Plan de Pruebas del Proyecto Titanic

1. Introducción

El presente plan de pruebas describe la estrategia y alcance de las pruebas unitarias desarrolladas para el proyecto **Titanic**, cuyo objetivo es validar la correcta funcionalidad de los módulos relacionados con la simulación del sistema de emergencia y rescate de botes salvavidas.

---

2. Objetivos

- Comprobar la **correcta generación y distribución de pasajeros** en los botes salvavidas.
- Verificar la **validez del procesamiento y transformación de información** durante el rescate.
- Validar la **lógica de ordenación, exportación y manejo de resultados** en los informes generados.
- Garantizar que el sistema maneje **entradas válidas e inválidas** correctamente, mostrando los errores adecuados cuando sea necesario.
- Confirmar que los archivos generados durante el proceso cumplen con los criterios de integridad (no vacíos y con formato correcto).

---

3. Alcance

El plan abarca las pruebas unitarias de los componentes del sistema responsables de:

- La creación y gestión de botes salvavidas.
- El procesamiento y verificación de los datos de los pasajeros.
- La generación y exportación de informes de rescate.

No se incluyen pruebas de integración ni pruebas de interfaz gráfica. Todas las pruebas están orientadas a la lógica interna del sistema.

---

4. Entorno de Pruebas

- **Lenguaje de desarrollo:** Java 17 o superior
- **Herramienta de pruebas:** Sistema automatizado basado en JUnit 5
- **Entorno recomendado:** visual estudio code
- **Dependencias externas:** Ninguna (todas las pruebas se ejecutan en memoria, salvo las que generan informes en el sistema de archivos).
- **Carpeta de salida:** `/informes/` (creada automáticamente durante la ejecución de las pruebas).

---

5. Casos de Prueba

5.1. Módulo de generación de datos

Objetivo: comprobar que los datos generados sobre los botes y los pasajeros son coherentes y cumplen las reglas establecidas.

Caso	Descripción	Resultado Esperado
1	Se comprueba que la distribución de pasajeros este dentro de los límites permitidos.	El número total de pasajeros está dentro del rango esperado.
2	Se validan los identificadores asignados a cada bote.	Los identificadores se mantienen correctos y sin duplicados.
3	Se comprueba que los datos generados incluyan toda la información necesaria.	Todos los campos requeridos están presentes y correctamente asignados.

5.2. Módulo de procesamiento de información

Objetivo: asegurar que los datos del sistema se procesan correctamente y que los posibles errores son gestionados de forma controlada.

Caso	Descripción	Resultado Esperado
1	Se transforma la información recibida en una estructura válida para el análisis.	Los datos se convierten correctamente sin pérdida de información.
2	Comprueba que se gestiona adecuadamente entradas vacías, incompletas o con formato incorrecto.	El sistema muestra errores controlados sin interrumpir el proceso.
3	Se comprueba la consistencia del total de pasajeros procesados.	La suma coincide con los valores esperados.

5.3. Módulo de informes y resultados

Objetivo: verificar que el sistema genera los informes correctamente y que los datos exportados son consistentes.

Caso	Descripción	Resultado Esperado
1	Se comprueba que los resultados se organizan de forma ordenada.	Los informes muestran los datos en el orden esperado.
2	Se genera un informe incluso si no hay datos de entrada.	Se crea un archivo válido y no vacío.
3	Se valida la respuesta del sistema ante la falta de datos.	El sistema notifica adecuadamente la ausencia de información.

6. Ejecución de las Pruebas

Las pruebas se ejecutan mediante el comando:

```
mvn test
```

o directamente desde el IDE mediante el runner de JUnit.
Cada clase de prueba se ejecuta de forma independiente, y los métodos están diseñados para no depender de resultados previos (principio de independencia de pruebas unitarias).

7. Conclusiones

El conjunto de pruebas garantiza una cobertura amplia de los componentes principales del sistema Titanic. Se validan tanto los **escenarios normales de uso** como las **condiciones de error**, lo que asegura que los módulos reaccionen correctamente ante entradas válidas, nulas o erróneas.
El resultado esperado es un sistema de rescate robusto, con generación coherente de datos, validaciones consistentes y exportación fiable de informes.