

**Project Name: GearMate**

## **Test Case (Happy Case)**

**Test Case ID:** SortingGear

**Test Priority (Low/Medium/High):** Low

**Module Name:** Gear List Screen

**Test Title:** Verify sorting gear list successfully

**Description:** Ensure gear list is correctly sorted based on selected criteria.

**Test Designed by:** Nattapat, Phonlapat

**Test Designed date:** 24 Oct 2025

**Test Executed by:** Nattapat, Phonlapat

**Test Execution date:** 24 Oct 2025

**Pre-conditions:** User has access to the “Gear List” page

**Dependencies:**

- Sorting menu must be implemented
- Gear list data must exist in the database
- Sorting rules must be defined

Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	Select “Sort by Name”	Sort_by = Name	Gear list sorted Alphabetically	Gear list sorted Alphabetically	Pass	
2	Select “Sort by Type”	Sort_by = Type	Gear list sorted by gear type	Gear list sorted by gear type	Pass	
3	Select “Sort by Maintenance Date”	Sort_by = Maintenance Date	Gear list sorted earliest to latest date	Gear list sorted earliest to latest date	Pass	

**Post-conditions:** Gear list is displayed in sorted order. No data is modified.

## Test Case (Happy Case)

**Test Case ID:** AddMaintenanceSchedule

**Test Priority (Low/Medium/High):** High

**Module Name:** Maintenance Schedule Screen

**Test Title:** Verify adding maintenance schedule with valid input

**Description:** Test that a maintenance schedule can be added successfully when all required fields are correctly filled.

**Test Designed by:** Nattapat, Phonlapat

**Test Designed date:** 24 Oct 2025

**Test Executed by:** Nattapat, Phonlapat

**Test Execution date:** 24 Oct 2025

**Pre-conditions:** User has access to the “Add Maintenance Schedule” page and has at least one gear already registered in the system.

**Dependencies:**

- UI form for adding maintenance schedule must be implemented.
- Database table for storing maintenance schedules must exist.

Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	Provide Gear ID	Gear_id="1"	Valid input string	Valid input string	Pass	
2	Select maintenance date	Maintenace_date = "2025-11-17"	Valid date	Valid date	Pass	
3	Click “Save Schedule” button	-	Inputs are validated and displayed correctly.	Inputs are validated and displayed correctly.	Pass	

**Post-conditions:** Maintenance schedule is successfully displayed with schedule details, including Gear ID and Schedule Date.

## Test Case (Happy Case)

**Test Case ID:** AddRepair

**Test Designed by:** Danai, Aleenta

**Test Priority (Low/Medium/High):** High

**Test Designed date:** 6 Nov 2025

**Module Name:** Add Repair & Inspection

**Test Executed by:** Danai, Aleenta

**Test Title:** Successful Submission of Repair/Inspection Log

**Test Execution date:** 6 Nov 2025

**Description:** Verify that user can successfully fill in all required fields and submit an inspection/repair log.

**Pre-conditions:** User has access to the “Add Repair & Inspections” page.

**Dependencies:**

- UI form for adding inspection/repair must be implemented.
- Database table for storing inspection/repair must exist.
- Data validation rules must be defined.

Step	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Notes
1	Select Gear ID	Gear_id=“ID 1 - Fire Helmet”	Valid input string	Valid input string	Pass	
2	Select Inspection Date	Inspection_date=“2025-11-16”	Valid date	Valid date	Pass	
3	Select Inspector ID	Inspector_id=“ID 2 - Ton Danai”	Valid input string	Valid input string	Pass	
4	Select Inspection Type	Inspection_type=“Routine”	Valid input string	Valid input string	Pass	
5	Provide Notes	Notes= String	Valid input type	Valid input type	Pass	
6	Click “Submit Inspection”		Inputs are validated and displayed correctly.	Inputs are validated and displayed correctly.	Pass	

**Post-conditions:** New inspection/repair is successfully displayed with all inspection details.