.
UT-DAO-05	Verificar ID Autogenerado	<code>parking</code> : Objeto válido sin ID asignado	1. Después de <code>persist()</code> , verificar que <code>parking.getId()</code> no sea null. 2. Verificar que el ID sea un valor válido autogenerado por la BD.