WFMS

(Walbro Failure Management System)

Software Requirements Specification

Version 1.0

11/25/2019

Created for:

- The Walbro Company

Created by:

- Anusha Madarapu
- Drake Schmidt
- James Wetters
- James Daws
- Justin Yelsik
- Paul Travis
- Michael Fuller
- Tucker McCarthy-Riley

Table of Contents

1.0 Introduction

This introduction section will describe and define the purpose of this Software Requirement Specification document. It will also include the scope of the system, the definitions and acronyms used, any references to external documents and an overview of the content which follows.

1.1 Purpose

The purpose of this document is to detail the proposed Walbro Failure Management System (WFMS) to be created for Walbro. The engineers who maintain Walbro's current failure management system will be the primary users. In detailing the WFMS, subsystems, and their components these will be defined and elaborated upon. The scope, constraints, requirements, and use cases will provide a proposal to the management team at Walbro.

1.2 System Scope

The WFMS will generally carry out the responsibilities of a FRACAS. Specifically, the WFMS will manage failure causes in a streamlined fashion allowing for engineers to spend less time recording, updating, and maintaining the current documentation. The WFMS will provide recent activities per user, a search bar for parts, and multiple views for failure causes. These additions will aid in reducing time spent searching for and recording data. The product of the features allows the users to spend more time involved with high priority engineering tasks.

The scope of the WFMS includes several features: An implementation of Walbro's current DFMEA/DVP process, Chrome and Firefox long term support, the ability to easily localize the system and WFMS admins having the ability to give customers reduced, customized access to the WFMS.

The Scope Includes
Provide replacement or implementation of Walbro's current DFMEA/DVP process.
Chrome and Firefox long term support browser support.
Localization support
Potential for customers external to Walbro to have access to the WFMS.

The Scope Excludes
Installed client software.
Software that is not free for commercial use
Backup strategies and implementation.

1.3 Definitions and Acronyms

Term	Definition
Action	Single section of a DVP document (comparable to a merge request).
AES	Advanced encryption standard.
ASCII	American Standard Code for Information Interchange. A standard for character encoding in electronic devices.
Assembly	A collection of parts.

Term	Definition
API	Application program interface. An API specifies how software components should interact.
BOM	Bill of materials (list of raw materials)
CAD/CAM	Computer Aided Design Computer Aided Manufacturing.
Completeness tag	A tag or label for DVP tests to show how complete a DVP test is.
DB	Database
DBMS	Database management system.
Detection	Time and means of failure detection
DFMEA	Design Failure Mode and Effect Analysis. A systematic group of activities used to recognize and evaluate potential systems, products or failures.
DVP	Design Validation Plan. A plan for detecting and identifying a specific failure cause.
DVP test	A plan, action, solution, or course of action, that is part of a DVP that acts as an instruction to carry out a DVP.
ESR	Extended support release.
Failure cause	A specific cause of a failure
Failure incident	The occurrence of a part/assembly that is to be recorded.
Failure mode	The result of a failure of a part or assembly.
FRACAS	Failure Reporting Analysis and Corrective Action System. Used to collect, record and analyze system failures.
HTTPS/SSL	Hypertext transport protocol secure/Secure Sockets Layer
Intranet	A local, restricted network, that is private and created using World Wide Web software.
Interface	Means by which the user and any part of the system interact.
Linux	An operating system for personal computers, servers, and mobile devices.
LDAP	Lightweight Directory Access Protocol
LTS	Long term support.
macOS	An operating system for Apple personal computers.
Markdown	A lightweight markup language with plain text formatting syntax used to create rich text from a plain text editor.
Milestone	Contains a specific DVP and its issues not yet started, ongoing issues and completed issues.
Occurrence	Frequency of failure occurrence
OrCAD	a proprietary software tool suite used primarily for electronic design automation.
Part	An individual part of a product
PTC Creo	A suite of Computer-aided design apps supporting product design.
Recent Activity	Any action taken in the system that creates, updates, or deletes any information.
Reliability	Ability to be restored
Severity	Measure of impact on the user of the product
SMTP	Simple Mail Transport Protocol. A protocol to email over the internet or an intranet.
TCP/IP	Transmission Control protocol/internet Protocol. These networking protocols allow computers to communicate.
TUCBW	The use case begins with.
TUCEW	The use case ends with.
UI	User interface.
Unicode	An industry standard for text and symbol encoding and displaying designed to be international and to cover all the world's symbols and languages regarding a computer system.

Term	Definition
Windows	A personal computer operation system.
WFMS	Walbro Failure Management System

1.4 References

No external documents referenced.

1.5 Overview

Moving forward in section 2, the Overall Description, this document provides a high-level overview of the WFMS and explains the functionality of the product. In the third chapter, the system and requirements are detailed. The IEEE 830-1998 recommended practice for Software Requirement Specifications is followed. There is also an appendix that further explains any outside work and details from the requirements.

2.0 Overall Description

This section will provide a high-level overview of the WFMS. The various interfaces the WFMS has and uses will be discussed. Use cases and expanded use cases, an operations overview, site adaptation, user characteristics, system constraints, assumptions about the system, and apportioning anything for future versions will also be overviewed.

2.1 Product Perspective

Currently, Walbro's engineering staff use an outdated, manual system to record failure reports, which includes using both tangible, hand-written records as well as Excel documents. In the current system, finding a specific piece of information on a part or assembly is a tedious, time-consuming process which encourages redundancy in the workflow.

The WFMS is intended to streamline the failure reporting process at Walbro. Not only will it establish a consistent and uniform database of all parts, tests, and failures it will also be easily accessible and provide an intuitive GUI which speeds up the process of reporting failures. Engineers will have the option to either fill out a comprehensive spreadsheet detailing a DFMEA or use a rapid entry form for making faster, less specific contributions.

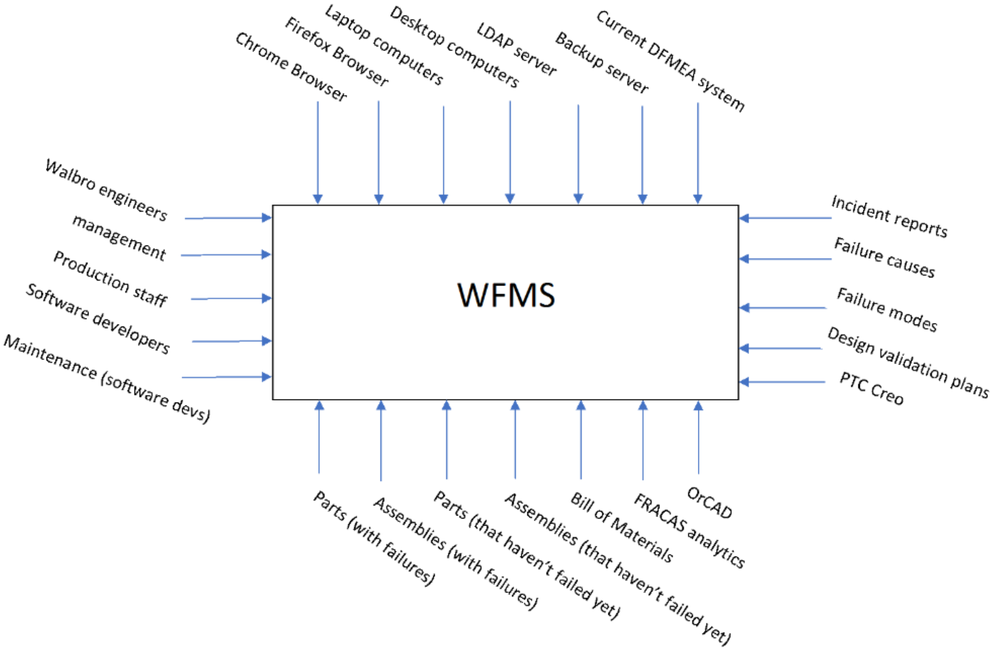


Figure 1 WFMS context diagram.

In addition, the WFMS will allow engineers to easily communicate through a comment section, where they can embed videos, pictures, and other information. While also being convenient, the engineers are comprehensively documenting their process, errors, discoveries, and solutions at the same time. This is something which is not easily achieved through private email or in person meetings.

The WFMS also provides a mechanism to keep track of engineers' progress within a DVP by allowing them to categorize the different DVP tests by their level of completeness. Many of the different elements of the failure reporting process, such as failure causes will be able to be viewed in more than one way. For instance, a list view. Also, a user will be able to sort this information based on different qualities, like date added.

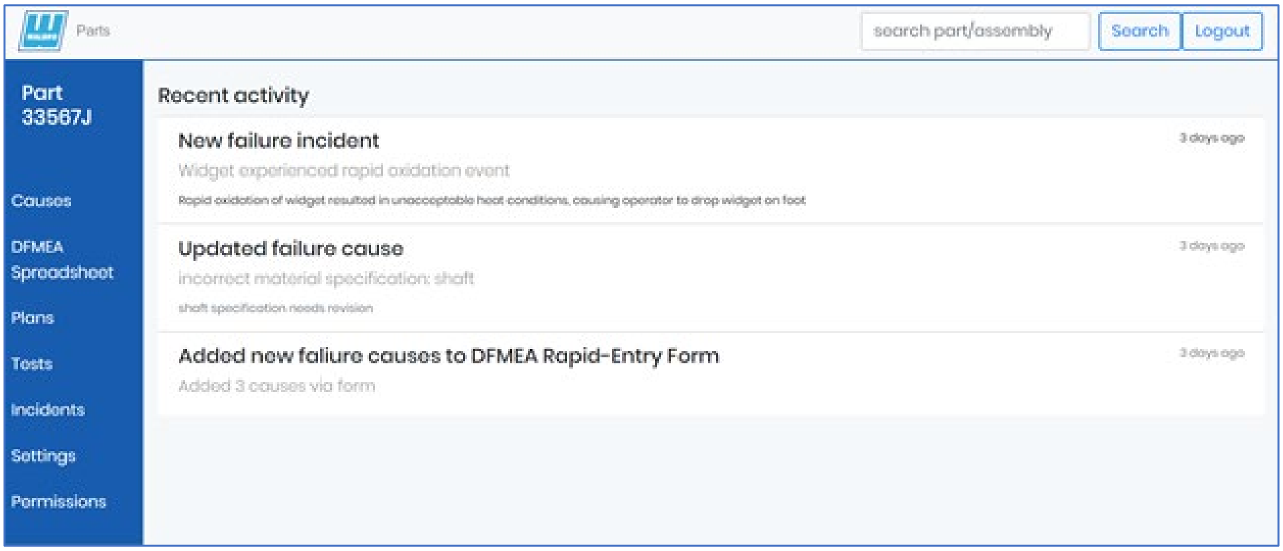
The design philosophy behind the WFMS is one of trust in the user's judgement and competence. The system will not be littered with annoying warnings and confirmation boxes popping up after every edit, deletion, or change made therein. Only in the most extreme cases will such warnings appear. This philosophy was adopted with the advice and direction of Walbro.

2.1.1 System Interfaces

The WFMS will have several system interfaces.

- The WFMS will run on the latest LTS (Long Term Support) version of the web browser Google Chrome. See section 3.5 for more.
- The WFMS will run on the latest ESR (Extend Support Release) version of the web browser Mozilla Firefox (68.1.0 ESR as of Nov. 2019). See section 3.5 for more.
- The WFMS will run on a desktop computer with Windows, Linux, or MacOS.
- The WFMS may run on a mobile phone or tablet, but no consideration is made for this in the design at the request of Walbro. Any mobile support is purely coincidental.

2.1.2 User Interfaces



Recent activity

New failure incident — 3 days ago

Widget experienced rapid oxidation event

Rapid oxidation of widget resulted in unacceptable heat conditions, causing operator to drop widget on foot

Updated failure cause — 3 days ago

incorrect material specification: shaft

shaft specification needs revision

Added new faliure causes to DFMEA Rapid-Entry Form — 3 days ago

Added 3 causes via form

Figure 2 Individual part page.

A user will interact with the WFMS through a web browser. An engineer using the system will have access to a login page, several landing pages, DFMEA interfaces, DVP interfaces, and failure incident interfaces. An admin account will have access to everything an engineer does along with an admin interface that has an interface for adding a new user. Both user types will also have a settings interface, but again, the admin will have more options. The interface has been designed to have fewer warning and confirmation messages to be friendlier to the engineers that will be the main users. The design of these interfaces has been done to the specifications and requests of Walbro.

See Appendix 3.0 for more details.

2.1.3 Hardware Interfaces

The initial deployment of the WFMS will have no hardware interfaces specified at the request of Walbro.

2.1.4 Software Interfaces

The WFMS will interface with PTC Creo, a CAD system frequently used by Walbro's engineering staff, for the purpose of transferring data from PTC Creo projects into the website. PTC Creo has a Java based API, which will be used to interface with the CAD system. Creo will be making requests to the WFMS.

See the requirements [CIR] section for more information.

2.1.5 Communication Interfaces

- The WFMS will interface with an email server over the SMTP protocol to interface with the email server.
- The WFMS will interface with a DBMS over ethernet or WIFI using TCP/IP.

  See section 3.1 for more details and requirements.

2.1.6 Memory

The initial deployment of the WFMS will be without regard for memory constraints at the request of Walbro. As the database of parts and failures continue to increase the disk space will be indefinitely expanded

2.1.7 Operations

An engineer will begin by logging into the WFMS using the log in page. They may enter a new part by clicking the "add part" button on the parts list page. Once an engineer has navigated to a part, they may use the part sidebar to jump to the DFMEA spreadsheet or rapid-entry form. Using the spreadsheet, they can enter in exhaustive information about a DFMEA and use the save feature to push their changes to the database. Using the rapid-entry form, an engineer can quickly report on a DFMEA without giving exhaustive detail. Using the "add cause" button located on the failure cause interface page, an engineer can enter in a new failure cause.

In the DVP section of the parts side bar, an engineer can navigate to either the plans page or the tests page. From the plans page, an engineer can create a new plan by clicking on the "add plan" button or write a comment and submit it using the "save" button. When writing a comment, an engineer can choose to embed images or videos related to the test. If the engineer navigates to the tests page under the DVP section of the sidebar, they can click on the "new test" button to enter in a new DVP test.

An admin can log in using the same log in page as the engineers. Admins have the ability to see and click on the "admin" button on the landing page. Using this, they will be taken to a page where they can add or deactivate the accounts of users. Admins have the access and functionality of engineers.

At the request of Walbro, when and how the system is backed up will be purely at Walbro's discretion.

2.1.8 Site Adaptation Requirements

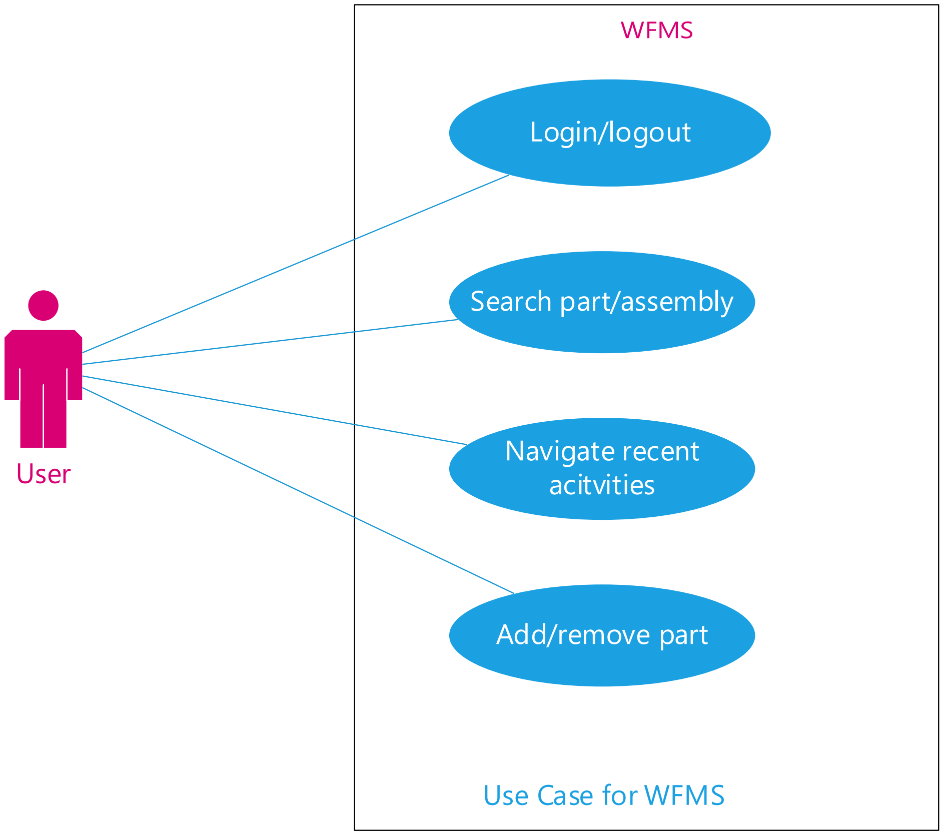
The initial deployment of the system will have no site adaptation requirements specified at the request of Walbro.

2.2 Product Function

This section outlines UML diagrams with expanded use cases where an engineer/admin interacts with the system.

Preconditions for expanded use cases: The system must be connected to the internet. The WFMS must stay open on a browser.

2.2.1 UML use case: WFMS



2.2.1.1 Expanded use case: Login to WFMS

Actor: Engineer	System: WFMS
	0. The login user interface is open and ready for use.
1. **TUCBW the user entering a username and password then pressing login.**	2. The system checks the appropriate database tables for the information the user has offered and: (A) Displays the user landing page of WFMS. Or (B) Displays a prompt specifying that the user has entered invalid information and should try again.

Actor: Engineer	System: WFMS
3. The user: (A) TUCEW the user gaining access to the system and beginning to navigate/use its contents. Or (B) the user reviewing their attempted credentials and returning to step one.	

2.2.1.2 Expanded use case: Search for part/assembly

Actor: Engineer	System: WFMS
	0. The system is idle at the user landing page and waiting for user interaction.
1. TUCBW the user clicking on the search bar at the top of the screen and entering a part/assembly number.	2. The system attempts to match the input with an attribute in the appropriate table from the database. If the attribute is located: (A) The system returns the specified part landing page Or (B) The system returns a notification that the part/assembly number is not registered in the database.
3. The user: **(A) TUCEW the user viewing the desired information for the part requested from the system.** **Or** **(B) The user reconsidering the input and returns to step 1.**	

2.2.1.3 Expanded use case: Logout of WFMS

Actor: Engineer	System: WFMS
	0. The system is idle on the page where the user has last worked and waits for user interaction.
1. TUCBW the user navigating to the top bar of the screen and clicking the logout.	2. The system listens to the logout buttons' click event and: (A) Checks for any unsaved form information on the page and asks the user if he/she wants to save and logout. Or (B) Logs off the user from the account and opens the login page back.

Actor: Engineer	System: WFMS
3. The user: **(A) decides either to save/edit information and Logs out. Repeat (1)** **Or** **(B) TUCEW the user seeing login page back**	

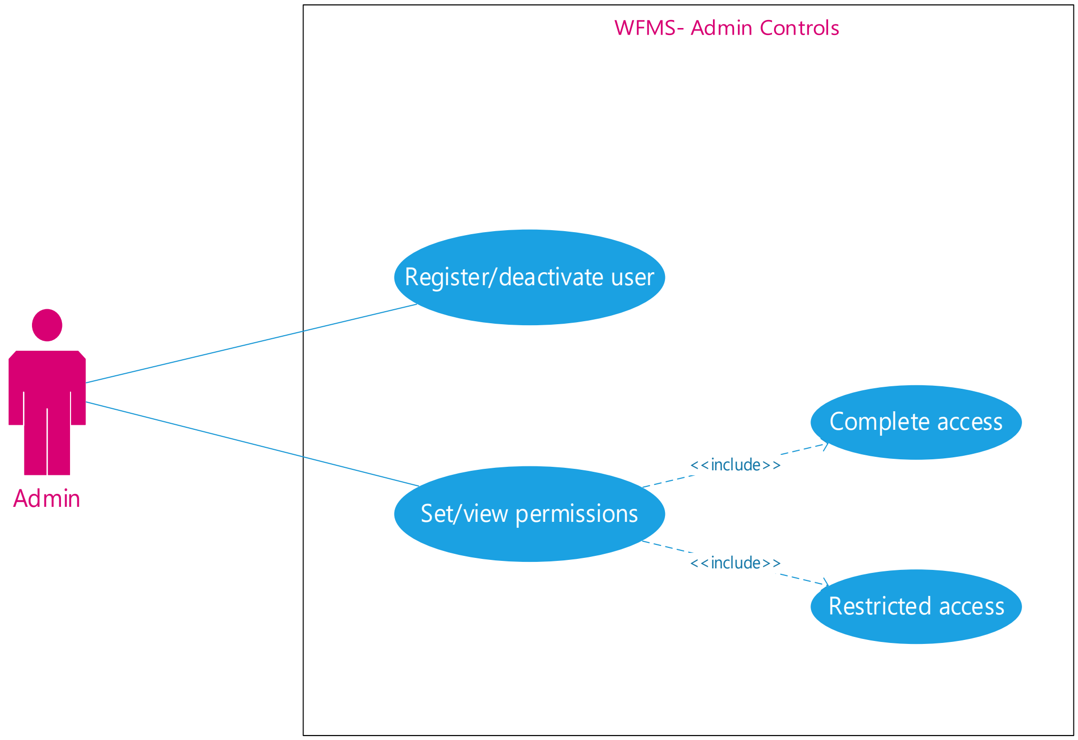
2.2.1.4 Expanded use case: Navigate recent activities

Actor: Engineer	System: WFMS
	0. The user has logged in and is ready to navigate the system.
1. TUCBW the user landing page displaying the ten most recent activities that can be clicked on by the user.	2. Once clicked, the system retrieves data for recent activity requested by user.
3. The user views the recent activity and returns to the list to view more activities.	4. The system continues to retrieve the desired activities as they are requested by the user.
5. TUCEW the user finishing viewing recent activities and exiting the system or moving on to another task.	

2.2.1.5 Expanded use case: Add part

Actor: Engineer	System: WFMS
	0. The user has logged in and is ready to navigate the system.
1. TUCBW the landing page displaying recent activities and a clickable parts button on top.	2. Once the parts button is clicked, the system retrieves all parts from the database and lists them.
3. The user clicks on the add part button available on the bottom of the parts list.	4. The system opens a form for the user to add a part related information.
5. TUCEW the user filling the form and saving it.	

2.2.2 UML use case: WFMS – Admin Controls



Use Case for WFMS

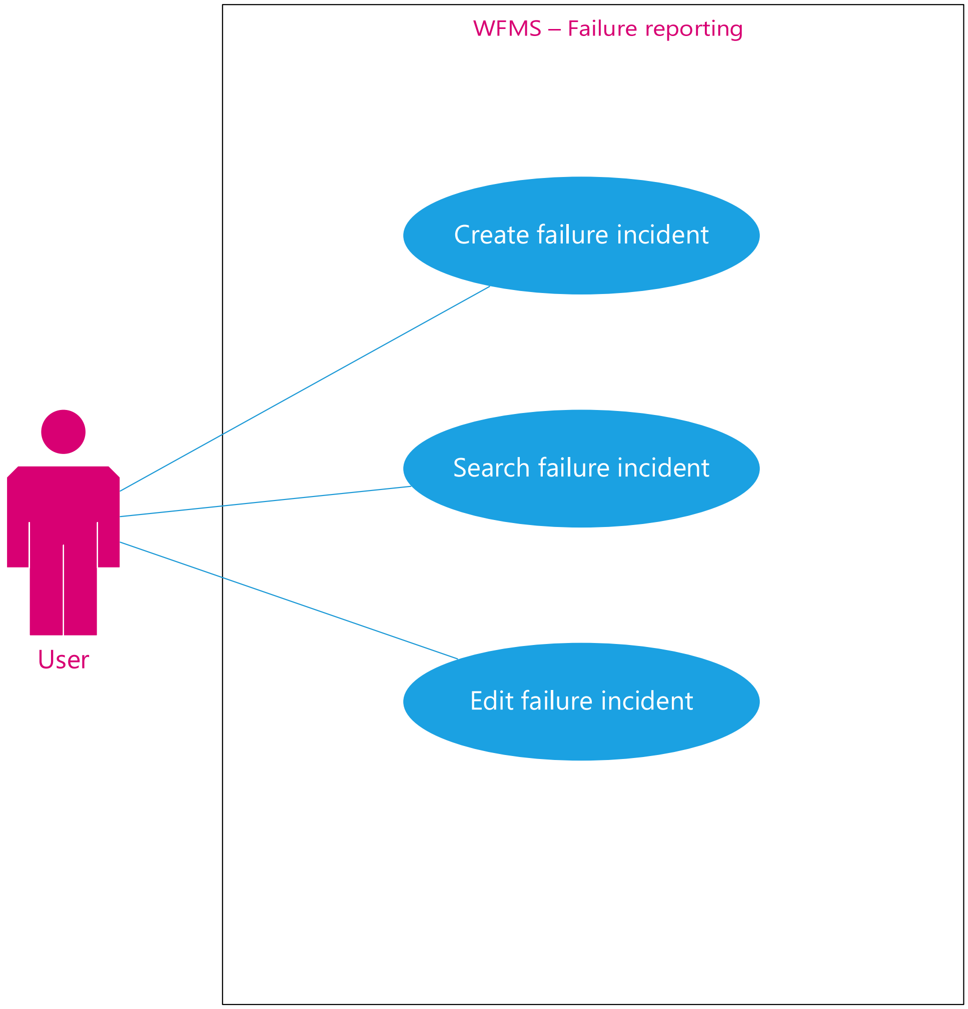
2.2.2.1 Expanded use case: Register new use

Actor: Administrator	System: WFMS
	0. The system has been accessed by an administrator and has displayed the landing page that follows logging in.
1. TUCBW the administrator clicking the admin button on the top right corner of the page.	2. The system returns a view containing all necessary fields to register a new user.
3. The administrator enters the following user credentials into the appropriate fields: first name, last name, and email address before clicking "register".	4. The system: (A) Stores the valid entries into the database and sends the new user an email notification. Or (B) Returns inline error messages for the fields that do not contain valid information.
5. The admin: **(A) TUCEW the user completing his/her registration through the email.** **Or** **(B) Returns to step (3).**	

2.2.2.2 Expanded use case: Deactivate a User

Actor: Administrator	System: WFMS
	1. The system has been accessed by an administrator and has displayed the landing page that follows logging in.
2. TUCBW the administrator clicking the admin button on the top right corner of the page.	3. The system returns a view containing all necessary fields to deactivate a user.
4. The administrator enters the following user credentials into the appropriate fields: first name, last name, and email address before clicking “deactivate”.	5. The system: (A) Terminates user access. Or (B) Returns inline error messages for the fields that do not contain valid information.
6. The admin: **(A) TUCEW the admin seeing a message pop up that says the selected user has been deactivated.** **Or** **(B) Returns to step (4)**	

2.2.3 UML use case: WFMS - Failure Reporting



Use Case for WFMS

2.2.3.1 Expanded use case: Create failure incident

Actor: Engineer	System: WFMS
	0. The system is idle at the user landing page and waiting for user interaction.
1. TUCBW the user clicking on the incidents link from the side bar.	2. The system navigates to the failure incident page.
3. The user clicks the "add incident" button.	4. The system returns a form to add a new incident.
5. The user enters appropriate information to log the incident.	6. The system updates the database list view with appropriate information.
7. TUCEW the user navigating to the failure list view and seeing that the incident has been added to the list of ongoing incidents.	

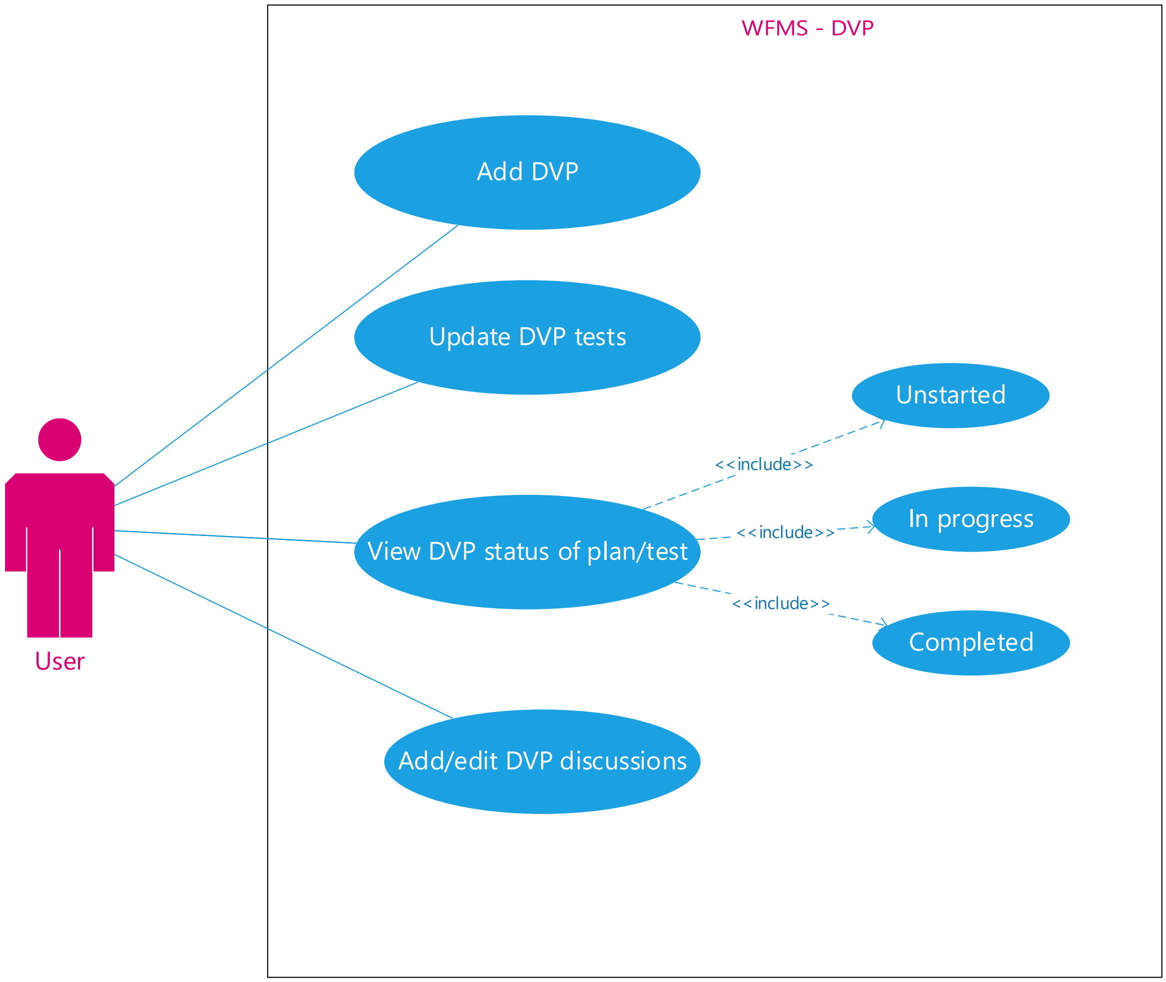
2.2.3.2 Expanded use case: Search failure incident

Actor: Engineer	System: WFMS
	0. The system is idle at the part landing page and waiting for user interaction.
1. TUCBW the user navigating to the incidents page by clicking on the "incidents" link from the side bar.	2. The system returns a list of the incidents associated with the specified part.
3. TUCEW the user viewing the list of failure incidents.	

2.2.3.3 Expanded use case: Edit failure incident

Actor: Engineer	System: WFMS
	0. The system is idle at the part landing page and ready for user interaction.
1. TUCBW the user clicking on the incidents link from the side bar.	2. The system returns a list of the incidents associated with the part.
3. The user clicks on an incident he/she wishes to edit.	4. The system displays a form that details the incident and allows the user to make edits.
5. The user clicks submit when the edits are finished.	6. The system stores the edits in the database.
7. TUCEW the user being able to view the updates in the current incident.	

2.2.4 UML use case: WFMS - DVP



Use Case for WFMS

2.2.4.1 Expanded use case: View DVP status of plan/test.

Actor: Engineer	System: WFMS
	0. The system is idle at the part landing page and waiting for user interaction.
1. TUCBW the user clicking on the "plans" link from the side bar.	2. The system returns a view of the plans and offers an associated view of each plans tests.
3. The user clicks on a plan he/she wishes to view.	4. The System returns the associated view of tests (and their progress) for the specified plan.
5. TUCEW the user viewing the status of the plan/tests.	

2.2.4.2 Expanded use case: Add DVP

Actor: Engineer	System: WFMS
	0. The system is idle at the part landing page and ready for user interaction.
1. TUCBW the user clicking on the "plans" link from the side bar.	2. The system returns the plans page.
3. The user clicks on the "New plan" button.	4. The system returns a prompt containing forms for the user to input a plan.
5. The user fills out the form for the plan and clicks submit.	6. The system stores the information in the appropriate tables in the database allowing for it to display in the list of plans for that part/assembly.
7. TUCEW the user viewing the list of plans confirming that the newly added plan has been added to the appropriate list.	

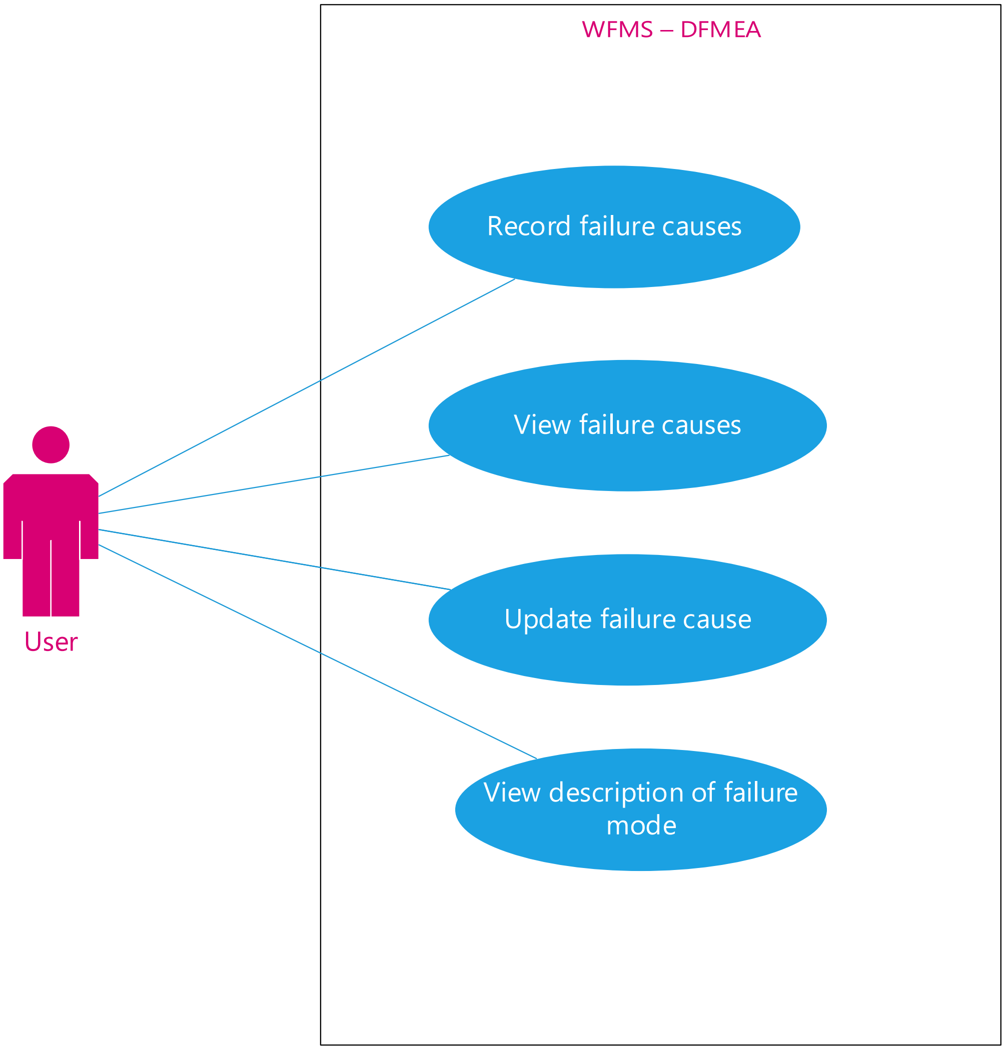
2.2.4.4 Expanded use case: Update DVP tests

Actor: Engineer	System: WFMS
	0. The system is idle at the part landing page and ready for user interaction.
1. TUCBW the user clicking the "tests" link from the side bar.	2. The system returns the tests page.
3. The user clicks on a test he/she wishes to update.	4. The system returns an editable form.
5. The user makes edits to test before clicking the "save" button.	6. The system updates the database to show that the value data is consistent with the updates made by the user.
7. TUCEW the user seeing that the appropriate edits have been displayed when the user views the description of the test.	

2.2.4.5 Expanded use case: Add/edit DVP discussions

Actor: Engineer	System: WFMS
	0. The system is idle at the part landing page and is waiting for user interaction.
1. TUCBW the user clicking on the "plans" link from the side bar.	2. The system returns the plans page.
3. The user enters desired information into discussion field located below the specified plan. The user clicks save.	4. The system stores the discussion thread.
5. TUCEW the user seeing the discussion has been recorded.	

2.2.5 UML use case: WFMS – DFMEA



Use Case for WFMS

2.2.5.1 Expanded use case: View failure causes

Actor: Engineer	System: WFMS
	0. The system is idle at the part landing page and ready for user interaction.
1. TUCBW the user clicking on the "Causes" link from the side bar.	2. The system displays a list view of failure causes and failure modes associated with the part that was searched.
3. The user clicks on a failure cause.	4. The system displays an editable form that provides detailed information about the cause.
5. TUCEW the user viewing the cause information.	

2.2.5.2 Expanded use case: View description of failure mode

Actor: Engineer	**System: WFMS**
	0. The system is idle at the part landing page and waiting for user interaction.
1. TUCBW the user clicking on the "Causes" link from the side bar.	2. The system returns a list of the causes associated with the part.
3. The user clicks on a desired failure cause.	4. The system returns a form showing the details of the cause.
5. TUCEW the user viewing the mode(s) details.	

2.2.5.3 Expanded use case: Record failures in DFMEA spreadsheet

Actor: Engineer	**System: WFMS**
	0. The system is idle at the part landing page and waiting for user interaction.
1. TUCBW the user clicking on the "DFMEA spreadsheet" link on the side bar.	2. The system returns a spreadsheet containing causes associated with the part.
3. The user enters desired cause information before clicking the "save" button.	4. The system stores the recorded information.
5. TUCEW the user viewing the submissions in the cause list (accessed from the side bar).	

2.2.5.4 Expanded use case: Update failure cause

Actor: Engineer	**System: WFMS**
	0. The system is idle at the part landing page and waiting for user interaction.
1. TUCBW the user clicking on the "Causes" link from the side bar.	2. The system returns a list view of the failure causes associated with the part.
3. The user on a desired failure cause.	4. The system returns an editable form that displays all information associated with the failure cause.
5. The user edits the form.	6. The system stores the changes to the form.
7. TUCEW the user seeing that the updates are reflected in the list view.	

2.3 User Characteristics

There will be two main users of the WFMS: users and administrators. There will also be an IT or system maintainer. Users and administrators will generally use the system in the same manner with administrators also able to add a new user to the WFMS.

- Users will generally be engineers at Walbro.
- Administrators will most likely be Walbro management or an engineer team leader or equivalent. They may also be an engineer that has been given special privileges solely for the WFMS.
- A system maintainer may be an administrator or user of the system, but also, they may be someone that only uses the WFMS for system maintenance.
- Walbro may eventually allow its customers as external users for WFMS.

2.4 Constraints

This is an overview of the constraints placed on WFMS by Walbro. The WFMS will be a web application that runs in a web browser and requires no software to be installed on the client machines. The WFMS must be free of cost and have no software licensing restrictions placed on commercial use. The WFMS will have a secure server that uses HTTP/SSL.

See section 3.5 for more details and information.

2.5 Assumptions and Dependencies

The WFMS will be designed and implemented under several assumptions and dependencies.

- Walbro engineers have an expert level of computer interface skills and can be trusted to not make many mistakes. Therefore, in many situations where a warning would traditionally be shown, nothing will be shown, to streamline the workflow.
- Walbro will provide and maintain a webserver for the WFMS to be served from.
- Walbro will provide and maintain a SMTP server for the WFMS to interface with.
- Walbro will provide and maintain a DBMS server for the WFMS to interface with.
- Walbro will provide and maintain a backup of the WFMS, SMTP server, webserver, and DBMS server.
- Walbro will take final delivery of the WFMS source code through and online code repository. A copy in an archive can also be provided.

2.6 Apportioning of Requirements

As of version 1.0 of the WFMS, several design considerations were requested by Walbro but were given a low priority. Pending developer time and manpower, these items may not be included in version 1.0.

- A dependency graph for tests that could replace the current manual graphing process.
- User permissions so that different users will only have access to certain sections or parts.
- The complete deployment be made reproducible and automated using a container technology or something equivalent.
- Users authentication through a LDAP server would be preferred, but not a requirement.

Walbro also asked that a few design considerations be made for the future, but not be implemented as of version 1.0.

- The ability for external users to the Walbro intranet to access the WFMS through the internet.
- Localization so the WFMS UI can be displayed in many different languages, especially Spanish.
- The ability to interface with the Walbro OrCAD system.

3.0 System Features and Requirements

This section details the WFMS requirements. The requirements are setup as rows in tables of ID tags, requirements, and tests. The ID tags use an [ID TAG] encoding for top level requirements. These are also in bold text. Sub-requirements use an [ID TAG – SUBTAG*NUMBER*] encoding. This was done to make the requirements more intuitive.

Requirements are addressed at the highest level of the WFMS possible. For example, if a requirement is system wide, it is address as system wide and not repeated several times throughout.

3.1 External Interfaces

[EI] Email Interface

ID	Requirement	Test
[EI]	**The WFMS shall provide a mechanism to interface with a SMTP email server.**	
[EI-IO]	**The EI shall provide a mechanism for data input and data output.**	
[EI-IO1]	The EI shall copy the subsequent (number determined by Walbro) characters after @username tag, then paste those characters into the email body.	Tag a WFMS user in a discussion thread. Check the subsequent email to verify a brief excerpt from the discussion.
[EI-IO2]	The EI shall include a one-click context switch to the WFMS event which triggered the notification.	Tag a WFMS user in a discussion thread. Check the subsequent email to verify a one-click context switch to the event which triggered the email message.
[EI-DB]	**The EI shall provide a mechanism for WFMS database associations.**	

ID	Requirement	Test
[EI-DB1]	The EI shall allow a mechanism to make queries against the WFMS database.	Tag a WFMS user in a discussion thread. Verify the correct user received an email notification. Check each WFMS username and email combination against an external source of reliable data.
[EI-LC]	**The EI shall provide a mechanism which enables a set of programmable logic constraints to govern the behavior of the EI within the WFMS.**	
[EI-LC1]	The EI shall send email notifications when certain Walbro-specified conditions are met regarding specific actions against WFMS data. All users involved within those conditions will receive an email notification.	Trigger an action which meets the Walbro condition for this event. Verify that all users tagged in the event received an email message.
[EI-LC2]	The EI shall send an email to a new user when they are attempting to create an account.	Create new user and verify the new user's email received the email.
[EI-LC3]	The EI shall be accessible via all WFMS discussion threads by using the '@' symbol followed by a valid WFMS username.	Access an existing discussion thread in the WFMS. Tag a WFMS user as described. Verify the existence of a new email message in the WFMS recipient's email inbox.
[EI-LC4]	The EI shall only be accessible via Walbro-approved email domains.	Use a Walbro-approved email domain. Verify the ability to receive emails from the WFMS. Try to register a new user with an email domain which is not approved by Walbro. Verify that registration is denied.

3.2 Functions – Functional Requirements

[SW] System-Wide

ID	Requirement	Test
[SW-TBC]	**The WFMS shall provide system-wide group of top bar controls (SW-TBC)**	
[SW-TBC1]	The SW-TBC shall provide a mechanism which allows a user to input search criteria.	Enter data into the search field. Verify that search field contents match the data entered.
[SW-TBC2]	The SW-TBC shall provide a mechanism which allows context switching to a WFMS account settings menu.	Activate the settings control. Verify the WFMS user's account settings menu options are displayed.
[SW-TBC3]	The SW-TBC shall provide a mechanism which allows a user to logout from the WFMS.	Login to the WFMS. Activate the logout control. Verify the WFMS now requires login to perform any actions against the WFMS.
[SW-TBC4]	The SW-TBC shall provide a mechanism which allows context switching to a WFMS account admin menu. *(only for WFMS administrator accounts)*	Login to WFMS with an administrator account and activate the admin control. Verify the system administrator menu options are functional. Login to the WFMS with a non-administrator account. Verify the admin control mechanism is not visible.
[SW-TBC5]	The SW-TBC shall provide a mechanism which allows one-click context switching to the user landing page.	Login to WFMS. First, click a part to switch the screen context away from the user landing page.

ID	Requirement	Test
	This control shall be activated by clicking the Walbro icon.	Then click the Walbro logo. Verify the screen displays the user landing page.
[SW-SBC]	**The WFMS shall provide system-wide group of sidebar controls (SW-SBC)**	
[SW-SBC1]	The SW-SBC shall provide a mechanism which displays the current part number in scope.	Perform any action within the WFMS that requires a part number to be in scope. Verify the SBC contains a visible reference to that part number.
[SW-SBC2]	The SW-SBC shall provide a mechanism which allows one-click context switching to a failure cause list for the part number currently in scope.	Whenever the SW-SBC is visible, click the causes control option. Verify the WFMS updates the screen contents with a list failure causes for the associated part.
[SW-SBC3]	The SW-SBC shall provide a mechanism which allows one-click context switching to a DFMEA spreadsheet for the part number currently in scope.	Whenever the SW-SBC is visible, click the DFMEA spreadsheet control option. Verify the WFMS updates the screen contents with DFMEA spreadsheet data for the associated part.
[SW-SBC4]	The SW-SBC shall provide a mechanism which allows one-click context switching to a list of DVPs for the part number currently in scope.	Whenever the SW-SBC is visible, click the plans control option. Verify the WFMS updates the screen contents with DVP-specific data for the associated part.
[SW-SBC5]	The SW-SBC shall provide a mechanism which allows one-click context switching to an exhaustive tests list for the part number currently in scope.	Whenever the SW-SBC is visible, click the tests control option. Verify the WFMS updates the screen contents with an exhaustive tests list for the associated part.
[SW-SBC6]	The SW-SBC shall provide a mechanism which allows one-click context switching to a failure incidents list for the part number currently in scope.	Whenever the SW-SBC is visible, click the incidents control option. Verify the WFMS updates the screen contents with a list of failure incidents for the associated part.
[SW-SBC7]	The SW-SBC shall provide a mechanism which allows one-click context switching to a settings menu for the part number currently in scope.	Whenever the SW-SBC is visible, click the settings control option. Verify the WFMS updates the screen contents with a settings menu for the associated part.
[SW-SBC8]	The SW-SBC shall provide a mechanism which allows one-click context switching to a permissions menu for the part number currently in scope.	Whenever the SW-SBC is visible, click the permissions control option. Verify the WFMS updates the screen contents with a settings menu for the associated part.
[SW-IO]	**The WFMS shall provide system-wide standards for data input and data output.**	
[SW-IO1]	The WFMS shall allow ASCII characters in text fields.	Enter data into a text field. Verify the text field contents match the data entered. Confirm result against an exhaustive list of ASCII characters allowed within the WFMS.
[SW-IO2]	The WFMS shall provide a mechanism which allows for programmable mouse and cursor events to be triggered by various activities within the WFMS.	Open a part’s DFMEA spreadsheet. Hover the cursor over a severity field entry. Verify the appearance of a pop-up bubble containing

ID	Requirement	Test
		information about Walbro's severity metrics. Confirm result against an exhaustive list of mouse/cursor events within the WFMS.
[SW-IO3]	The WFMS shall provide a mechanism which allows system-wide data validation for all user input.	Open a part's DFMEA spreadsheet. Try to input letters into the severity text field. Verify the restriction of data types to integer-only for the severity field. Confirm result against an exhaustive list of data validation conditions within the WFMS.
[SW-IO4]	The WFMS shall provide a mechanism which allows system-wide predictive data input auto-complete. *Note: parsing local data only – not talking Google level deep learning AI. For example, repeated causes & modes in a DFMEA.*	Open a part's DFMEA spreadsheet. Begin typing a failure mode which already exists within the WFMS. Verify the existence of predictive auto-complete options for the currently keyed text. Confirm result against an exhaustive list of data auto-complete conditions within the WFMS.
[SW-IO5]	The WFMS shall provide a mechanism which allows system-wide data input autofill. *Note: limited scope. Fields which can be populated via inference from a database association or an efficient use of OS tools.*	Open a part's failure cause list. Create a new failure cause. Verify the failure mode field has been pre-populated with the associated failure mode from the previous screen. Confirm result against an exhaustive list of data autofill conditions within the WFMS.
[SW-IO6]	The WFMS shall provide a mechanism which determines conditional handling of blank text fields.	Open a part's failure cause list. Click a cause to view more details about that cause. Delete any existing contents within the discussion text field. Verify the conditional allowance of this field being empty. Open a part's failure cause list. Click the control to add a new cause. Attempt to save with an empty title field. Verify the conditional forbiddance of this field being empty. Confirm results against an exhaustive list of conditional blank field handlings within the WFMS.
[SW-IO7]	The WFMS shall provide a mechanism which enforces a character limit on text fields.	Enter a string of data which exceeds the WFMS ability for processing (limit set by Walbro). Verify the WFMS throws an exception and warns the user.
[SW-DB]	**The WFMS shall provide system-wide database associations.**	
[SW-DB1]	The WFMS shall provide a mechanism which logs all user activities against the WFMS database.	Login to WFMS. Commit several actions against the WFMS database. Verify those actions are both committed to WFMS database and appear on top of the user's ULP under the recent activity list.
[SW-DB2]	The WFMS shall provide a mechanism for the SW-TBC to make queries against the WFMS database.	Enter a part number to be searched into the SW-TBC. Compare the query result to an external source of reliable data.
[SW-DB3]	The WFMS shall provide a mechanism for the SW-SBC 'Causes' menu option to make queries against the WFMS database.	With a part in scope, click the 'Causes' menu option on the SW-SBC. Compare the query result to an external source of reliable data.

ID	Requirement	Test
[SW-DB4]	The WFMS shall provide a mechanism for the SW-SBC 'DFMEA Spreadsheet' menu option to make queries against the WFMS database.	With a part in scope, click the 'DFMEA Spreadsheet' menu option on the SW-SBC. Compare the query result to an external source of reliable data.
[SW-DB5]	The WFMS shall provide a mechanism for the SW-SBC 'Plans' menu option to make queries against the WFMS database.	With a part in scope, click the 'Plans' menu option on the SW-SBC. Compare the query result to an external source of reliable data.
[SW-DB6]	The WFMS shall provide a mechanism for the SW-SBC 'Tests' menu option to make queries against the WFMS database.	With a part in scope, click the 'Tests' menu option on the SW-SBC. Compare the query result to an external source of reliable data.
[SW-DB7]	The WFMS shall provide a mechanism for the SW-SBC 'Incidents' menu option to make queries against the WFMS database.	With a part in scope, click the 'Incidents' menu option on the SW-SBC. Compare the query result to an external source of reliable data.
[SW-DB8]	The WFMS shall provide a mechanism for the SW-SBC 'Settings' menu option to make queries against the WFMS database.	With a part in scope, click the 'Settings' menu option on the SW-SBC. Compare the query result to an external source of reliable data.
[SW-DB9]	The WFMS shall provide a mechanism for the SW-SBC 'Permissions' menu option to make queries against the WFMS database.	With a part in scope, click the 'Permissions' menu option on the SW-SBC. Compare the query result to an external source of reliable data.
[SW-DB10]	The WFMS shall provide a mechanism for the SW-SBC 'Settings' menu option to make update statements to the WFMS database.	With a part in scope, click the 'Settings' menu option on the SW-SBC. Change a setting. Verify the change is evident in WFMS behavior.
[SW-DB11]	The WFMS shall provide a mechanism for the SW-SBC 'Permissions' menu option to make update statements to the WFMS database.	With a part in scope, click the 'Permissions' menu option on the SW-SBC. Change a permission. Verify the change is evident in WFMS behavior.
[SW-LC]	**The WFMS shall provide a mechanism which enables a set of programmable logic constraints to govern all behavior within the WFMS.**	
[SW-LC1]	The WFMS shall provide a mechanism for error handling with a unique error code for each error type. The WFMS shall then attach that unique error code to every error message.	Trigger an error event within the WFMS. Verify the error message code against an external list of error codes. Repeat for each type of error on the list.
[SW-LC2]	The WFMS shall provide a mechanism for enforcing an upper limit on the amount of time which an account may be logged in.	Login to the WFMS. Leave the account logged in for 24 consecutive hours. Verify the account is automatically logged out once the session timer reaches 24 hours.

[SA] System Administrator

ID	Requirement	Test
[SA-C]	**The WFMS shall provide system administrator controls.**	

ID	Requirement	Test
[SA-C1]	The WFMS shall provide the system administrator with a mechanism for creating new user accounts.	Login to the WFMS with an administrator account and activate the admin control on the SW-TBC. Verify that a control exists with clear and obvious intent to add a user. Complete and then submit the new user form. Verify the successful creation of a new account after the new user completes the invitation process by following the link sent to the user's email.
[SA-C2]	The WFMS shall provide the system administrator with a mechanism for deactivating current user accounts.	Login to the WFMS with an administrator account and activate the admin control on the SW-TBC. Verify that a control exists with clear and obvious intent to deactivate a user. Complete and then submit the deactivate user form. Verify the successful deactivation of the account by trying to login to the WFMS from the deactivated account.
[SA-IO]	**The WFMS shall provide system administrator data input and data output.**	
[SA-IO1]	The WFMS shall provide a registration interface for adding new users.	Login to the WFMS with an administrator account and activate the admin control on the SW-TBC. Verify the account creation form is shown on the screen.
[SA-IO2]	The WFMS shall provide a deactivation interface for deactivating current users.	Login to the WFMS with an administrator account and activate the admin control on the SW-TBC. Verify the account deactivation form is shown on the screen.
[SA-DB]	**The WFMS shall provide system administrator database associations.**	
[SA-DB1]	The WFMS shall provide a mechanism for system administrators to make transactions against the WFMS database users.	Login to the WFMS with an administrator account and activate the admin control on the SW-TBC. Add and/or deactivate several test users. Query the WFMS database for a list of active users. Compare the query results against a reliable external data source.
[SA-LC]	**The WFMS shall provide a mechanism which enables a set of programmable logic constraints to govern the behavior of the system administrator within the WFMS.**	
[SA-LC1]	The WFMS shall allow only the system administrators to add new users.	Complete the add new user steps as described above in the test for SA-CA1. Now login to the WFMS with a non-administrator account. Verify the admin control does not appear in the SW-TBC.

ID	Requirement	Test
[SA-LC2]	The WFMS shall allow only the system administrators to deactivate current users	Complete the add new user steps as described above in the test for SA-CA2. Now login to the WFMS with a non-administrator account. Verify the admin control does not appear in the SW-TBC.
[SA-LC3]	The WFMS shall provide a mechanism to ensure that all text fields in the user registration (and deactivation) forms are non-empty at the time of submission.	Attempt to fill out the registration (or deactivation) form while leaving any combination of text fields blank. Verify the WFMS prompts the user with a message regarding which field(s) is the cause of the error message.
[SA-LC4]	The WFMS shall provide a mechanism to ensure that a username is unique before allowing new user registration.	Attempt to fill out the registration form using a duplicate username. Part of the WFMS new user registration protocol will query the proposed username against the list of current (plus deactivated) users. If the username is unique, then activation is permitted. If the username is a duplicate, then verify the WFMS prompts the user with a message to try again with a new username.
[SA-LC5]	The WFMS shall provide a mechanism to ensure that an email address is both unique and from a Walbro-approved domain.	Attempt to fill out the registration form using a duplicate email address or an email address from an unrecognized domain. Part of the WFMS new user registration protocol will query the proposed email address against a list of current (plus deactivated) users. If the email address is unique and from a domain approved by Walbro, then activation is permitted. If the email address is a duplicate or not from a permitted domain, then verify the WFMS prompts the user with a message to try again using a new email address.
[SA-LC6]	The WFMS shall provide a mechanism to require a repeated entry of an intended new user password at the time of registration (I.e. both inputs must match).	Attempt to fill out the registration form with the correct password in one text field, but a mismatching entry in the other text field. Verify the WFMS prompts the user with a message to try again while using two identical password entries.
[SA-LC7]	The WFMS shall provide a mechanism to require a new password to be at least eight characters in length.	Attempt to fill out the registration form with a password that is fewer than eight characters. Verify the WFMS prompts the user with a message to try again while using a password that is at least eight characters in length.

[LP] Login Page

ID	Requirement	Test
[LP]	**The WFMS shall have a user login page (LP).**	
[LP-C]	**The LP shall allow a mechanism for WFMS user controls.**	
[LP-C1]	The LP shall provide a mechanism which allows a user to save the username & password login credentials locally.	Verify that a control exists with clear and obvious intent to save a user's credentials. Enable the 'Remember me' control while logging in to the WFMS. Logout of the WFMS. Verify the user's login credentials are saved upon next login attempt.
[LP-C2]	The LP shall provide a mechanism to begin a login event.	Verify that a control exists with clear and obvious intent to begin a login event. Enter login credentials into the username and password fields. Activate the login control. Assuming valid credentials, verify the WFMS is now shown on the screen.
[LP-C3]	The LP shall provide a mechanism to trigger an email-based password recovery and password update protocol.	From the WFMS login screen, verify that a control exists with clear and obvious intent to begin a password recovery event. Activate the password recovery control. Follow the directions for password recovery in the user's Walbro-approved email. Verify the process upon successful login.
[LP-IO]	**The LP shall allow a mechanism for data input and data output.**	
[LP-IO1]	The LP shall include a mechanism which allows username input into a text field.	Enter data into the username text field via keyboard (or mouse shortcut). Verify each character shows up as entered.
[LP-IO2]	The LP shall include a mechanism which allows user password input into a text field.	Enter data into the user password text field via keyboard (or mouse shortcut). Since entered data will be masked, verify the input via successful login attempt.
[LP-DB]	**The LP shall allow a mechanism for WFMS database associations.**	
[LP-DB1]	The LP shall provide a mechanism which allows queries against the WFMS database.	Upon login attempt, the WFMS will query the database for the user's login credentials. If the user exists in the database and the credentials are valid, then verify this process via successful login attempt.
[LP-LC]	**The LP shall provide a mechanism which enables a set of programmable logic constraints to govern the behavior of the LP within the WFMS.**	
[LP-LC1]	The LP shall provide a mechanism which will timestamp a current session's begin time upon successful login. Timestamp data shall be stored locally.	Upon successful login attempt, verify the accurate timestamp data exists in the appropriate memory location.

[ULP] User Landing Page

ID	Requirement	Test
[ULP]	**The WFMS shall have a user landing page (ULP).**	
[ULP-C]	**The ULP shall allow a mechanism for WFMS user controls.**	
[ULP-C1]	The ULP shall provide a mechanism to include the system-wide top bar controls (SW-TBC).	Visual inspection. Verify the top bar control will always be shown on the ULP.
[ULP-C2]	The ULP shall provide a mechanism which populates the screen with a list of parts.	Upon login, verify that a list of parts shows up on the screen. The list contents shall include only the user's activity upon parts within the WFMS made during the past 8 days.
[ULP-C3]	The ULP shall provide a mechanism which allows one-click navigation to a new list containing only the activity of an associated (clicked) part.	From the ULP, click on a part. Verify that a new list is generated. The new list shall contain an exhaustive list of activity (including other WFMS users) made against that part.
[ULP-IO]	**The ULP shall allow a mechanism for data input and data output.**	
[ULP-IO1]	The ULP shall implement a list view.	Visual inspection. Verify that a list containing the user's recent activity shows up after login. (Assumes user activity exists against the WFMS)
[ULP-IO2]	The ULP shall allow the list contents to be updated by queries against the WFMS database.	Login to the WFMS. Notice the ULP contents. Commit several actions against several WFMS parts. Revisit the ULP. Verify the ULP contents update to include the most recent actions.
[ULP-DB]	**The ULP shall allow a mechanism for WFMS database associations.**	
[ULP-DB1]	The ULP shall allow a mechanism to make queries against the WFMS database.	From the ULP, view the list of recent activity for a given user. Compare the list to an external source of reliable data.
[ULP-DB2]	The ULP shall allow a mechanism to filter the results shown from WFMS database queries.	From the ULP, view the list of recent activity for a given user. Notice the ULP list contains a summary rather than an exhaustive representation of recent activity. Compare the list to an external source of reliable data.
[ULP-LC]	**The ULP shall provide a mechanism which enables a set of programmable logic constraints to govern the behavior of the ULP within the WFMS.**	
[ULP-LC1]	The ULP list (assuming it is non-empty) shall only display a filtered set of fields from each record which exists in the list. The fields displayed are dependent upon the edit(s) made.	The ULP list is meant to function like a quick summary for recent activity. For testing, commit an action against a WFMS part. Then revisit the ULP. Verify that every aspect from the edit may be shown within the ULP summary. Revisit the edit to ensure data persistence.

ID	Requirement	Test
[ULP-LC2]	The ULP list shall allow a mechanism to apply a unique set of query filter criteria for edits regarding failure incidents.	Submit an edit against a failure incident in the WFMS. Visit the ULP. Verify the top result in the ULP list contains a summary of the edit.
[ULP-LC3]	The ULP list shall allow a mechanism to apply a unique set of query filter criteria for edits regarding failure causes.	Submit an edit against a failure cause in the WFMS. Visit the ULP. Verify the top result in the ULP list contains a summary of the edit.
[ULP-LC4]	The ULP list shall allow a mechanism to apply a unique set of query filter criteria for edits regarding design validation plans.	Submit an edit against a design validation plan in the WFMS. Visit the ULP. Verify the top result in the ULP list contains a summary of the edit.
[ULP-LC5]	The ULP list shall allow a mechanism to apply a unique set of query filter criteria for edits regarding design validation plan tests.	Submit an edit against a design validation plan test in the WFMS. Visit the ULP. Verify the top result in the ULP list contains a summary of the edit.
[ULP-LC6]	The ULP list shall allow a mechanism to apply a unique set of query filter criteria for edits regarding the exhaustive parts list.	Submit an edit against the exhaustive parts list in the WFMS. Visit the ULP. Verify the top result in the ULP list contains a summary of the edit.
[ULP-LC7]	The ULP list shall allow a mechanism to apply a unique set of query filter criteria for edits regarding parts tests.	Submit an edit against a parts test in the WFMS. Visit the ULP. Verify the top result in the ULP list contains a summary of the edit.
[ULP-LC8]	The ULP list shall allow a mechanism to apply a unique set of query filter criteria for edits regarding DFMEA spreadsheets.	Submit an edit against a DFMEA spreadsheet in the WFMS. Visit the ULP. Verify the top result in the ULP list contains a summary of the edit.
[ULP-LC9]	The ULP list shall allow a mechanism to apply an additional timeframe filter upon each unique set of query filter criteria for all edits within the WFMS.	Submit an edit against the WFMS database. Wait a week, then visit the ULP. Verify the edit still exists in the ULP list. Wait two more days, then revisit the ULP. Verify the edit is no longer displayed in the ULP list.

[EPL] Exhaustive Parts List

ID	Requirement	Test
[EPL]	**The WFMS shall have an exhaustive parts list (EPL).**	
[EPL-C]	**The EPL shall provide a mechanism for WFMS user controls.**	
[EPL-C1]	The EPL shall provide a mechanism to include the system-wide top bar controls (SW-TBC).	Visual inspection. Verify the top bar control will always be shown on the EPL.
[EPL-C2]	The EPL shall provide a mechanism which adds a part to the exhaustive parts list in the WFMS database.	Visual inspection. Verify that a control exists with clear and obvious intent to add a part.
[EPL-C3]	The EPL shall provide a mechanism which allows the screen contents to be repopulated with database records from the next sequential set of results in the exhaustive parts list.	Visual inspection. Verify that a control exists with clear and obvious intent to view the next sequential set of records.

ID	Requirement	Test
[EPL-C4]	The EPL shall provide a mechanism which allows the screen contents to be repopulated with database records from the previous sequential set of results in the exhaustive parts list.	Visual inspection. Verify that a control exists with clear and obvious intent to view the previous sequential set of records.
[EPL-C5]	The EPL shall provide a mechanism which allows the screen contents to be repopulated with database records from sequential sets of results in the exhaustive parts list. These numerically labeled controls shall update their labels based upon a relative position within the exhaustive parts list.	Visual inspection. Verify that a control exists with clear and obvious intent to select from sequential sets of records. Verify the labels update as the list is traversed.
[EPL-IO]	**The EPL shall provide a mechanism for data input and data output.**	
[EPL-IO1]	The EPL shall display a list of part records.	Visual inspection. Verify that a list of parts is displayed on screen.
[EPL-IO2]	The EPL shall provide a mechanism which allows list contents to be updated by queries against the WFMS database.	Submit an update against a part in the WFMS. Visit the EPL. Verify the result in the EPL contains the updated part.
[EPL-DB]	**The EPL shall provide a mechanism for WFMS database associations.**	
[EPL-DB1]	The EPL shall allow a mechanism to make queries against the WFMS database.	From the EPL, view a set of records returned from a query against the WFMS database. Compare the EPL results to an external source of reliable data.

[EPLI] Exhaustive Parts List Interface

ID	Requirement	Test
[EPLI]	**The WFMS shall have an exhaustive parts list (EPLI).**	
[EPLI-C]	**The EPLI shall provide a mechanism for WFMS user controls.**	
[EPLI-C1]	The EPLI shall provide a mechanism to save the data contained in the EPLI fields to the exhaustive parts list.	Visual inspection. Verify that a control exists with clear and obvious intent to save a new record. Input all required data to the appropriate fields. Activate the save control. Return to the EPL to verify the part has been successfully added.
[EPLI-IO]	**The EPLI shall provide a mechanism for data input and data output.**	
[EPLI-IO1]	The EPLI shall provide a mechanism which allows a user to input a part number.	Enter data into the part number text field via keyboard (or mouse shortcut). Verify each character shows up as entered.
[EPLI-IO2]	The EPLI shall provide a mechanism which allows a user to input a date.	Enter data into the date text field via keyboard (or mouse shortcut). Verify each character shows up as entered.

ID	Requirement	Test
[EPLI-IO3]	The EPLI shall provide a mechanism which allows a user to input a part serial number.	Enter data into the serial number text field via keyboard (or mouse shortcut). Verify each character shows up as entered.
[EPLI-IO4]	The EPLI shall provide a mechanism which allows a user to input a description of the part being added to the EPL.	Enter data into the description text field via keyboard (or mouse shortcut). Verify each character shows up as entered.
[EPLI-DB]	**The EPLI shall provide a mechanism for WFMS database associations.**	
[EPLI-DB1]	The EPL shall allow a mechanism for making insert statements to the WFMS parts database.	From EPLI, insert data for a given part. Compare result to external (reliable) data.

[PLP] Parts Landing Page

ID	Requirement	Test
[PLP]	**The WFMS shall have a parts landing page (PLP).**	
[PLP-C]	**The PLP shall allow a mechanism for WFMS user controls.**	
[PLP-C1]	The PLP shall provide a mechanism to include the system-wide top bar controls (SW-TBC).	Visual inspection. Verify the top bar control will always be shown on the PLP.
[PLP-C2]	The PLP shall provide a mechanism to include the system-wide parts sidebar controls (SW-PSB).	Visual inspection. Verify the parts sidebar will always be shown on the PLP.
[PLP-IO]	**The PLP shall allow a mechanism for data input and data output.**	
[PLP-IO1]	The PLP shall display a list of recent activities associated with a chosen part.	Visual inspection. Verify that a list of activities against the chosen part is shown on screen.
[PLP-IO2]	The PLP shall allow the list contents to be updated by queries against the WFMS database.	Make an edit to a part. Return to the PLP list. Verify the recent activity is shown at the top of the chosen part's activity list.
[PLP-DB]	**The PLP shall allow a mechanism for WFMS database associations.**	
[PLP-DB1]	The PLP shall allow a mechanism to make queries against the WFMS DFMEA database tables.	From PLP interface, view data for a given part. Compare result to external (reliable) data.
[PLP-DB2]	The PLP shall allow a mechanism to make queries against the WFMS DVP database tables.	From PLP interface, view data for a given part. Compare result to external (reliable) data.
[PLP-LC]	**The PLP shall provide a mechanism which enables a set of programmable logic constraints to govern the behavior of the ULP within the WFMS.**	
[PLP-LC1]	The PLP list (assuming it is non-empty) shall only display a filtered set of fields from each record which exists the list. The fields displayed are dependent upon the edit(s) made.	The PLP list is meant to function like a quick summary for a part's activity. For testing, commit an action against a WFMS part. Then revisit the PLP. Verify that every aspect from the edit may be shown within the PLP summary. Revisit the edit to ensure data persistence.

ID	Requirement	Test
[PLP-LC2]	The PLP list shall allow a mechanism to apply a unique set of query filter criteria for edits regarding failure incidents.	Submit an edit against a failure incident in the WFMS. Visit the PLP. Verify the top result in the PLP list contains a summary of the edit.
[PLP-LC3]	The PLP list shall allow a mechanism to apply a unique set of query filter criteria for edits regarding failure causes.	Submit an edit against a failure cause in the WFMS. Visit the PLP. Verify the top result in the PLP list contains a summary of the edit.
[PLP-LC4]	The PLP list shall allow a mechanism to apply a unique set of query filter criteria for edits regarding design validation plans.	Submit an edit against a design validation plan in the WFMS. Visit the PLP. Verify the top result in the PLP list contains a summary of the edit.
[PLP-LC5]	The PLP list shall allow a mechanism to apply a unique set of query filter criteria for edits regarding design validation plan tests.	Submit an edit against a design validation plan test in the WFMS. Visit the PLP. Verify the top result in the PLP list contains a summary of the edit.
[PLP-LC6]	The PLP list shall allow a mechanism to apply a unique set of query filter criteria for edits regarding the exhaustive parts list.	Submit an edit against the exhaustive parts list in the WFMS. Visit the PLP. Verify the top result in the PLP list contains a summary of the edit.
[PLP-LC7]	The PLP list shall allow a mechanism to apply a unique set of query filter criteria for edits regarding parts tests.	Submit an edit against a parts test in the WFMS. Visit the PLP. Verify the top result in the PLP list contains a summary of the edit.
[PLP-LC8]	The PLP list shall allow a mechanism to apply a unique set of query filter criteria for edits regarding DFMEA spreadsheets.	Submit an edit against a DFMEA spreadsheet in the WFMS. Visit the PLP. Verify the top result in the PLP list contains a summary of the edit.

[DFMEA] Design Failure Mode and Effect Analysis Spreadsheet

ID	Requirement	Test
[DFMEA]	**The WFMS shall have a DFMEA spreadsheet (DFMEA).**	
[DFMEA-C]	**The DFMEA shall provide a mechanism for WFMS user controls.**	
[DFMEA-C1]	The DFMEA shall provide a mechanism to include the system-wide top bar controls (SW-TBC).	Visual inspection. Verify the top bar control will always be shown on the DFMEA spreadsheet.
[DFMEA-C2]	The DFMEA shall provide a mechanism to include the system-wide parts sidebar controls (SW-PSB).	Visual inspection. Verify the parts sidebar control will always be shown on the DFMEA spreadsheet.
[DFMEA-C3]	The DFMEA spreadsheet shall always have an empty row at the bottom of the table.	Visual inspection. Verify the bottom row of the DFMEA is always available for adding a new record. For testing, add a record then verify the appearance of a new line below the added record.
[DFMEA-C4]	The DFMEA spreadsheet shall provide a mechanism to save new data to the DFMEA spreadsheet.	Visual inspection. Verify that a control exists with clear and obvious intent to save a new record. Enter a new record from the bottom line of the

ID	Requirement	Test
		DFMEA spreadsheet. Activate the save control. Revisit the page to ensure data persistence.
[DFMEA-C5]	The DFMEA spreadsheet shall provide a mechanism to save edits to existing data that was changed within the DFMEA spreadsheet.	Visual inspection. Verify that a control exists with clear and obvious intent to save an edited record. Edit a record in the DFMEA spreadsheet. Activate the save control. Revisit the page to ensure data persistence.
[DFMEA-C6]	The DFMEA spreadsheet shall have a mechanism that displays a test-specific metric for the keyed integer value when the user hovers over the severity input.	Visual inspection. Verify that a notification occurs when hovering the cursor over severity data entries. The notification shall contain a relative range for a metric specified by Walbro.
[DFMEA-C7]	The DFMEA spreadsheet shall have a mechanism that displays a test-specific metric for the keyed integer value when the user hovers over the occurrence input.	Visual inspection. Verify that a notification occurs when hovering the cursor over occurrence data entries. The notification shall contain a relative range for a metric specified by Walbro.
[DFMEA-C8]	The DFMEA spreadsheet shall have a mechanism that displays a test-specific metric for the keyed integer value when the user hovers over the detection input.	Visual inspection. Verify that a notification occurs when hovering the cursor over detection data entries. The notification shall contain a relative range for a metric specified by Walbro.
[DFMEA-C9]	The DFMEA spreadsheet shall implement the autofill and auto-complete features specified in the system-wide section.	Visual inspection. Verify that features work for both cases. First begin typing a failure mode which is present in the DFMEA spreadsheet. The WFMS should offer an autocomplete option after several characters are typed in. Then verify that causes with a previously specified failure mode have an option for auto-filling the data field.
[DFMEA-C10]	The DFMEA spreadsheet shall have a mechanism to allow the user to adjust the width of columns.	Position the cursor at the border between two columns. Verify the cursor then changes to a double arrow. Verify that column width is now able to be changed via drag & drop mouse control.
[DFMEA-IO]	**The DFMEA shall provide a mechanism for data input and data output.**	
[DFMEA-IO1]	The DFMEA spreadsheet shall provide a mechanism which allows a failure cause to be recorded.	Enter data into the ‘Cause’ cell via keyboard (or mouse shortcut). Verify each character shows up as entered.
[DFMEA-IO2]	The DFMEA spreadsheet shall provide a mechanism which allows a failure mode to be recorded.	Enter data into the ‘Mode’ cell via keyboard (or mouse shortcut). Verify each character shows up as entered.
[DFMEA-IO3]	The DFMEA spreadsheet shall provide a mechanism which allows a failure cause description to be recorded.	Enter data into the ‘Description’ cell via keyboard (or mouse shortcut). Verify each character shows up as entered.

ID	Requirement	Test
[DFMEA-IO4]	The DFMEA spreadsheet shall provide a mechanism which allows the severity of a failure cause to be recorded.	Enter data into the 'Severity' cell. Integer values from 0 to 10 should be accepted. Numeric values outside the specified range should be rejected. Non-numeric values should be rejected. Non-integer numeric values (I.e. Decimals) should be rejected.
[DFMEA-IO5]	The DFMEA spreadsheet shall provide a mechanism which allows the occurrence rate of a failure cause to be recorded.	Enter data into the 'Occurrence' cell. Integer values from 0 to 10 should be accepted. Numeric values outside the specified range should be rejected. Non-numeric values should be rejected. Non-integer numeric values (I.e. Decimals) should be rejected.
[DFMEA-IO6]	The DFMEA spreadsheet shall provide a mechanism which allows the detection rate of a failure cause to be recorded.	Enter data into the 'Detection' cell. Integer values from 0 to 10 should be accepted. Integer values from 0 to 10 should be accepted. Numeric values outside the specified range should be rejected. Non-numeric values should be rejected. Non-integer numeric values (I.e. Decimals) should be rejected.
[DFMEA-IO7]	The DFMEA spreadsheet shall provide a mechanism which allows current controls to be recorded.	Enter data into the 'Current Controls' cell via keyboard (or mouse shortcut). Verify each character shows up as entered.
[DFMEA-IO8]	The DFMEA spreadsheet shall provide a mechanism which allows Recommended Actions(s) to be recorded.	Enter data into the 'Recommended Actions(s)' cell via keyboard (or mouse shortcut). Verify each character shows up as entered.
[DFMEA-IO9]	The DFMEA spreadsheet shall provide a warning message if an attempt to make a context switch is made while the spreadsheet contains multiple rows of unsaved data.	Enter multiple rows of data. Do not activate the save control. Attempt a context switch away from the DFMEA spreadsheet. Verify the existence of a warning message before a context switch is processed.
[DFMEA-DB]	**The DFMEA shall provide a mechanism for WFMS database associations.**	
[DFMEA-DB1]	The DFMEA spreadsheet shall allow a mechanism to make queries against the WFMS database.	From the DFMEA spreadsheet, view a set of records for a specified part. Verify the query results against an external source of reliable data.
[DFMEA-DB2]	The DFMEA spreadsheet shall allow a mechanism to make insert statements to the WFMS database.	From the DFMEA spreadsheet, insert data into a specified part record and submit the data to the database. Verify data persistence by reloading the DFMEA spreadsheet. Verify the results against an external source of reliable data.
[DFMEA-DB3]	The DFMEA spreadsheet shall allow a mechanism to make update statements against the WFMS database.	From the DFMEA spreadsheet, update data against a specified part record and submit the data to the database. Verify data persistence by

ID	Requirement	Test
		reloading the DFMEA spreadsheet. Verify the results against an external source of reliable data.

[FCL] Failure Cause List

ID	Requirement	Test
[FCL]	**The system shall have a failure cause list (FCL).**	
[FCL-C]	**The FCL shall allow a mechanism for WFMS user controls.**	
[FCL-C1]	The FCL shall provide a mechanism to include the system-wide top bar controls (SW-TBC).	Visual inspection. Verify the top bar control will always be shown on the failure cause list.
[FCL-C2]	The FCL shall provide a mechanism to include the system-wide parts sidebar controls (SW-PSB).	Visual inspection. Verify the top bar control will always be shown on the failure cause list.
[FCL-C3]	The FCL shall provide a mechanism which allows a failure cause to be added to the failure cause list for a selected part.	Visual inspection. Verify that a control exists with clear and obvious intent to add a new cause.
[FCL-C4]	The FCL list record containers shall be expandable and collapsible.	Visual inspection. Click to verify that a collapsed container will expand. Click to verify that an expanded container will collapse.
[FCL-C5]	The FCL record container (expanded) shall allow context switching to a new screen with additional discussion, documentation, and attachment file options.	Visual inspection. Verify the expanded containers show more data. Click to verify context switches.
[FCL-C6]	The FCL list view element (collapsed) shall expand revealing the options mentioned above.	Visual inspection. Verify the collapsed containers will expand after being clicked.
[FCL-IO]	**The FCL shall allow a mechanism for data input and data output.**	
[FCL-IO1]	The FCL shall display a list of failure cause records.	Visual inspection. Verify the list contains a set of failure cause records.
[FCL-IO2]	The FCL shall allow the list view contents to be updated by queries against the WFMS parts database table.	Add a failure cause to a part. Return to the FCL. Verify the recent activity is displayed in the failure cause list contents.
[FCL-IO3]	The FCL list view shall contain all child parts when viewing an assembly.	Search the WFMS for an assembly. Context switch to the part's FCL. Verify the list contents include aggregate child parts.
[FCL-DB]	**The FCL shall allow a mechanism for WFMS database associations.**	
[FCL-DB1]	The FCL shall allow a mechanism to make queries against the WFMS database.	From the FCL, view a set of records for a specified part. Verify the query results against an external source of reliable data.
[FCL-DB2]	The FCL shall allow a mechanism to make insert statements to the WFMS database.	From the FCL, insert data into a specified part record and submit the data to the database. Verify data persistence by reloading the FCL

ID	Requirement	Test
		Verify the results against an external source of reliable data.
[FCL-DB3]	The FCL shall allow a mechanism to make update statements to the WFMS database.	From the FCL, update data against a specified part record and submit the data to the database. Verify data persistence by reloading the FCL Verify the results against an external source of reliable data.

[FCI] Failure Cause Interface

ID	Requirement	Test
[FCI]	**The system shall have a failure cause interface (FCI).**	
[FCI-C]	**The FCI shall allow a mechanism for WFMS user controls.**	
[FCI-C1]	The FCI shall provide a mechanism which allows a new failure cause to be added to the failure cause list.	Visual inspection. Verify that a control exists with clear and obvious intent to add a new cause.
[FCI-IO]	**The FCI shall allow a mechanism for data input and data output.**	
[FCI-IO1]	The FCI shall provide a mechanism which allows a user to input a title for a new failure cause.	Enter data into the title text field via keyboard (or mouse shortcut). Verify each character shows up as entered.
[FCI-IO2]	The FCI shall provide a mechanism which allows a user to input a brief description for a new failure cause.	Enter data into the brief description text field via keyboard (or mouse shortcut). Verify each character shows up as entered.
[FCI-IO3]	The FCI shall provide a mechanism which automatically populates a data field with the failure mode from the previous screen.	Visual inspection. Verify failure mode field is pre-populated.
[FCI-IO4]	The FCI shall provide a mechanism which allows a user to input a detailed description for a new failure cause.	Enter data into the detailed description text field via keyboard (or mouse shortcut). Verify each character shows up as entered.
[FCI-DB]	**The FCI shall allow a mechanism for WFMS database associations.**	
[FCI-DB1]	The FCI shall allow a mechanism to make insert statements to the WFMS database.	From the FCI, insert data into a specified part record and submit the data to the database. Verify data persistence by reloading the FCL Verify the results against an external source of reliable data.

[FIL] Failure Incident List

ID	Requirement	Test
[FIL]	**The WFMS shall have a failure incident list (FIL).**	

ID	Requirement	Test
[FIL-C]	**The FIL shall allow a mechanism for WFMS user controls.**	
[FIL-C1]	The FIL shall provide a mechanism to include the system-wide top bar controls (SW-TBC).	Visual inspection. Verify the top bar control will always be shown on the failure incident list.
[FIL-C2]	The FIL shall provide a mechanism to include the system-wide parts sidebar controls (SW-PSB).	Visual inspection. Verify the parts sidebar control will always be shown on the failure incident list.
[FIL-C3]	The FIL shall provide a mechanism which allows a failure incident to be added to the failure incident list for a selected part.	Visual inspection. Verify that a control exists with clear and obvious intent to add a new incident.
[FIL-C4]	The FIL shall provide a mechanism which allows one-click context switching to a screen with information about a failure incident.	Visual inspection. Click the failure incident. Verify the WFMS updates the screen contents with information about a failure incident.
[FIL-C5]	The FIL shall provide a mechanism which allows one-click context switching to a screen with information about a failure cause.	Visual inspection. Click the failure cause. Verify the WFMS updates the screen contents with information about a failure cause.
[FIL-IO]	**The FIL shall allow a mechanism for data input and data output.**	
[FIL-IO1]	The FIL shall allow the list contents to be updated by queries against the WFMS parts database.	Add a failure incident to a part. Return to the FIL. Verify the recent activity is displayed in the failure incident list contents.
[FIL-DB]	**The FIL shall allow a mechanism for WFMS database associations.**	
[FIL-DB1]	The FIL shall allow a mechanism to make query statements against the WFMS database.	From the FIL, view the incident data for a specified part. Verify the query results against an external source of reliable data.
[FIL-DB2]	The FIL shall allow a mechanism to make insert statements to the WFMS database.	From the FII, insert data into a specified incident record and submit the data to the database. Verify data persistence by reloading the FIL Verify the results against an external source of reliable data.
[FIL-DB3]	The FIL shall allow a mechanism to make update statements against the WFMS database.	From the FII, update data for a specified incident record and submit the data to the database. Verify data persistence by reloading the FIL Verify the results against an external source of reliable data.

[FII] Failure Incident Interface

ID	Requirement	Test
[FII]	**The WFMS shall have a failure incident interface (FII).**	
[FII-C]	**The FII shall allow a mechanism for WFMS user controls.**	
[FII-C1]	The FII shall provide a mechanism which allows a failure incident to be added to the failure incident list for a selected part.	Visual inspection. Verify that a control exists with clear and obvious intent to add a new incident.

ID	Requirement	Test
[FII-C2]	The FII shall provide a mechanism which allows an existing failure incident to be edited.	Visual inspection. Verify that a control exists with clear and obvious intent to submit changes to an incident record.
[FII-IO]	**The FII shall allow a mechanism for data input and data output.**	
[FII-IO1]	The FII shall provide a mechanism which allows a description to be added to a failure incident record.	Enter data into the title text field via keyboard (or mouse shortcut). Verify each character shows up as entered.
[FII-IO2]	The FII shall provide a mechanism which allows the description field to be populated by a query against the WFMS database *(for existing incidents).*	Click an existing failure incident record from the FIL. Verify the description content appears in the description text field. Verify the query against external reliable data.
[FII-DB]	**The FII shall allow a mechanism for WFMS database associations.**	
[FII-DB1]	The FII shall allow a mechanism to make query statements against the WFMS database.	From the FII, view the description data for a specified incident. Verify the query results against an external source of reliable data.
[FII-DB2]	The FII shall allow a mechanism to make insert statements to the WFMS database. *(note: mismatch FII vs FIL is intentional in the test. Insert is committed in FII, but viewed in FIL)*	From the FII, insert data into a specified incident record and submit the data to the database. Verify data persistence by reloading the FIL Verify the results against an external source of reliable data.
[FII-DB3]	The FII shall allow a mechanism to make update statements against the WFMS database. *(note: mismatch FII vs FIL is intentional in the test. Edit is committed in FII, but viewed in FIL)*	From the FII, update data for a specified incident record and submit the data to the database. Verify data persistence by reloading the FIL Verify the results against an external source of reliable data.

[DVP] Design Validation Plan

ID	Requirement	Test
[DVP]	**The WFMS shall have a design validation plan (DVP).**	
[DVP-C]	**The DVP shall allow a mechanism for WFMS user controls.**	
[DVP-C1]	The DVP shall provide a mechanism to include the system-wide top bar controls (SW-TBC).	Visual inspection. Verify the top bar control will always be shown on the design validation plan.
[DVP-C2]	The FIL shall provide a mechanism to include the system-wide parts sidebar controls (SW-PSB).	Visual inspection. Verify the parts sidebar control will always be shown on the design validation plan.
[DVP-C3]	The DVP shall provide a mechanism which allows a new plan to be added to a DVP.	Visual inspection. Verify that a control exists with clear and obvious intent to add a new plan.
[DVP-C4]	The DVP shall provide a mechanism which allows a new comment to be added to a DVP discussion thread.	Visual inspection. Verify that a control exists with clear and obvious intent to add a new comment to a discussion thread.

ID	Requirement	Test
[DVP-C5]	The DVP shall provide three status display containers which represent the following states: ‘Unstarted’ - ‘In Progress’ - ‘Completed’.	Visual Inspection. Verify the existence of three unique containers which represent the three distinct testing states.
[DVP-C6]	The DVP shall provide a mechanism which allows a user to change the status of an existing DVP test state (via drag & drop between test states).	Move a test from one status to another (drag & drop). Reload the DVP context screen. Verify the test status accurately displays the update.
[DVP-IO]	**The DVP shall allow a mechanism for data input and data output.**	
[DVP-IO1]	The DVP shall display a list of design validation plans.	Visual inspection. Verify a list of design validation plans is shown on screen.
[DVP-IO2]	The DVP discussion thread shall provide an input mechanism.	Visual inspection. Verify that a text field exists. Input text into the field. Verify the discussion field accurately displays the new text.
[DVP-IO3]	The DVP shall allow the list contents to be updated by queries against the WFMS database.	Add a plan to a DVP. Return to the DVP. Verify the recent activity is displayed in the DVP list contents.
[DVP-IO4]	The DVP mechanism shall provide a visual representation of relative progress through a completeness mechanism. *(note: assumes left to right is an increase in progress)*	Select a DVP with plans and incomplete tests. Move several tests at least one state to the right. Via visual inspection, verify the completeness indicator represents the change in test states.
[DVP-DB]	**The DVP shall allow a mechanism for WFMS database associations.**	
[DVP-DB1]	The DVP shall allow a mechanism to make queries against the WFMS database.	From the DVP screen, view the data for a specified DVP. Verify the query results against an external source of reliable data.
[DVP-DB2]	The DVP shall allow a mechanism to make insert statements to the WFMS database.	From the DVP screen, insert data into a specified plan and submit the data to the database. Verify data persistence by reloading the DVP. Verify the results against an external source of reliable data.

[DVPPI] Design Validation Plan (New Plan Interface)

ID	Requirement	Test
[DVPPI]	**The WFMS shall have a design validation plan (DVPPI).**	
[DVPPI-C]	**The DVPPI shall allow a mechanism for WFMS user controls.**	
[DVPPI-C1]	The DVPPI shall provide a mechanism which allows a user to add a new plan.	Visual inspection. Verify that a control exists with clear and obvious intent to add a new plan.
[DVPPI-IO]	**The DVPPI shall allow a mechanism for data input and data output.**	
[DVPPI-IO1]	The DVPPI shall provide a mechanism which allows a name to be added to a new plan.	Enter data into the name text field via keyboard (or mouse shortcut). Verify each character shows up as entered.
[DVPPI-IO2]	The DVPPI shall provide a mechanism which allows a description to be added to a new plan.	Enter data into the description text field via keyboard (or mouse shortcut). Verify each character shows up as entered.
[DVPPI-DB]	**The DVPPI shall allow a mechanism for WFMS database associations.**	
[DVPPI-DB1]	The DVP shall allow a mechanism to make insert statements to the WFMS database. *(note: mismatch DVP vs DVPPI is intentional in the test. Insert is committed in DVPPI, but viewed in DVP)*	From the DVPPI screen, insert a new plan into a DVP and submit the data to the database. Verify data persistence by reloading the DVP. Verify the results against an external source of reliable data.

[TL] Tests List

ID	Requirement	Test
[TL]	**The WFMS shall have a design validation plan (TL).**	
[TL-C]	**The TL shall allow a mechanism for WFMS user controls.**	
[TL-C1]	The TL shall provide a mechanism to include the system-wide top bar controls (SW-TBC).	Visual inspection. Verify the top bar control will always be shown on the tests list.
[TL-C2]	The TL shall provide a mechanism to include the system-wide parts sidebar controls (SW-PSB).	Visual inspection. Verify the parts sidebar control will always be shown on the tests list.
[TL-C3]	The TL shall provide a mechanism which allows a new test to be added to the tests list.	Visual inspection. Verify that a control exists with clear and obvious intent to add a new test.
[TL-IO]	**The TL shall allow a mechanism for data input and data output.**	
[TL-IO1]	The TL shall display a list of tests for a specified part.	Visual inspection. Verify the list shows a set of tests for a specified part.
[TL-IO2]	The TL shall allow the list contents to be updated by queries against the WFMS database.	Add a test to the tests list. Return to the tests list. Verify the recent activity is displayed in the tests list contents.

ID	Requirement	Test
[TL-DB]	**The TL shall allow a mechanism for WFMS database associations.**	
[TL-DB1]	The TL shall allow a mechanism to make queries against the WFMS DVP database table.	From the TL screen, view the data for a specified part's tests list. Verify the query results against an external source of reliable data.
[TL-DB2]	The TL shall allow a mechanism to make insert statements to the WFMS DVP database table.	From the TL screen, insert a new test into a tests list and submit the data to the database. Verify data persistence by reloading the tests list. Verify the results against an external source of reliable data.

[TI] New Tests Interface

ID	Requirement	Test
[TI]	**The WFMS shall have a design validation plan (TI).**	
[TI-C]	**The TI shall allow a mechanism for WFMS user controls.**	
[TI-C1]	The TI shall provide a mechanism which allows a new test to be added to the tests list.	Visual inspection. Verify that a control exists with clear and obvious intent to add a new test.
[TI-IO]	**The TI shall allow a mechanism for data input and data output.**	
[TI-IO1]	The TI shall provide a mechanism which allows a title to be added to the new test.	Enter data into the title text field via keyboard (or mouse shortcut). Verify each character shows up as entered.
[TI-IO2]	The TI shall provide a mechanism which allows an optional DVP association for the new test.	Visual inspection. Two scenarios for testing. Firstly, leave it blank. Secondly, select from a menu of DVP associations for the part.
[TI-IO3]	The TI shall provide a mechanism which allows a brief description to be added to the new test.	Enter data into the title text field via keyboard (or mouse shortcut). Verify each character shows up as entered.
[TI-IO4]	The TI shall provide a mechanism which allows a detailed description to be added to the new test.	Enter data into the detailed description text field via keyboard (or mouse shortcut). Verify each character shows up as entered.
[TI-DB]	**The TI shall allow a mechanism for WFMS database associations.**	
[TI-DB1]	The TI shall allow a mechanism to make insert statements to the WFMS database. *(note: mismatch TI vs tests list is intentional in the test. Insert is committed in TI, but viewed in the tests list)*	From the TI screen, insert a new test into a tests list and submit the data to the database. Verify data persistence by reloading the tests list. Verify the results against an external source of reliable data.
[TI-LC]	**The TI shall provide a mechanism which enables a set of programmable logic constraints to govern the behavior of the TI within the WFMS.**	

[US] User Settings

ID	Requirement	Test
[US]	**The WFMS shall have user settings (US).**	
[US-C]	**The US shall provide a mechanism for WFMS users to view their account details.**	
[US-C1]	The US shall provide a mechanism for users to view their permissions, roles, name, and email.	Visual Inspection. Verify separately the user's permissions, roles, name, and email are displayed.
[US-IO]	**The US shall provide a mechanism for WFMS users to modify their account details.**	
[US-IO1]	The US shall provide a mechanism for WFMS users to change their password.	Login to the WFMS as a valid user. Navigate to User Settings and use the mechanism to change the password. Logout of the user's account and login to the WFMS with the user's new password.
[US-DB]	**The US shall provide a mechanism for WFMS database associations.**	
[US-DB1]	The US shall allow a mechanism to make queries against the WFMS database.	From US, view the current user's data. Verify the query results against an external source of reliable data.
[US-DB2]	The US shall allow a mechanism to make update statements against the WFMS database.	From the US, user inputs a new password and submits the data to the database. Verify the update by logging out and then back in with the user's new password.

[CIR] CAD Interfacing Requirements

ID	Requirement	Test
[CIR]	The system shall interface with the PTC Creo CAD system using the built in Java API.	A tester will use the website to import information from a test PTC Creo project. Visual inspection will be used to see if the information that was imported is the same as the information in the test PTC Creo project.

3.3 Performance Requirements

Performance requirements like the number of simultaneous users supported, the maximum number of connections supported, the amount of information that can be processed, and the number of transactions that will be processed in a given amount of time shall be solely at the discretion of Walbro. This is because these statistics will be determined by the hardware used/constraints placed on the system by Walbro itself.

3.4 Logical Database Requirements

Most of the logical database requirements are better illustrated in the database schema and UML diagrams shown in Appendix 4.0. Therefore, in the interest of reducing redundancy, they shall not be repeated in this section.

Some of the basic database requirements which the WFMS must fulfil are:

- The system shall be able to read from the database.
- The system shall be able to write to the database.
- The system shall be associated with only one database.

3.5 Design Constraints

The WFMS has several design constraints place on it at the request of Walbro. These are not formal requirements.

- The WFMS will use an HTTPS/SSL configured webserver with a recognized, signed certificate.
- The WFMS will need to be hosted on a system that is free from commercial license restrictions. Windows server is not acceptable.
- The WFMS will not have any software that needs to be installed on a client interface machine and will be accessible through a web browser.
- The WFMS will be constrained to user interfaces on desktop or laptop personal computers. Any mobile support will be purely coincidental.
- The WFMS must be accessible on the latest LTS of Google Chrome. Version 77.0.3865.90 as of 11/25/2019.
- The WFMS must be accessible on the latest ESR of Mozilla Firefox. Version 68.2.0 as of 11/25/2019.
- The WFMS must include a spreadsheet like view for with a collapsible group of cells.
- The WFMS must be accessible by multiple, simultaneous users.
- The WFMS must be able to be backed up and restored trivially.

3.5.1 Standards Compliance

The WFMS will have no requirements that relate to industry standards compliance.

3.6 Software System Attributes

3.6.1-2 Reliability and Availability

The WFMS will be installed on a web server of Walbro’s choosing. Reliability and availability of the WFMS will be wholly dependent on the web server it is installed on.

3.6.3 Security

The WFMS shall have the following security feature based on confidentiality, integrity, and availability, to protect the data from misuse.

- Registered users’ passwords for WFMS shall be encrypted in the database.

3.6.4 Maintainability

This section specifies the attributes of the WFMS as they relate to ease of maintenance.

- The WFMS shall be editable with any standard text editor.

3.6.5 Portability

This section specifies the attributes of the WFMS as they relate to portability.

- The WFMS shall be deployed through an interface that is automated and reproducible.

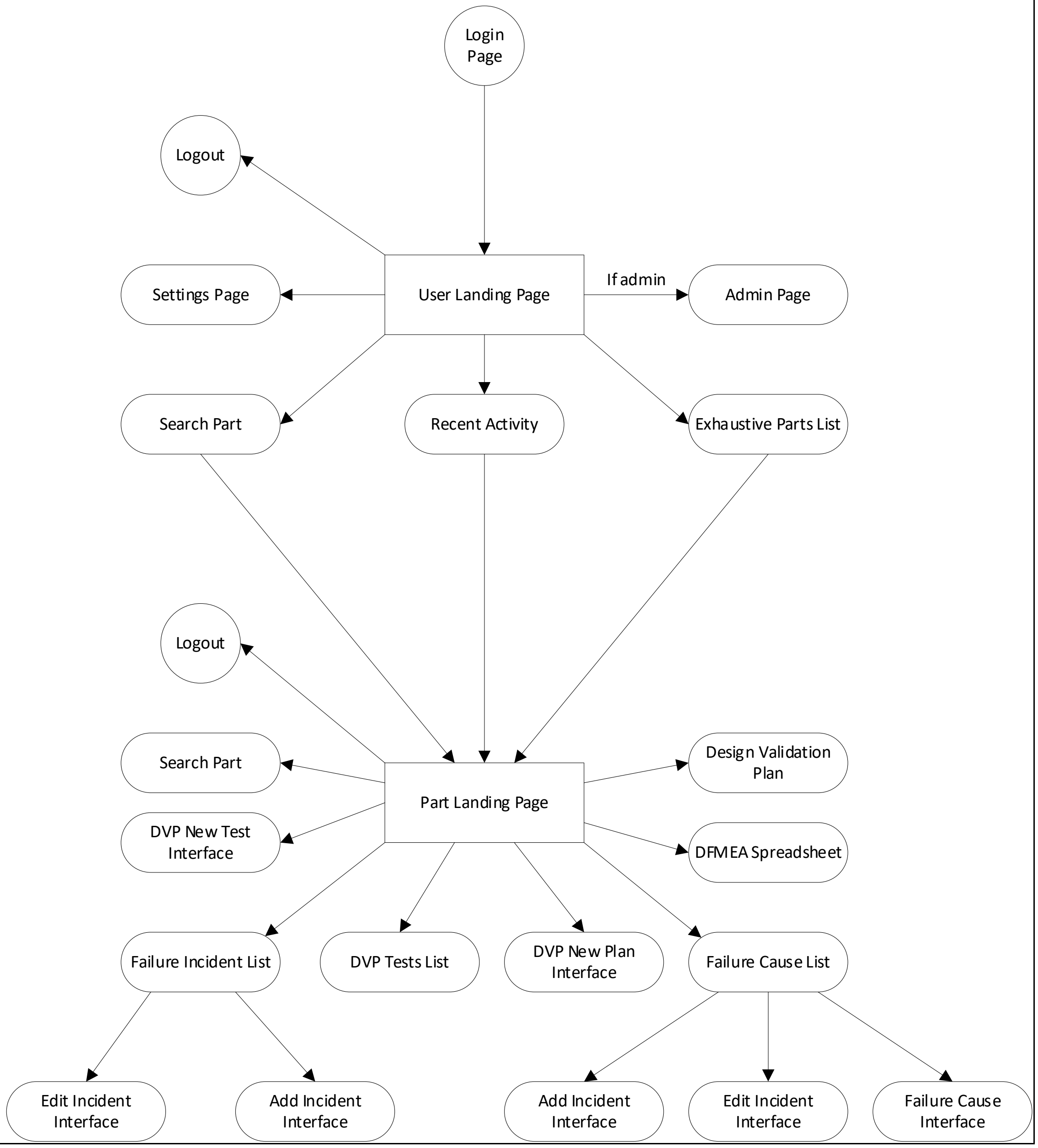
- The WFMS shall have a code base that is 100% portable and require no additional software aside from the web server itself.

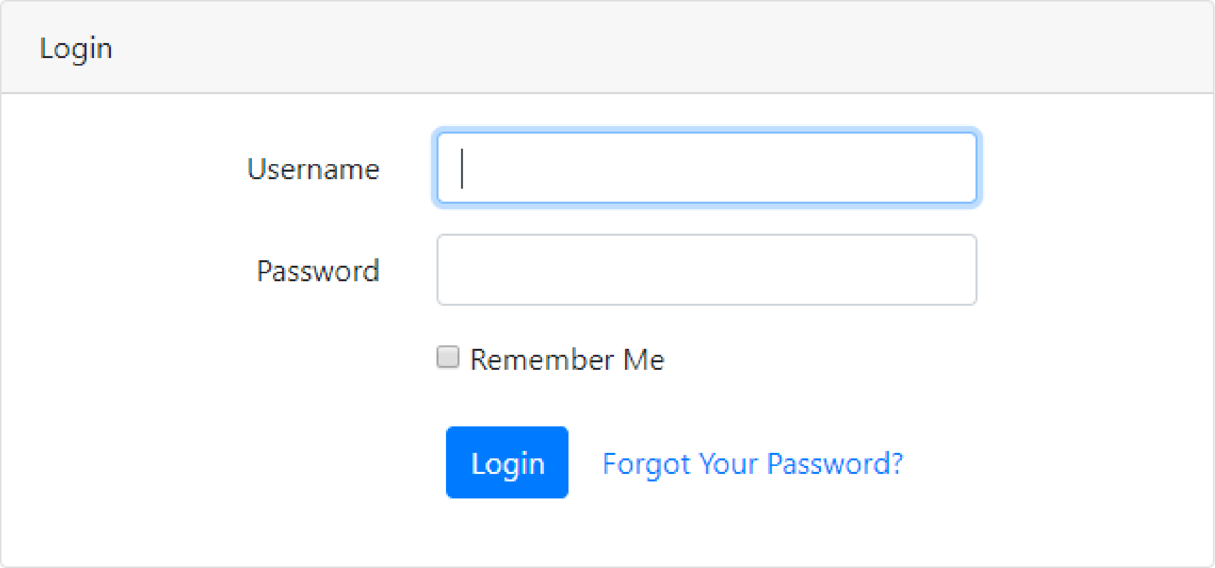
Appendices

Appendix 1.0 Storyboards

Story board flowchart

The flowchart provides a visual representation of how a user will be able to navigate the WFMS. There are two central hubs that allow for quick access to various features that a user may often request.

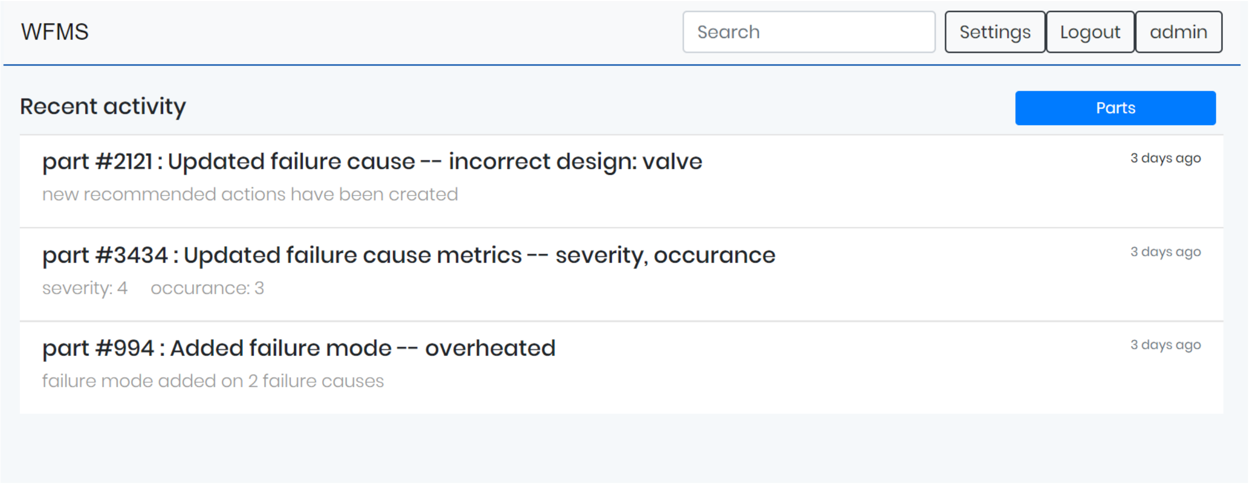




Login Page (LP) – All users that exist in the WFMS will be greeted with the login page when they attempt to access the WFMS. If a valid username and password are entered the WFMS will show the welcome page. Entering an invalid username and/or password will result in the system returning an error message prompting the user to try again. Clicking on the “forgot your password” link will prompt the user to enter an email address and will ask the user to check his/her email after submitting the form. The user will be given a new password in the email to use.

First time users will be prompted with the same interface except it will have an additional field below the password field and will be labeled “confirm password”. A first-time user will create and confirm a new password that is at least eight characters long.

characters long.



Recent activity

part #2121 : Updated failure cause -- incorrect design: valve 3 days ago
new recommended actions have been created

part #3434 : Updated failure cause metrics -- severity, occurance 3 days ago
severity: 4 occurance: 3

part #994 : Added failure mode -- overheated 3 days ago
failure mode added on 2 failure causes

User Landing Page (ULP) – On successful login, the user will see a screen containing a list of their recent activities in the WFMS. The list includes user-specific updates made against the WFMS database made up to 8 days prior. Clicking on a recent activity will direct the user to the WFMS part landing page (PLP).

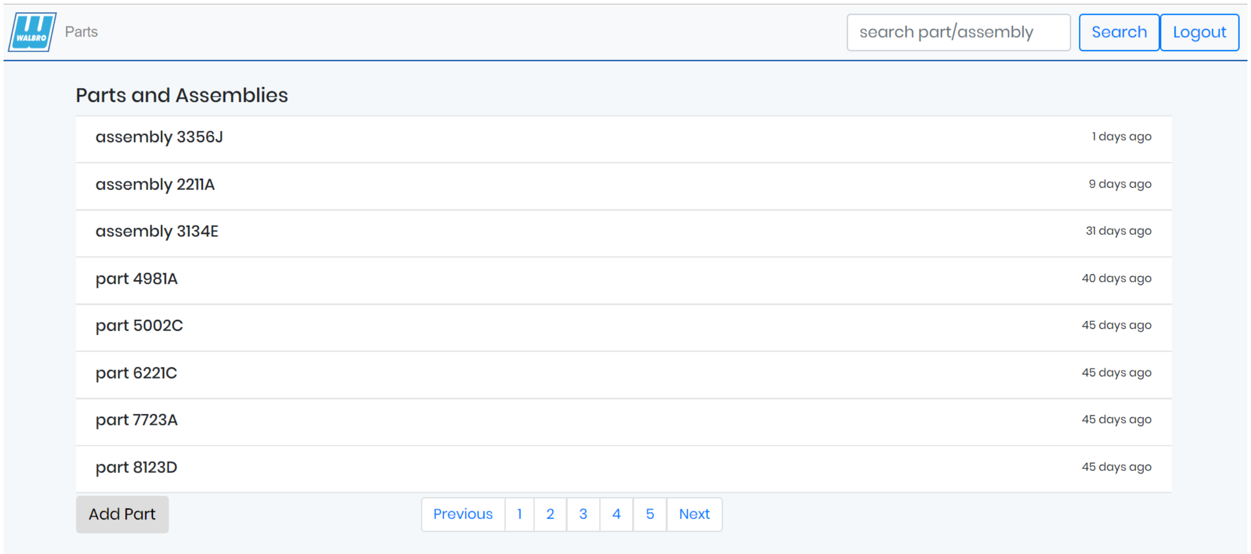
The Parts button will take the user to another page which displays a list of all parts and assemblies contained within the WFMS database.

The Search Bar allows a user to quickly navigate to a part-specific (or assembly-specific) landing page (PLP) by entering a part number and pressing the enter key.

The Settings button will allow a user to view and edit their account, permission, and role settings.

The Logout button will exit the WFMS and return the user to the login page.

The Admin button (only visible for WFMS administrators) will allow the execution of activities which are role-specific to the WFMS administrators. (I.e. Add new user, deactivate user, etc.)



Parts and Assemblies

assembly 3356J	1 days ago
assembly 2211A	9 days ago
assembly 3134E	31 days ago
part 4981A	40 days ago
part 5002C	45 days ago
part 6221C	45 days ago
part 7723A	45 days ago
part 8123D	45 days ago

Exhaustive Parts List (EPL) – Allows users to view an exhaustive list of all parts and assemblies within the WFMS database (chronologically ordered). Pagination allows for the user to quickly navigate between pages in the list. Clicking on a part or assembly in the list will direct the user to the parts landing page (PLP) for the specified part.

The Add Part button allows a user to insert a new part into the list of parts.

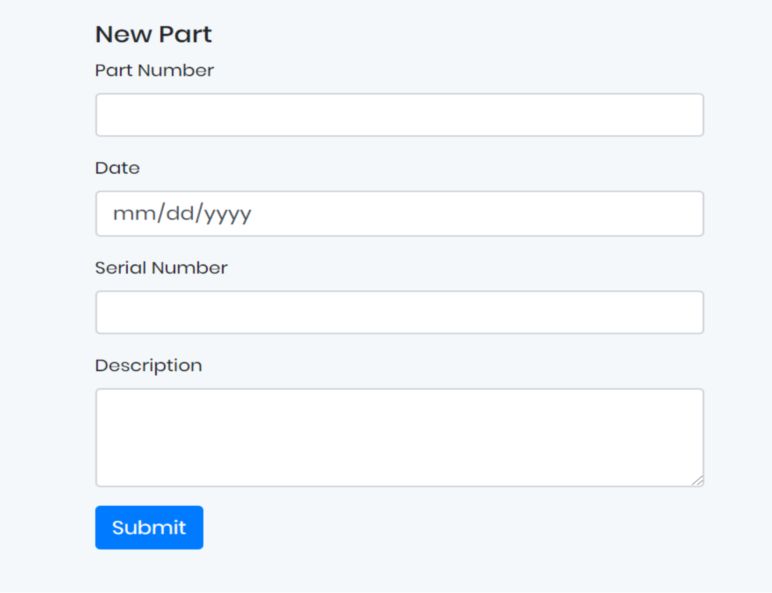
The Previous button will display the previous page in the list of parts.

The Next button will display the next page in the list of parts.

The Numeric (enumerated) buttons will update their values and navigational destination based upon the current position within list of parts. The middle button will display the current page number, with the two buttons on either side of it being the previous and or next two pages.

The Logout button will exit the WFMS and return the user to the login page.

The Search Bar allows a user to quickly navigate to a part-specific (or assembly-specific) landing page (PLP) by entering a part number and pressing the enter key.



New Part

Part Number

Date

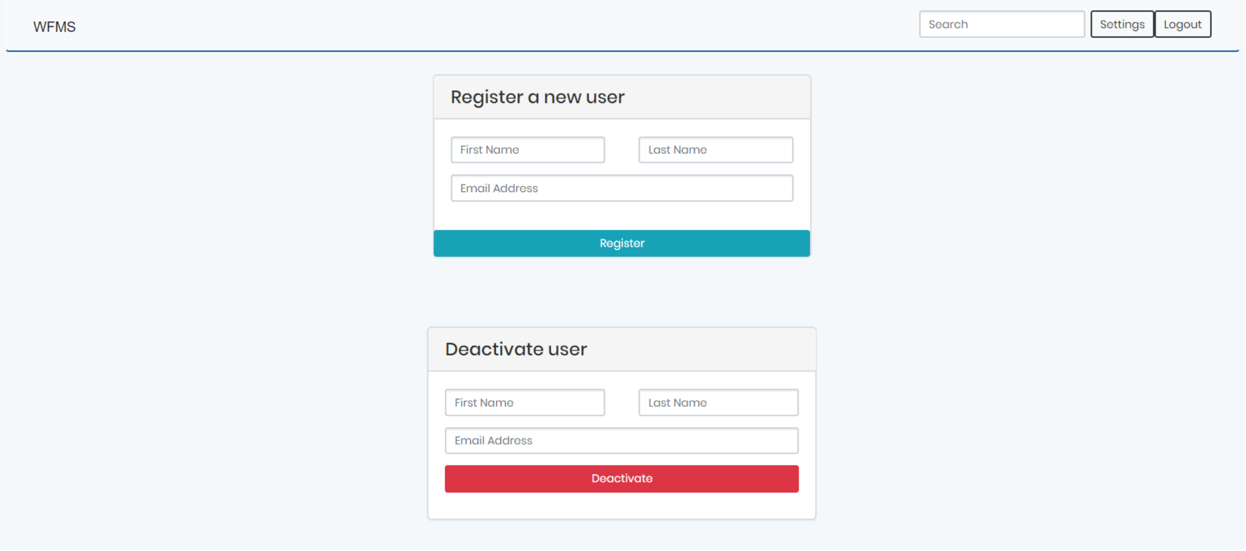
mm/dd/yyyy

Serial Number

Description

New Part Interface (EPL) – Displayed after a user clicks the Add Part button from the EPL screen. From here, a user may complete the form and click the submit button to add a new part into the EPL database. Assuming valid data entry and a successful insertion, the user will be directed back to the EPL screen.

The Submit button will add a new part to the EPL (assuming valid data entry) and return the user to the EPL.

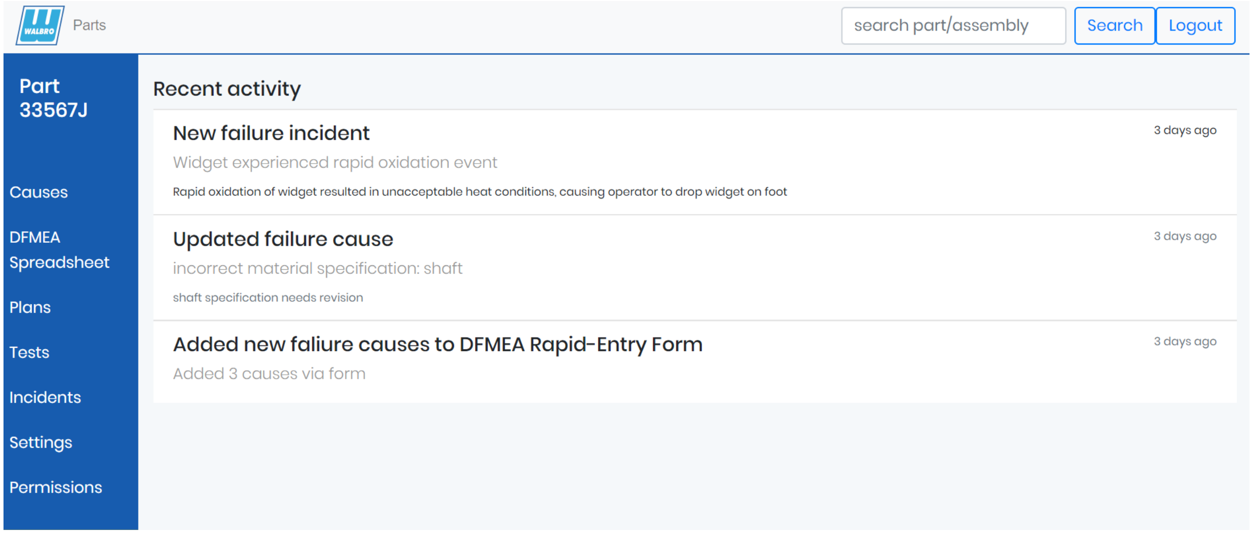


System Administrator (SA) – An administrator can register a new user or deactivate an existing user from the system administrator page. This page is unique to WFMS administrators and accessible from the user landing page (ULP). Filling out the registration form will send an email invitation to the new user's Walbro email account. The email will include a link which takes the user to the login page allowing them to create a password.

The deactivate user form will archive a user account. The account remains viewable by WFMS administrators.

The Register button will send an email invitation to the entered user.

The Deactivate button will disable an existing account's ability to access the WFMS system but will not delete the account.



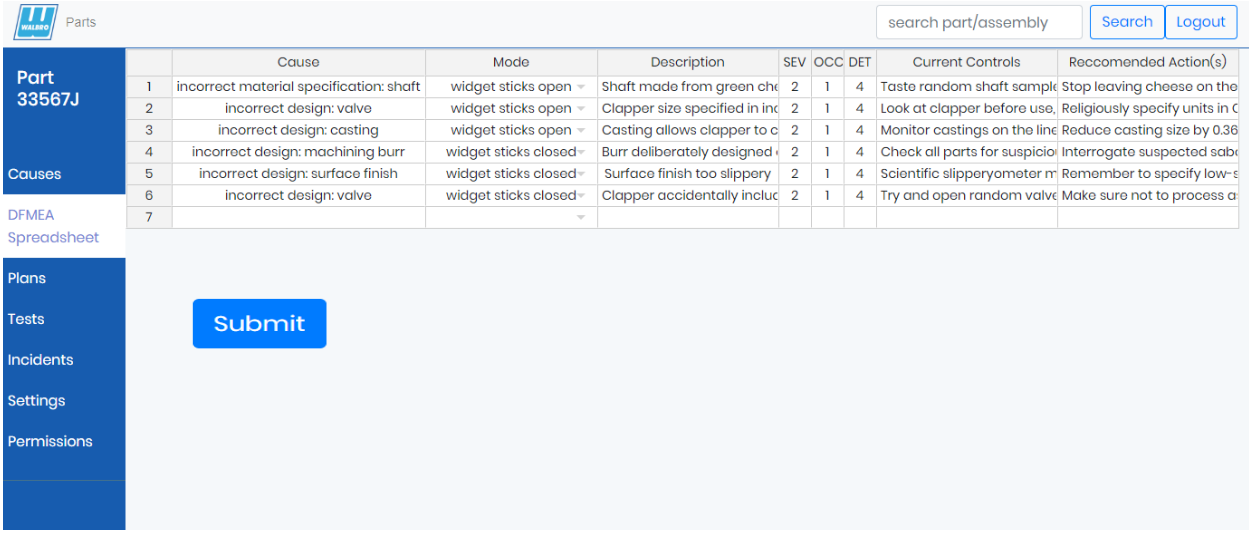
Recent activity

New failure incident
3 days ago
Widget experienced rapid oxidation event
Rapid oxidation of widget resulted in unacceptable heat conditions, causing operator to drop widget on foot

Updated failure cause
3 days ago
incorrect material specification: shaft
shaft specification needs revision

Added new faliure causes to DFMEA Rapid-Entry Form
3 days ago
Added 3 causes via form

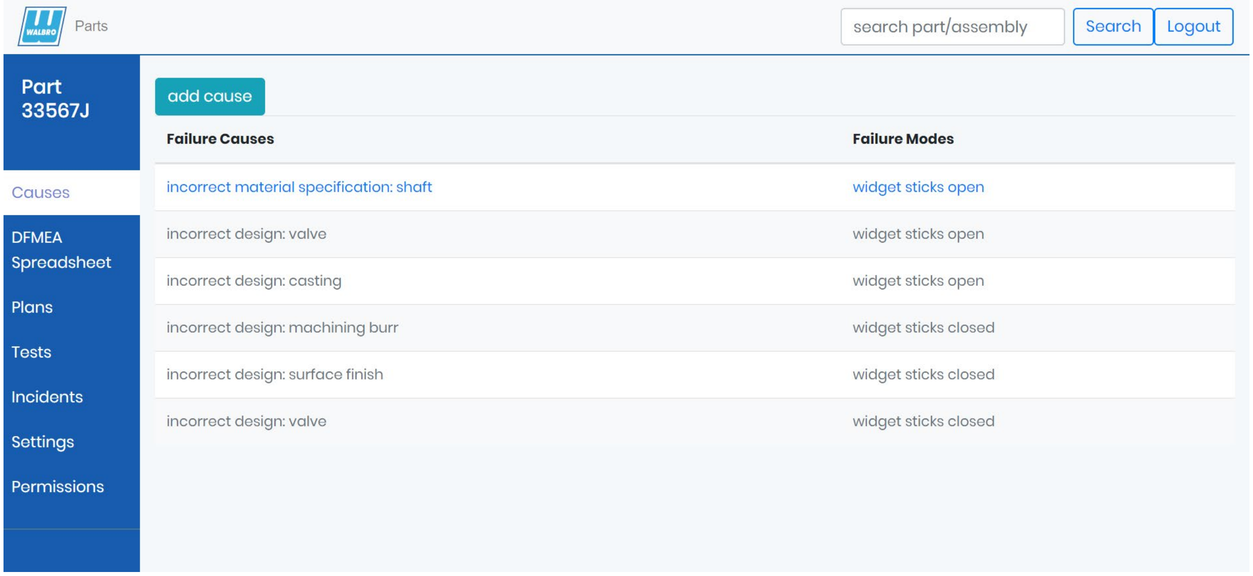
Parts Landing Page (PLP) – Displayed after clicking a part in the ULP part activity list, clicking a part in the EPL, or searching for a part in the global search bar. The global search bar appears at the top of most pages within the WFMS, allowing navigation from most pages in the WFMS directly to an updated PLP. An updated PLP displays all activity made against a part in the WFMS database.



	Cause	Mode	Description	SEV	OCC	DET	Current Controls	Reccomended Action(s)
1	incorrect material specification: shaft	widget sticks open	Shaft made from green che	2	1	4	Taste random shaft sample	Stop leaving cheese on the
2	incorrect design: valve	widget sticks open	Clapper size specified in inc	2	1	4	Look at clapper before use,	Religiously specify units in C
3	incorrect design: casting	widget sticks open	Casting allows clapper to c	2	1	4	Monitor castings on the line	Reduce casting size by 0.36
4	incorrect design: machining burr	widget sticks closed	Burr deliberately designed	2	1	4	Check all parts for suspicio	Interrogate suspected sab
5	incorrect design: surface finish	widget sticks closed	Surface finish too slippery	2	1	4	Scientific slipperyometer m	Remember to specify low-s
6	incorrect design: valve	widget sticks closed	Clapper accidentally incluc	2	1	4	Try and open random valve	Make sure not to process a
7								

DFMEA spreadsheet (DFMEA) – Allows rapid data entry of multiple failure cause criteria for a part in a single session. A user can quickly navigate between cells and add new rows via the tab and enter keys. Hovering the cursor over the severity, occurrence, and/or detection inputs will display a test-specific metric for the keyed integer value. If a user attempts to navigate away from the spreadsheet with multiple rows of unsaved data, then the WFMS will prompt the user to help prevent unintended data loss.

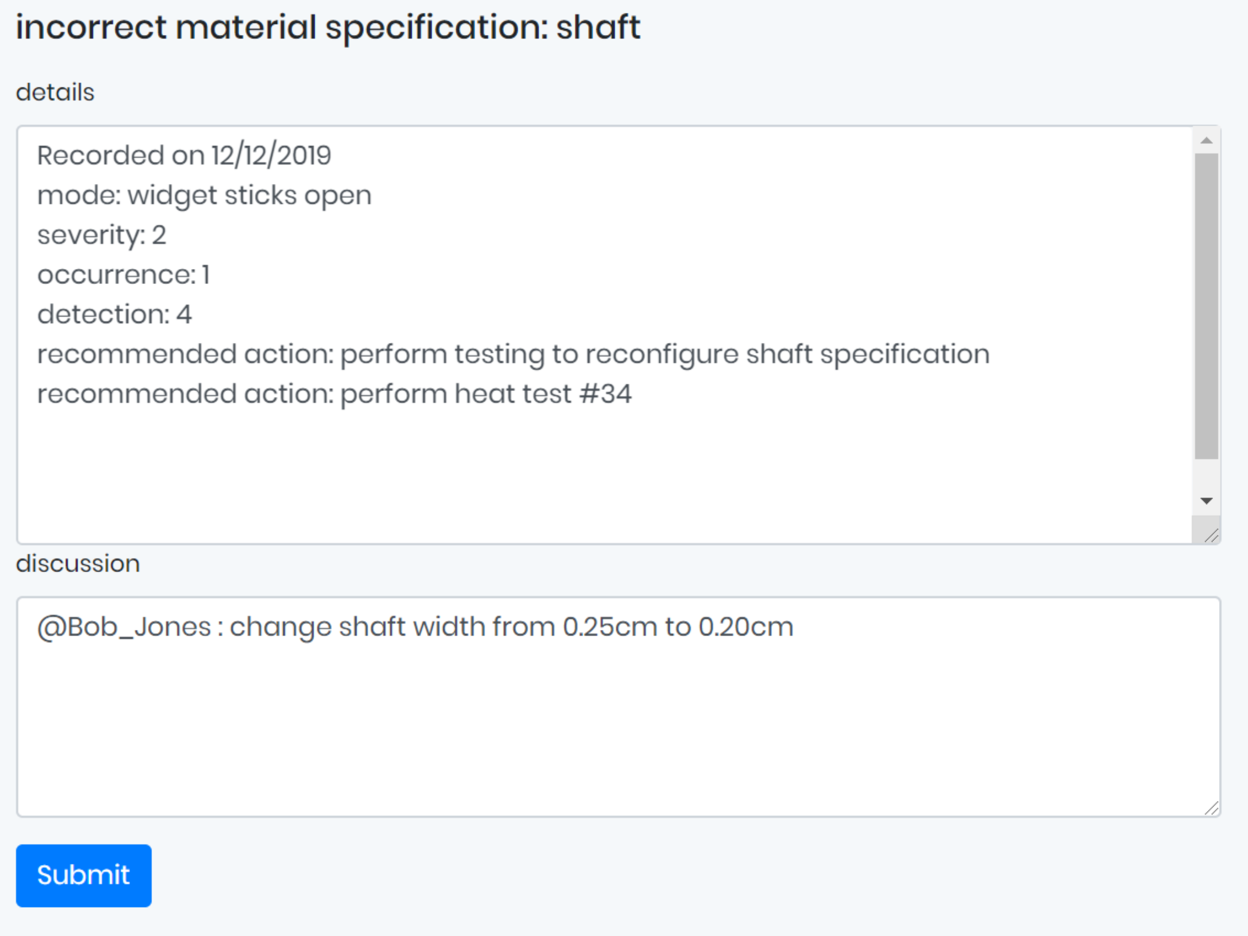
The Submit button will commit any changes to the WFMS database.



Failure Causes	Failure Modes
incorrect material specification: shaft	widget sticks open
incorrect design: valve	widget sticks open
incorrect design: casting	widget sticks open
incorrect design: machining burr	widget sticks closed
incorrect design: surface finish	widget sticks closed
incorrect design: valve	widget sticks closed

Failure Cause List (FCL) – Offers an organized and exhaustive list of all failure causes associated with a specified part. Clicking a failure cause will open a view with links to further documentation, discussion, and attached files for the failure cause. A user may edit all links and files within the extended view. The list will only display the causes associated with the selected failure mode. Button clicks will navigate to their interface and then bring the user back to the FCL page when complete.

The Add Cause button will add a new failure cause to the selected part and failure mode.



incorrect material specification: shaft

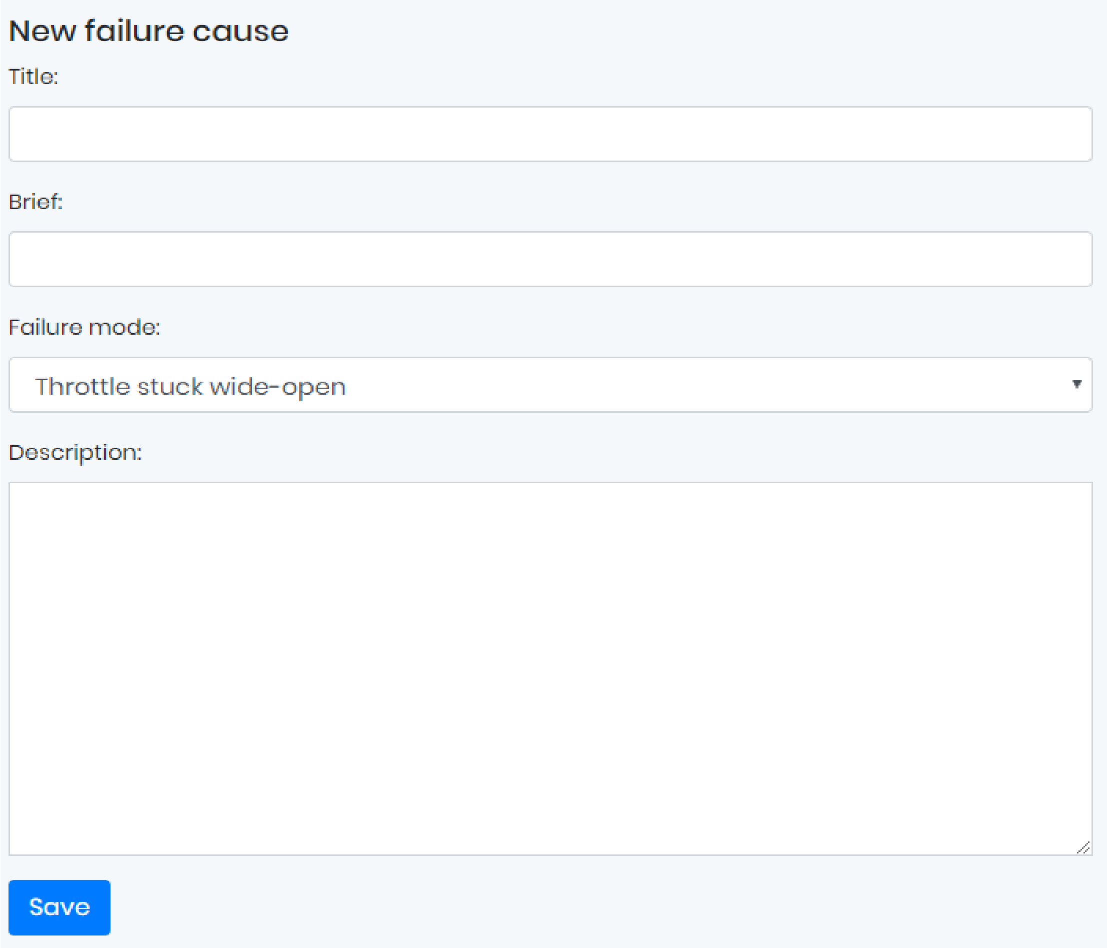
details

Recorded on 12/12/2019
mode: widget sticks open
severity: 2
occurrence: 1
detection: 4
recommended action: perform testing to reconfigure shaft specification
recommended action: perform heat test #34

discussion

@Bob_Jones : change shaft width from 0.25cm to 0.20cm

Failure Cause View/Update Form (FCL) – This form is displayed when the user clicks on a failure cause from the failure cause list view. When clicked, the failure cause list view loses focus and only this form can be accessed. The user has the capability of viewing and/or updating any details or discussions that are necessary. Clicking submit will return the user to the failure cause list view.



New failure cause

Title:

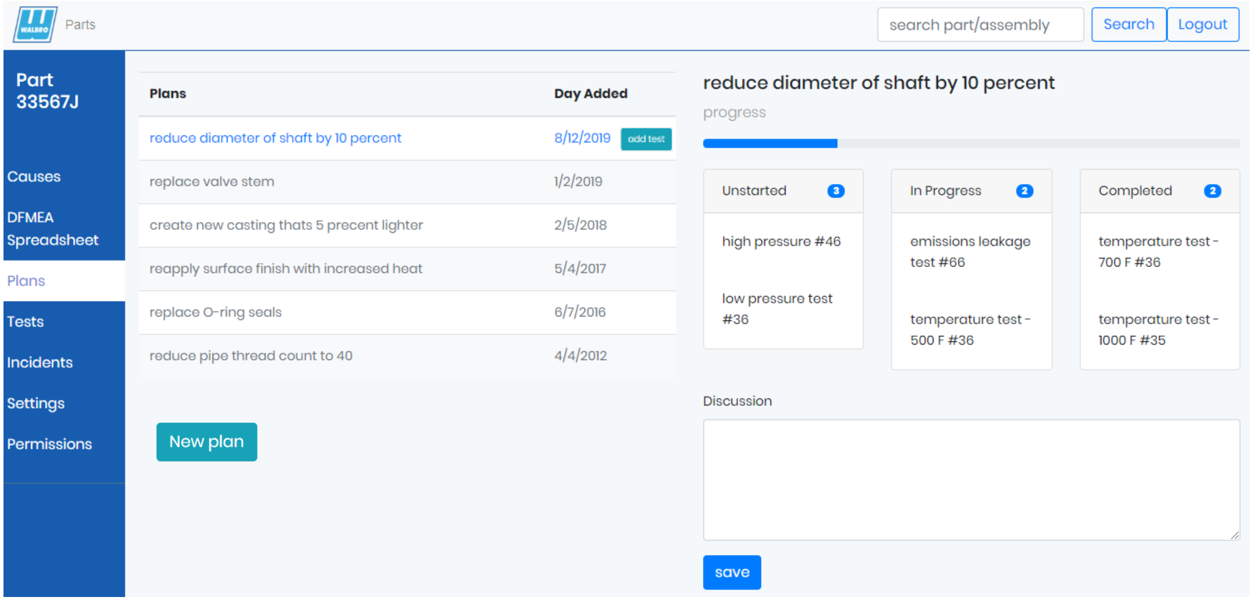
Brief:

Failure mode: Throttle stuck wide-open

Description:

Failure Cause Interface (FCI) – Accessed by clicking Add Cause from the FCL screen. Allows the failure mode data to be pre-populated from the FCL screen. This screen will prompt a user if a save is attempted while any required data fields (title, failure mode) are invalid/empty. After a successful save, the user will be returned to the FCL screen.

The Save button will attempt to save the data entered in the FCI to the associated failure mode.



Plans	Day Added
reduce diameter of shaft by 10 percent	8/12/2019
replace valve stem	1/2/2019
create new casting thats 5 precent lighter	2/5/2018
reapply surface finish with increased heat	5/4/2017
replace O-ring seals	6/7/2016
reduce pipe thread count to 40	4/4/2012

reduce diameter of shaft by 10 percent

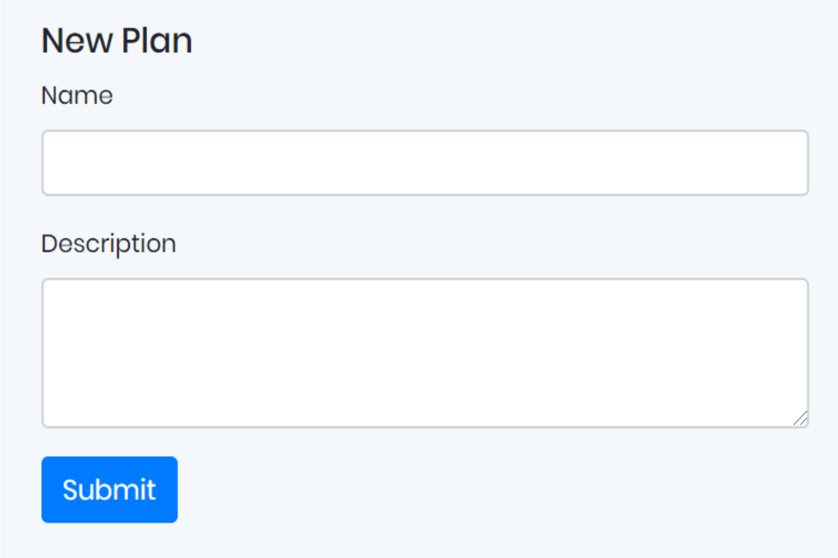
Unstarted 3	In Progress 2	Completed 2
high pressure #46	emissions leakage test #66	temperature test - 700 F #36
low pressure test #36	temperature test - 500 F #36	temperature test - 1000 F #35

Design Validation Plan (DVP) –A DVP shows all plans and tests associated with a specified part. A selected plan's overall progression and associated tests will be displayed on the right side of the screen. Tests associated with a plan exist in one of three states: "Unstarted", "In Progress", or "Completed". Each state is represented by a bucket on the DVP page and tests may be dragged and dropped between states to represent progress. A discussion box associated with the selected plan is displayed below the tests section. The discussion box allows users to document details about a plan or test. The discussion text field also allows users to tag other users (@ + username) which triggers an automated email message sent to the tagged user.

The Save button will attempt to save the data entered in the discussion text field.

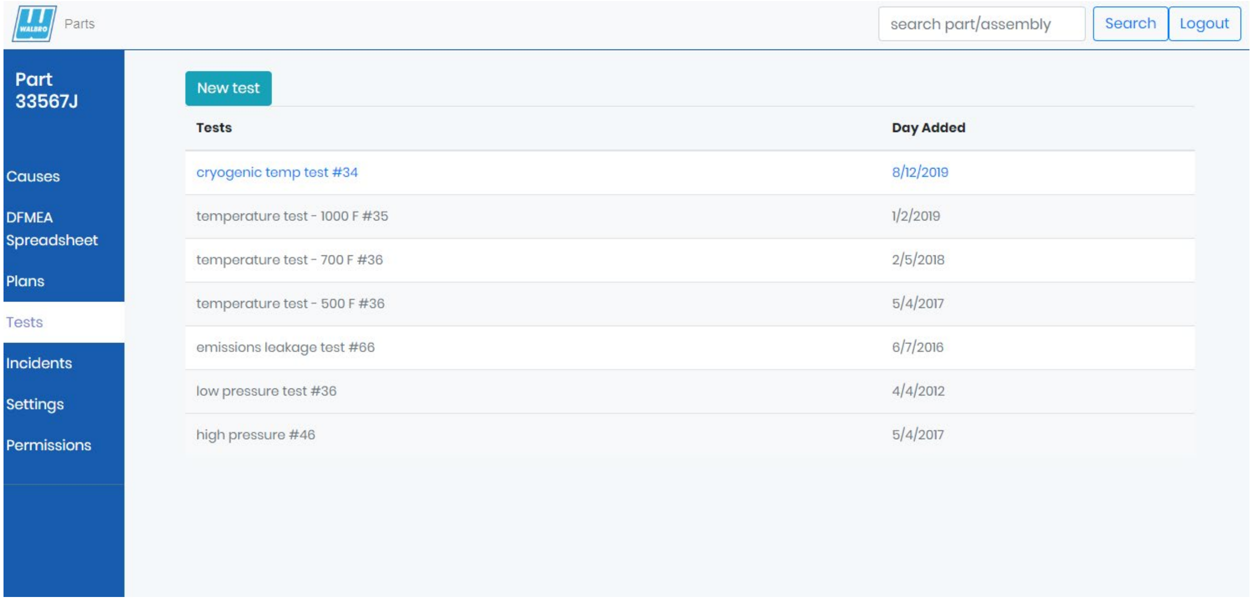
The New Plan button will allow a user to add a new DVP to an existing part via the new plan interface.

The Add Test button will allow a user to add an existing test or create a new test for a specific plan.



DVP New Plan Interface (DVPPI) – Allows a user to create a new plan. After a new plan is submitted, the user will return to an updated DVP page. In the event of a mis-click or accidental navigation to this page, empty plans will be filtered and not associated with any part.

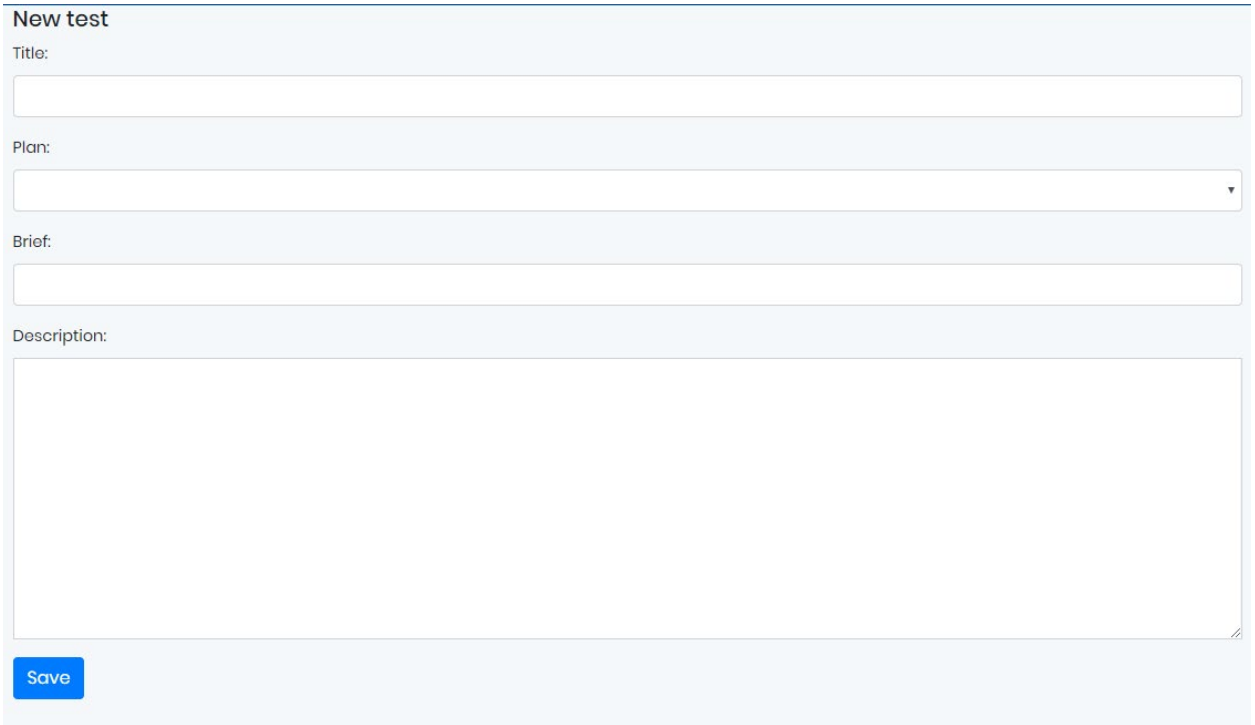
The Submit button will attempt to save the plan and create an association with specified part.



Tests	Day Added
cryogenic temp test #34	8/12/2019
temperature test - 1000 F #35	1/2/2019
temperature test - 700 F #36	2/5/2018
temperature test - 500 F #36	5/4/2017
emissions leakage test #66	6/7/2016
low pressure test #36	4/4/2012
high pressure #46	5/4/2017

Tests List (TL) – Displays a chronologically ordered list of all tests for a specified part. Clicking a test will show the following: an associated plan (if it has one), details and discussion about the test, and media related to the test.

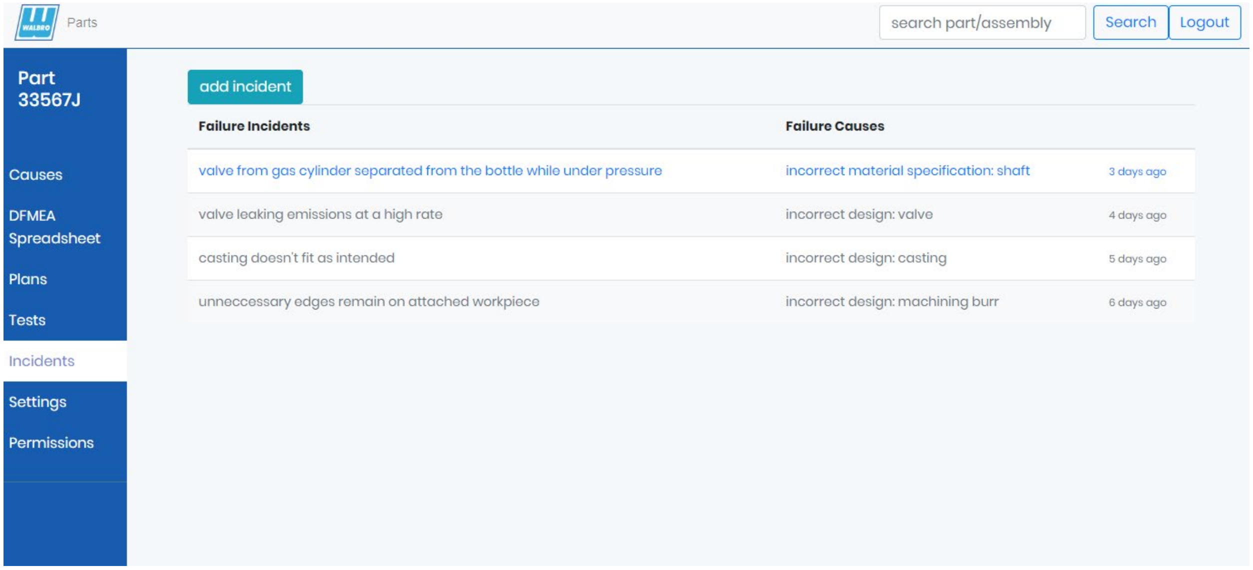
The New Test button will create a new test and associate the test with the specified plan (via DVPTI).



New Test Interface (TI) – The form on the new test page allows the user to create a new test, provides an option to assign it to a plan, and give brief details and/or a description. Once the new test is saved it will be prepended to the list of tests on the previous page. The new test will also be stored in the "Unstarted" column for the plan in which it was assigned. Finally, saving the completed form will return the user to the tests page.

When accessed by the "add test" button from the DVP, the form will allow for a user to enter an existing test or a new test. Entering an existing test will be aided with auto-complete and will mask all fields following the title. Creating a new test will mask only the plan field (since this form was accessed from a specific plan).

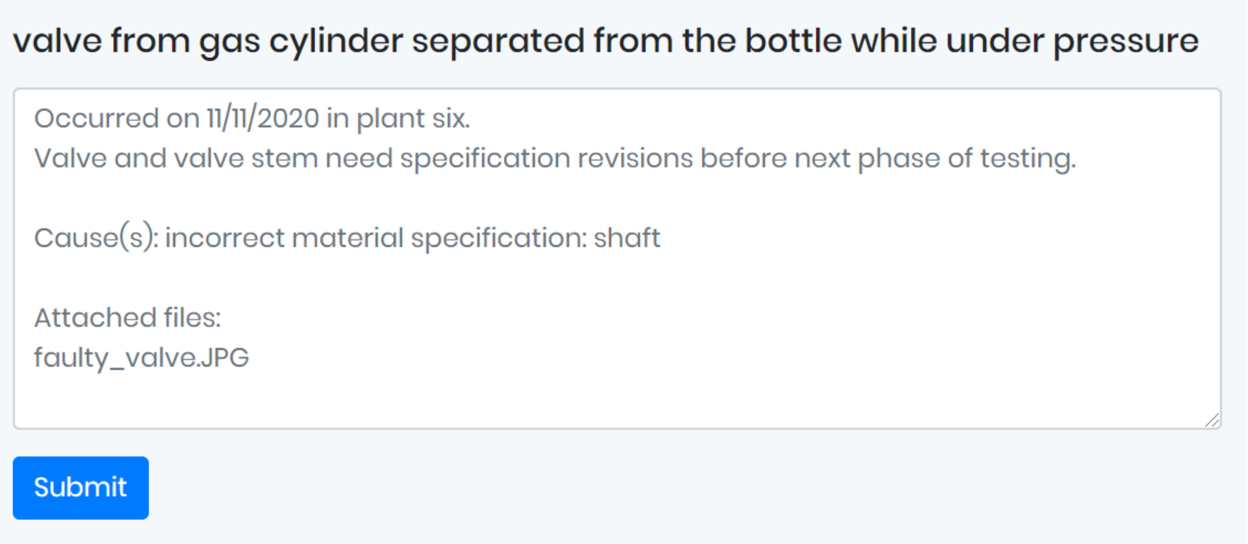
The Save button will attempt to save the new test and create an association with the specified plan.



Failure Incidents	Failure Causes	
valve from gas cylinder separated from the bottle while under pressure	incorrect material specification: shaft	3 days ago
valve leaking emissions at a high rate	incorrect design: valve	4 days ago
casting doesn't fit as intended	incorrect design: casting	5 days ago
unneccessary edges remain on attached workpiece	incorrect design: machining burr	6 days ago

Failure Incident List (FIL) – Found in the Failures dropdown menu within the parts sidebar. Allows a user to add, edit, and view details about failure incidents of an associated failure cause.

The Add Incident button will add a new failure incident in a new screen (via FIL-A).

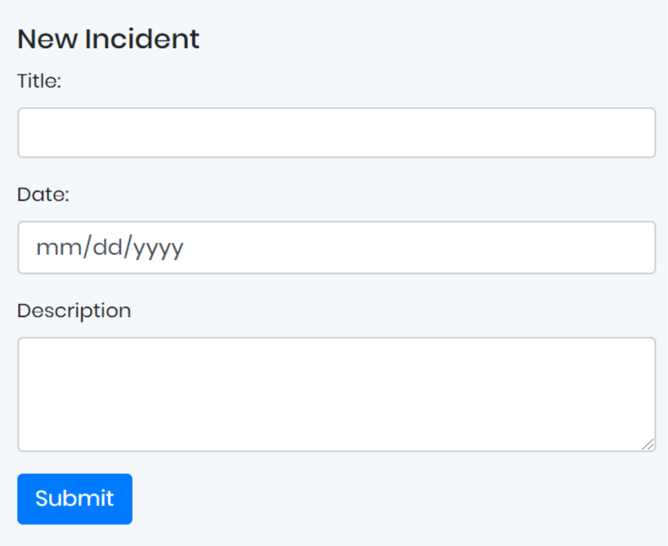

valve from gas cylinder separated from the bottle while under pressure

Occurred on 11/11/2020 in plant six.
Valve and valve stem need specification revisions before next phase of testing.

Cause(s): incorrect material specification: shaft

Attached files:
faulty_valve.JPG

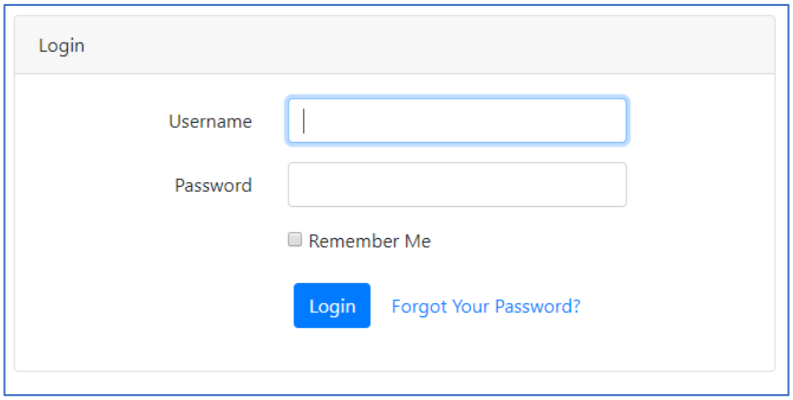
Edit Incident Interface (FIL) – This appears when a user clicks on an incident from the previous page. When this form is accessed, the failure view list page loses focus and nothing besides this form is accessible. Edits can be made before clicking submit to return to the previous page.

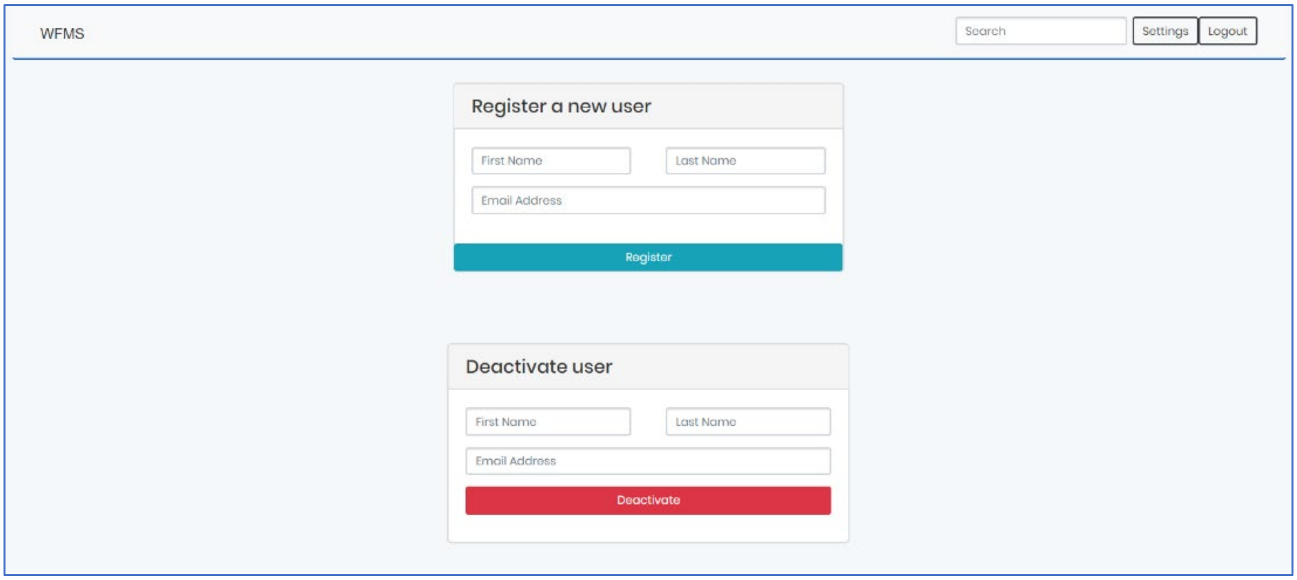


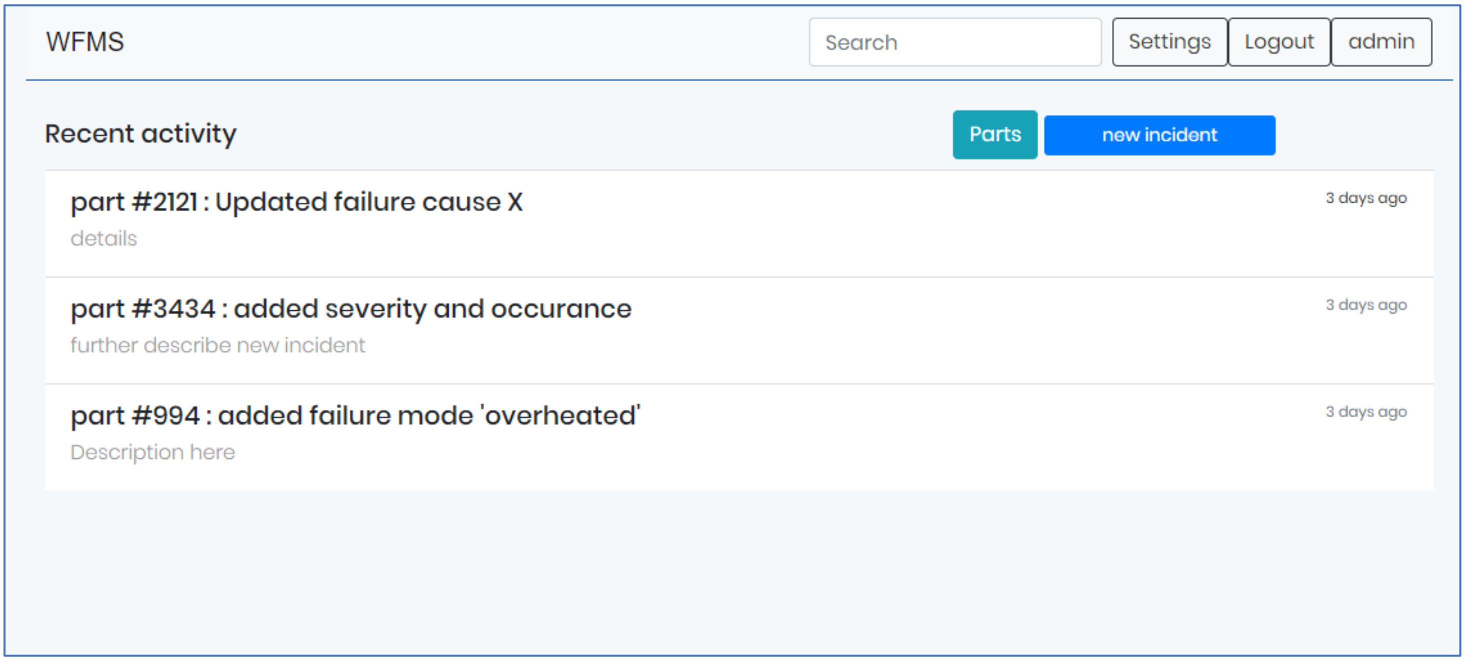
Failure Incident Interface (FII) – Visible after a user clicks the Add Incident button on the FIL page. A user may add data within the text fields. When finished, the user will be taken back to an updated FIL screen.

The Submit button will save edits made within both text fields.

Appendix 2.0 UI Group Screenshots





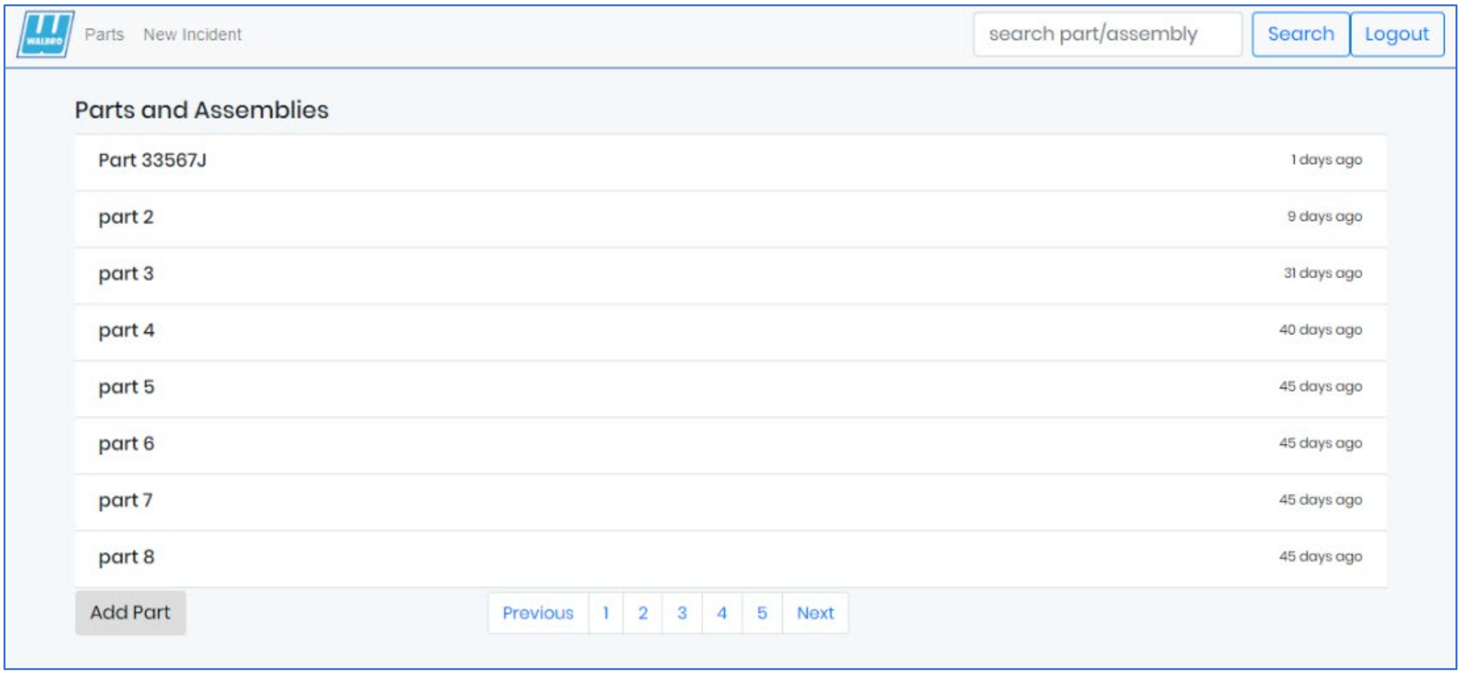


Recent activity

part #2121 : Updated failure cause X
details
3 days ago

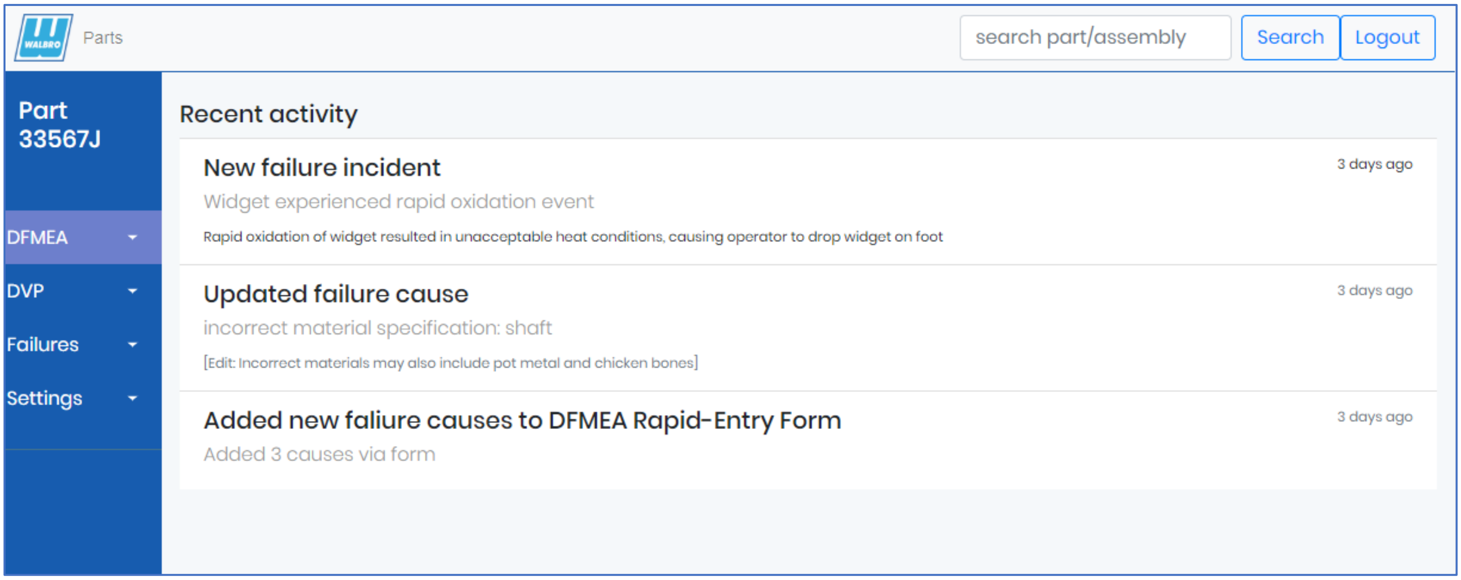
part #3434 : added severity and occurance
further describe new incident
3 days ago

part #994 : added failure mode 'overheated'
Description here
3 days ago



Parts and Assemblies

Part 33567J	1 days ago
part 2	9 days ago
part 3	31 days ago
part 4	40 days ago
part 5	45 days ago
part 6	45 days ago
part 7	45 days ago
part 8	45 days ago



Recent activity

New failure incident — 3 days ago

Widget experienced rapid oxidation event

Rapid oxidation of widget resulted in unacceptable heat conditions, causing operator to drop widget on foot

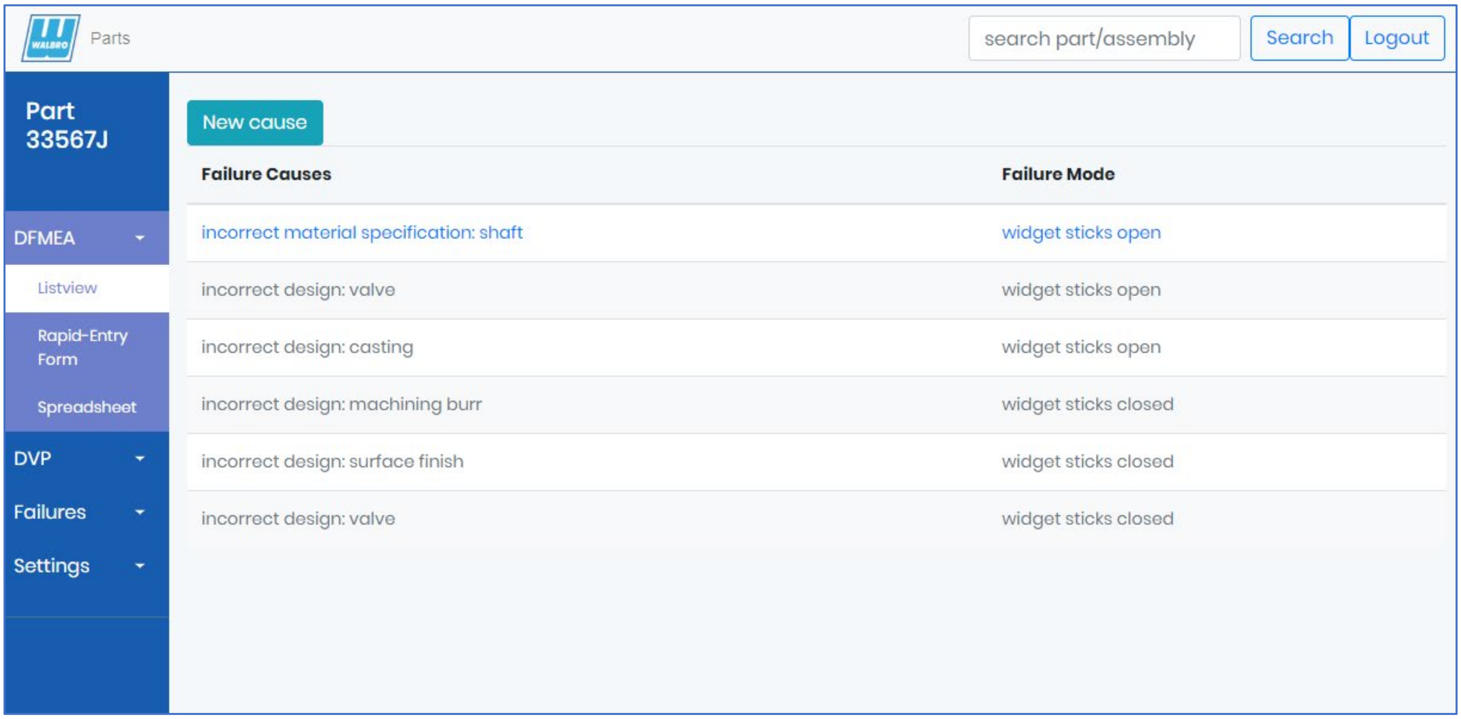
Updated failure cause — 3 days ago

incorrect material specification: shaft

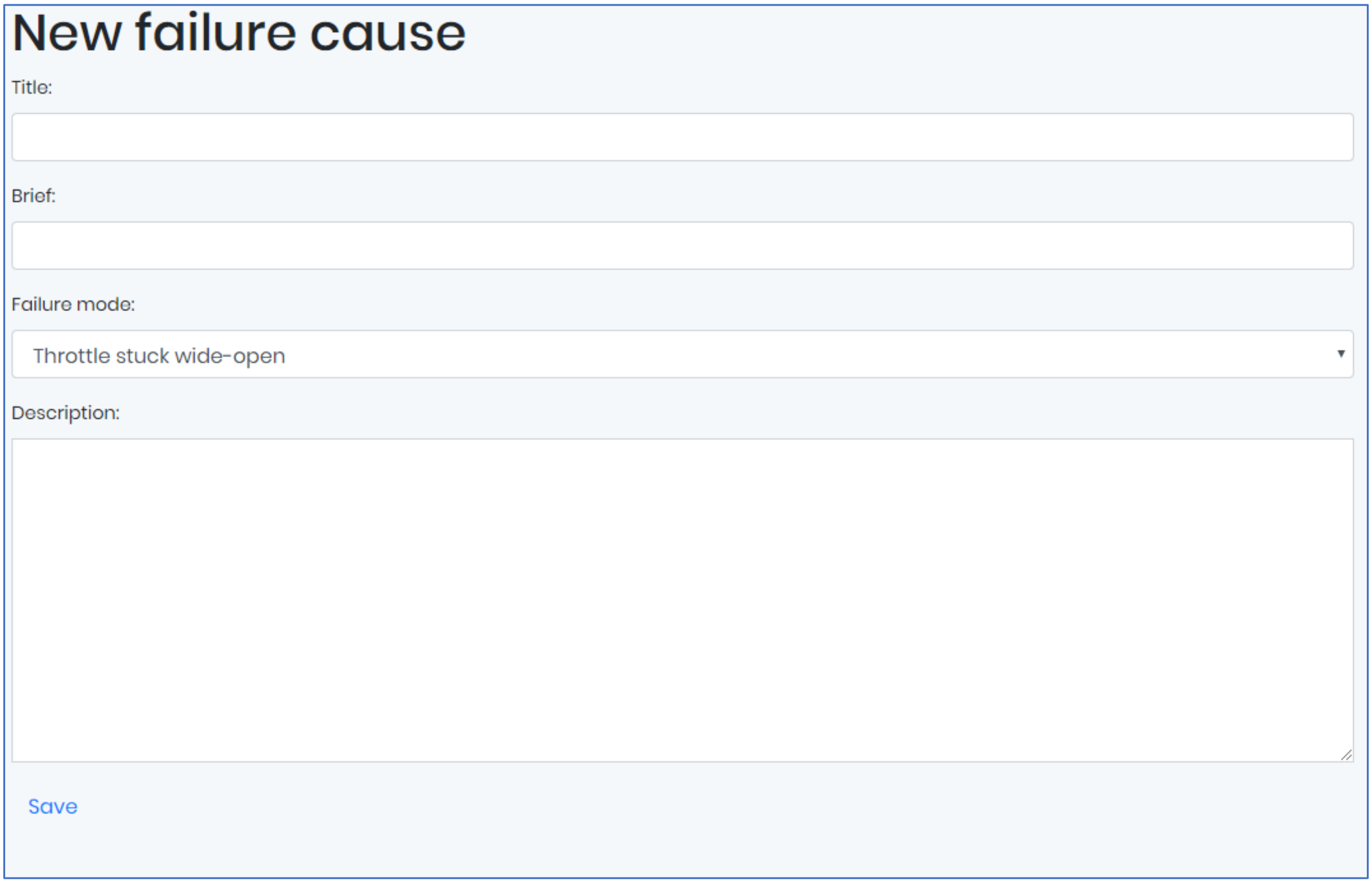
[Edit: Incorrect materials may also include pot metal and chicken bones]

Added new faliure causes to DFMEA Rapid-Entry Form — 3 days ago

Added 3 causes via form



Failure Causes	Failure Mode
incorrect material specification: shaft	widget sticks open
incorrect design: valve	widget sticks open
incorrect design: casting	widget sticks open
incorrect design: machining burr	widget sticks closed
incorrect design: surface finish	widget sticks closed
incorrect design: valve	widget sticks closed



New failure cause

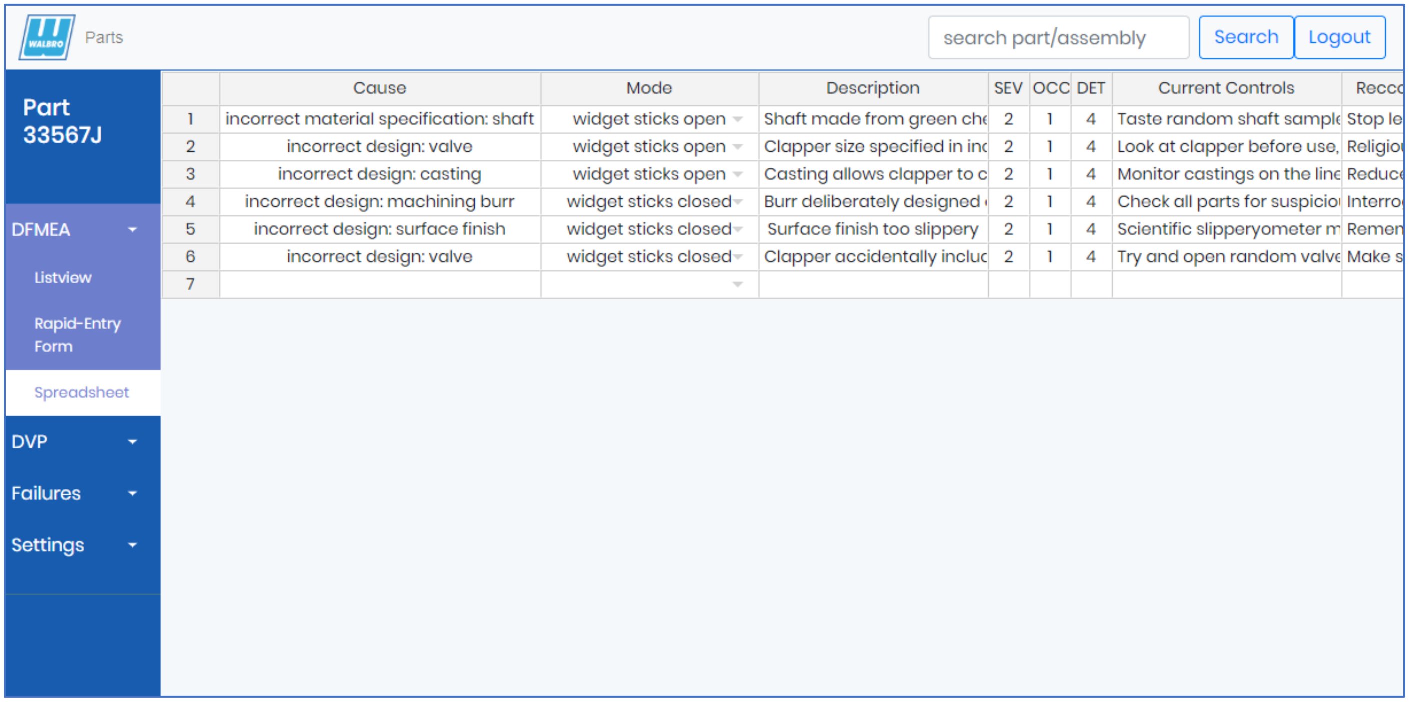
Title:

Brief:

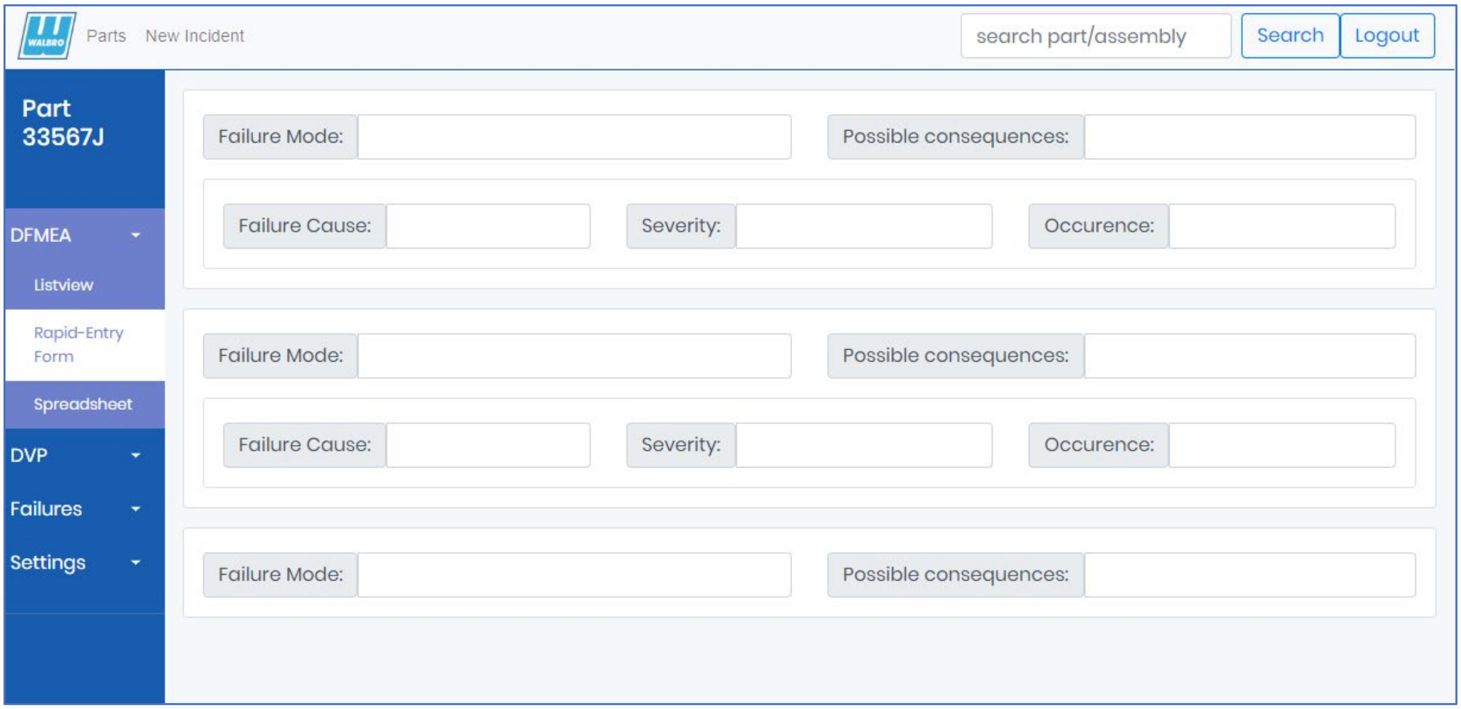
Failure mode:

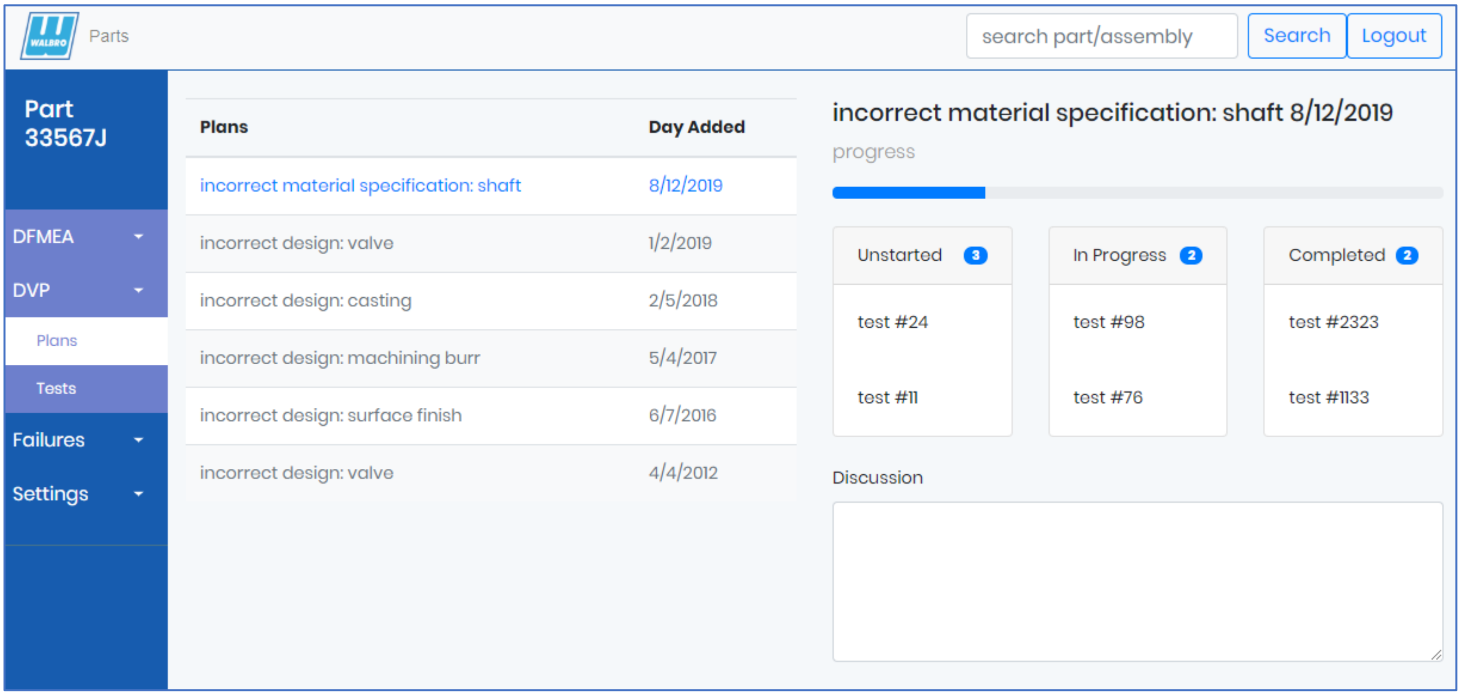
Throttle stuck wide-open ▾

Description:



	Cause	Mode	Description	SEV	OCC	DET	Current Controls	Recco
1	incorrect material specification: shaft	widget sticks open	Shaft made from green che	2	1	4	Taste random shaft sample	Stop le
2	incorrect design: valve	widget sticks open	Clapper size specified in inc	2	1	4	Look at clapper before use,	Religio
3	incorrect design: casting	widget sticks open	Casting allows clapper to c	2	1	4	Monitor castings on the line	Reduc
4	incorrect design: machining burr	widget sticks closed	Burr deliberately designed	2	1	4	Check all parts for suspicio	Interro
5	incorrect design: surface finish	widget sticks closed	Surface finish too slippery	2	1	4	Scientific slipperyometer m	Remen
6	incorrect design: valve	widget sticks closed	Clapper accidentally incluc	2	1	4	Try and open random valve	Make s
7								


WALBRO
Parts
New Incident
search part/assembly
Search
Logout
Part 33567J
DFMEA
Listview
Rapid-Entry Form
Spreadsheet
DVP
Failures
Settings
Failure Mode:
Possible consequences:
Failure Cause:
Severity:
Occurence:
Failure Mode:
Possible consequences:
Failure Cause:
Severity:
Occurence:
Failure Mode:
Possible consequences:



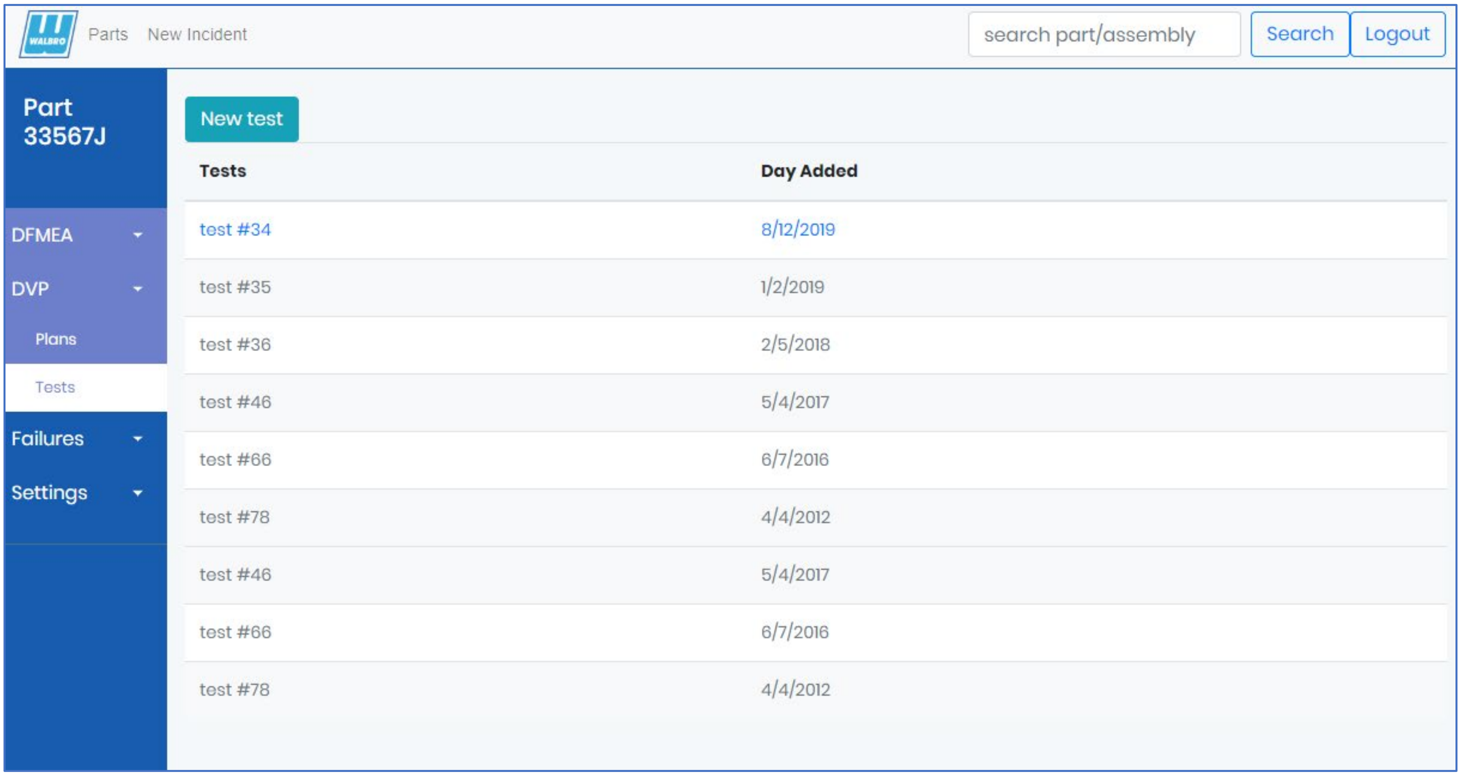
Plans	Day Added
incorrect material specification: shaft	8/12/2019
incorrect design: valve	1/2/2019
incorrect design: casting	2/5/2018
incorrect design: machining burr	5/4/2017
incorrect design: surface finish	6/7/2016
incorrect design: valve	4/4/2012

incorrect material specification: shaft 8/12/2019

progress

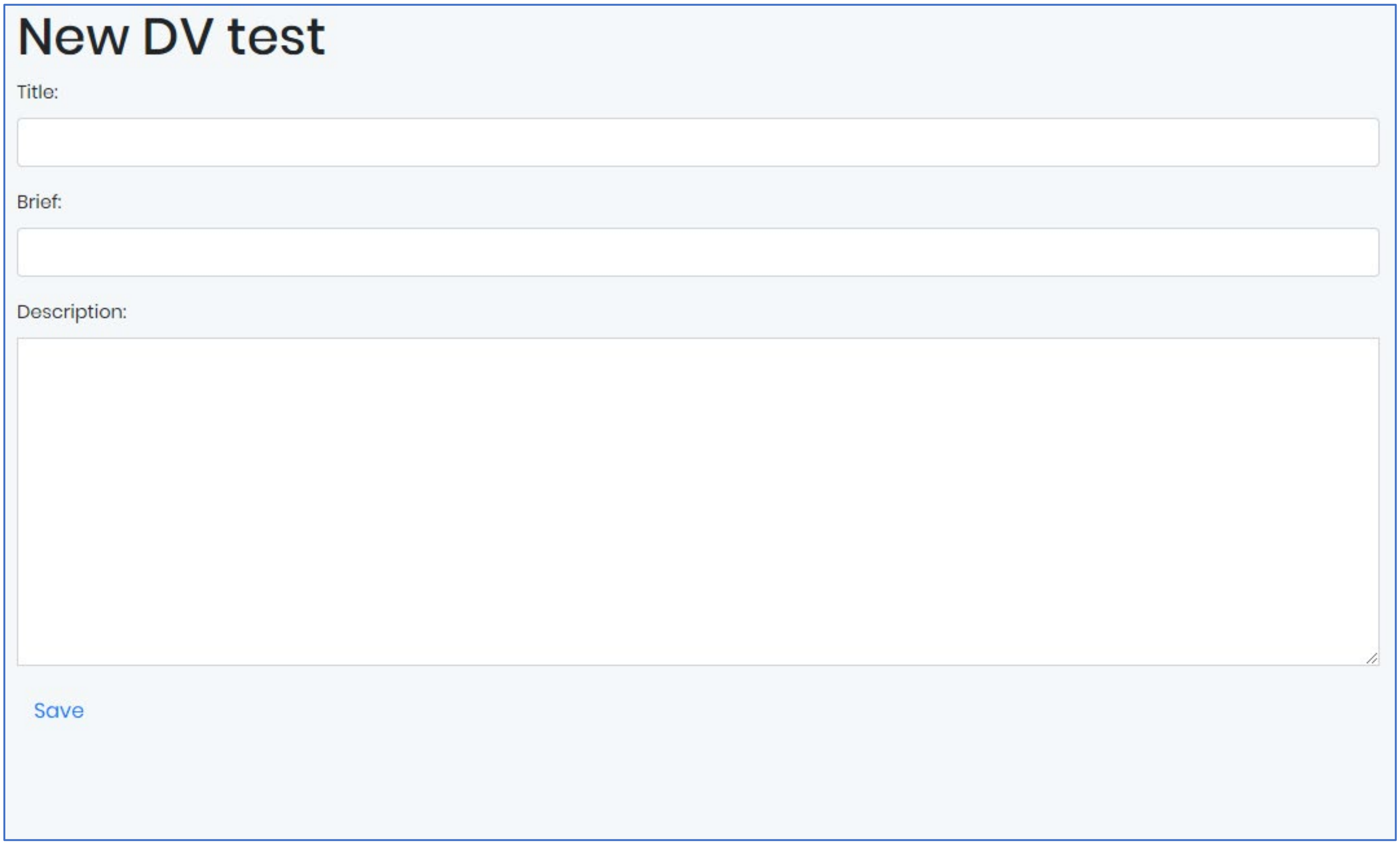
Unstarted 3	In Progress 2	Completed 2
test #24	test #98	test #2323
test #11	test #76	test #1133

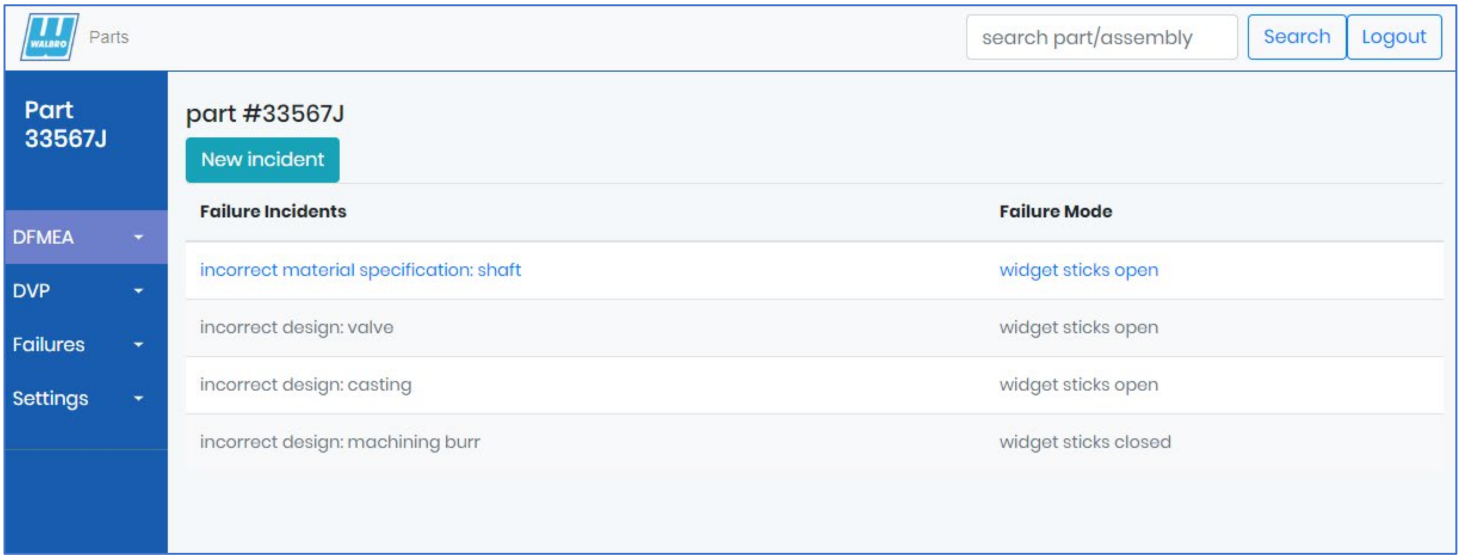
Discussion



Tests	Day Added
test #34	8/12/2019
test #35	1/2/2019
test #36	2/5/2018
test #46	5/4/2017
test #66	6/7/2016
test #78	4/4/2012
test #46	5/4/2017
test #66	6/7/2016
test #78	4/4/2012

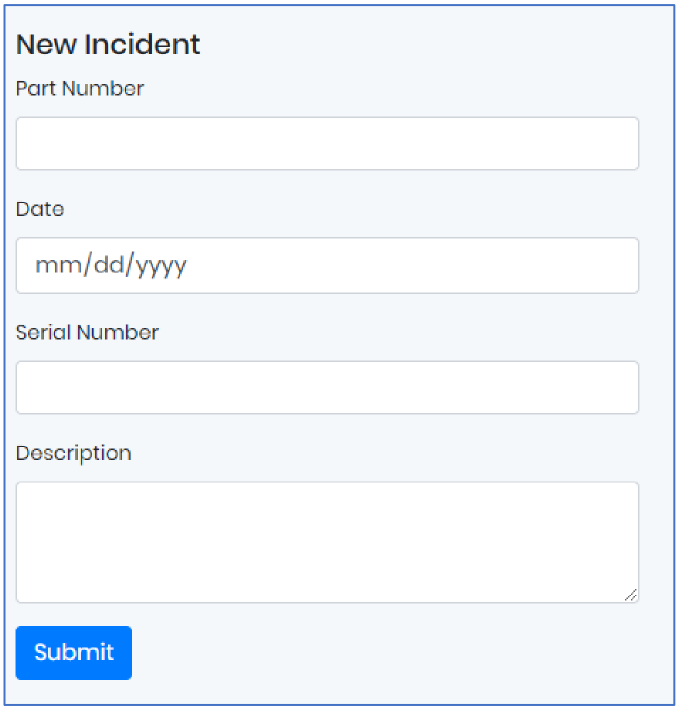
New Test





part #33567J

Failure Incidents	Failure Mode
incorrect material specification: shaft	widget sticks open
incorrect design: valve	widget sticks open
incorrect design: casting	widget sticks open
incorrect design: machining burr	widget sticks closed



New Incident

Part Number

Date

mm/dd/yyyy

Serial Number

Description

Appendix 3.0 Use Case Traceability Matrix

Appendix 4.0 Database UML and Schema