Documento de Pruebas - Tarea Integradora 1

# Escenarios y Pruebas

Se describen a continuación los escenarios de prueba implementados para validar las funcionalidades del sistema de gestión de movilidad.

## Funcionalidad: Agregar Conductor

Escenario: Ingresar conductor con datos válidos

Resultado esperado: Conductor agregado exitosamente.

## Funcionalidad: Agregar Conductor

Escenario: Ingresar conductor con nombre vacío

Resultado esperado: Debe lanzar IllegalArgumentException.

## Funcionalidad: Buscar Conductor

Escenario: Buscar conductor existente por nombre

Resultado esperado: Debe retornar el objeto conductor.

## Funcionalidad: Buscar Conductor

Escenario: Buscar conductor no existente

Resultado esperado: Debe lanzar ConductorNoEncontradoException.

## Funcionalidad: Agregar Incidente

Escenario: Ingresar incidente con datos válidos

Resultado esperado: Incidente agregado correctamente.

## Funcionalidad: Agregar Incidente

Escenario: Ingresar incidente con descripción vacía

Resultado esperado: Debe lanzar IllegalArgumentException.

## Funcionalidad: Buscar Incidente

Escenario: Buscar incidente por ID existente

Resultado esperado: Debe retornar el incidente.

## Funcionalidad: Buscar Incidente

Escenario: Buscar incidente con ID no existente

Resultado esperado: Debe lanzar IncidenteNoEncontradoException.

## Funcionalidad: Ordenar Rutas

Escenario: Ordenar rutas por distancia

Resultado esperado: Rutas ordenadas correctamente.

## Funcionalidad: Ordenar Incidentes

Escenario: Ordenar incidentes por fecha

Resultado esperado: Incidentes ordenados del más reciente al más antiguo.

## Funcionalidad: Guardar y Cargar JSON

Escenario: Guardar y luego cargar datos

Resultado esperado: Los datos deben persistir correctamente.

# Indicadores de Calidad

Se presentan los indicadores calculados para las dos iteraciones de desarrollo del sistema:

Iteración 1 : <commit-sha>

Densidad de errores-fallos = 0.2012

Confiabilidad = 0.7988

Completitud = 2.32

Iteración 2 : <commit-sha>

Densidad de errores-fallos = 0.4043

Confiabilidad = 0.5957

Completitud = 3.01

Fórmulas utilizadas:

Densidad de errores-fallos = total de fallos / total de pruebas

Confiabilidad = 1 - densidad de fallos

Completitud = casos de prueba / total funcionalidades

# Requerimientos del Sistema

## Requerimientos Funcionales

- RF1: El sistema debe permitir registrar rutas con su ID, distancia y tiempo estimado.

- RF2: El sistema debe permitir registrar incidentes con tipo, descripción, ubicación y estado.

- RF3: El sistema debe registrar conductores con nombre, ID, estado y vehículo.

- RF4: El sistema debe permitir buscar conductores por nombre.

- RF5: El sistema debe permitir buscar incidentes por ID.

- RF6: El sistema debe ordenar rutas por distancia o tiempo.

- RF7: El sistema debe ordenar incidentes por fecha/hora.

- RF8: El sistema debe generar reportes simples en consola.

- RF9: El sistema debe cargar y guardar datos en archivos JSON.

- RF10: El sistema debe manejar errores mediante excepciones personalizadas.

## Requerimientos No Funcionales

- RNF1: El sistema debe ejecutarse en consola, de forma clara y ordenada.

- RNF2: El código debe estar estructurado utilizando el paradigma de programación orientada a objetos.

- RNF3: Se deben utilizar buenas prácticas de programación, incluyendo nombres claros y comentado cuando sea necesario.

- RNF4: El sistema debe ser persistente mediante archivos JSON.

- RNF5: Se debe incluir un conjunto de pruebas unitarias que garanticen el correcto funcionamiento de las funcionalidades.

- RNF6: El proyecto debe estar versionado utilizando Git y subirlo a un repositorio remoto.