**PRI DATABASE SPECIFICATION**

**Product and Infrastructure Database**

****

**Document: DRAFT – REV. B**

**Date: 2019.06.07**

**CHANGE LOG**

|  |  |  |
| --- | --- | --- |
| **Revision** | **Date** | **Changes** |
| A | 2016.11.01 | Initial Release (Internal) |
| B | 2019.06.07 | Release Under Apache License v 2.0 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**TABLE OF** CONTENTS

[1.0 SCOPE 5](#_Toc465757350)

[2.0 DEFINITIONS 5](#_Toc465757351)

[3.0 NARRATIVE 6](#_Toc465757352)

[3.1 User Story - BASIC 6](#_Toc465757353)

[3.2 User Story – Design and As-Built Lists 9](#_Toc465757354)

[3.3 Vendor Designs / Products 10](#_Toc465757355)

[3.4 Specifications and ECOs 11](#_Toc465757356)

[3.5 Search 11](#_Toc465757357)

[3.6 Future Features 12](#_Toc465757358)

[4.0 WORKFLOWS 13](#_Toc465757359)

[5.0 TOOLS 14](#_Toc465757360)

[6.0 SCHEDULE 14](#_Toc465757361)

[7.0 MILESTONES / REVIEWS 14](#_Toc465757362)

[8.0 DELIVERABLES 14](#_Toc465757363)

[9.0 EXAMPLE USE CASES and SCENARIOS 15](#_Toc465757364)

[9.1 The Database Will Be Used To: 15](#_Toc465757365)

[9.2 A Database User Will Be Able To: 15](#_Toc465757366)

[10.0 EXAMPLE DESIGN and AS-BUILT LISTS 16](#_Toc465757367)

[10.1 Design List 16](#_Toc465757368)

[10.2 As-Built List 17](#_Toc465757369)

[11.0 WIREFRAMES 19](#_Toc465757370)

[11.1 Create Design Item 19](#_Toc465757371)

[11.2 DESIGN ITEM 20](#_Toc465757372)

[11.3 New Build 21](#_Toc465757373)

[11.4 PRODUCT 22](#_Toc465757374)

[11.5 Create Anomaly 23](#_Toc465757375)

[11.6 ANOMALY 24](#_Toc465757376)

[11.7 Revise 25](#_Toc465757377)

[11.8 PROCEDURE 26](#_Toc465757378)

[11.9 AS-RUN 27](#_Toc465757379)

[11.10 VENDOR DESIGN 28](#_Toc465757380)

[11.11 VENDOR PRODUCT 29](#_Toc465757381)

[11.12 ECO 30](#_Toc465757382)

[11.13 SEARCH TOOL (partial) 31](#_Toc465757383)

# SCOPE

This document is intended to provide an introduction to and overview of a database and front end web application that will archive, relate and present data products generated during design, fabrication and testing activities at Planetary Resources, Inc. (PRI).

PRI is a ~50 person “cradle-to-grave” spacecraft development company; i.e. activities range from product definition to design to fabrication/assembly to inspection to testing/validation to deployment to use.

Traditionally, product data software tools tend to focus either on the design process (PLMs) or the production process (ERPs). The problem for PRI is that existent software tools also tend to be far, far larger and more capable than it needs and there is no coherent system that spans between the design and production phases of a project.

PRI needs a stand-alone, communally accessible repository and user interface for data related to the development and delivery of its products.

# DEFINITIONS

Below are definitions for the main item types in the database. These are the items used throughout a project to answer questions like:

1. What do I need to make?
2. What do I plan to make?
3. What did I make?
4. Does what I actually made match what I planned to make?
5. Does what I made do what I need it to do?

**DESIGN (1234567)**≡ A description of one or more related items in response to a particular set of requirements. A Design might describe anything from a spacecraft to an electronics board to a bracket to a bolt to a tube of glue. The description might be accomplished through CAD models, data sheets, drawings, schematics, BOMs, post-it notes, etc.

**Revision (A, B, C….AA, AB)**≡ An update to a Design for the purpose of correction, improvement or to meet updated requirements. E.g. a Design, then, is referenced as 1234567 Rev A.

**Engineering Change Order (ECO)** ≡ Description of and justification for a change to a Design. An ECO is an incremental way to update a Design and get approval for that update without Revising the Design. One or more ECOs will always eventually be rolled up into in a Revision but a Revision does not require an ECO.

**PART (1234567-1, -2…)**≡ A component or assembly as described by a particular Design. A Design is a set of descriptions/instructions; a Part is what results from following those instructions. A single Design may describe more than one similar-but-distinct Parts. E.g.:

* -1 and -2 Brackets that are mirror-images of each other
* A -1 bracket made from aluminum and a -2 bracket with the same geometry made from steel.
* A -1 electronics board with a 100 ohm resistor and a -2 board with a 1 kohm resistor

**PRODUCT (1234567-1 S/N001)** ≡ A physical instance of a Part. If ten copies of Bracket P/N 1234567-1 are made, they are uniquely identified by a Serial Number that is used to track pedigree and use; inspection data, calibrations, use history, etc.

**DISCREPANCY** ≡ Any deviation of a Product from the intended Design (i.e. not built as designed)

**ANOMALY** ≡ Any deviation of a Product from expected behavior or performance (i.e. built as designed but not designed correctly)

**PROCEDURE (DOC-12345)** ≡ Instructions for doing work. E.g. a listing of the parts, tools and processes needed to build a spacecraft.

**AS-RUN PROCEDURE (DOC-12345.001)** ≡ A numbered instance of a Procedure that captures information specific to that work activity. E.g. five copies of a spacecraft are built to DOC-12345; an As-Run for each captures operator notes, hardware measurements, component serial numbers, test data, etc. for each specific spacecraft.

Additional data types:

**Specification** ≡ General PRI standards and/or process instructions, not specific to any one particular Design. E.g. how to apply epoxy paint, soldering standards, thermal chamber user’s guide.

**Build** ≡ A group of related Products, i.e. a batch of Products purchased or fabricated together - convenient for capturing notes on fabrication techniques/results, purchase orders, documents and other information applicable to all those Products

**Family** ≡ A group of related Designs. Two or more Designs, with no other similar or linked metadata, may need to be associated together because they share features such that an Anomaly on one should also point to the other.

# NARRATIVE

What follows is a narrative of one employee’s, let’s call him Sean, trip through the database and what he needs to be able to do and learn to accomplish his task.

## User Story - BASIC

Sean has been tasked with delivering a solution to a particular set of requirements. Sean brainstorms and uses existing in-house design tools (CAD, spreadsheets, napkin sketches, FEA, etc.) to generate documentation that fully describes a Widget that meets the set of requirements.

The full set of documentation, of information, that describes the Widget is a **DESIGN** and will have a unique seven-digit ID.

[See Wireframe 1.] Sean logs into the database and creates a single **DESIGN** record with the unique and clever ID of **1234567** and some basic metadata. Because PRI follows the standard ANSI revision scheme this record is **Revision A**.

[See Wireframe 2.] Because Sean created the **DESIGN** record he is the record’s default **Owner** and has write access to it; he’s able to edit metadata and attach documents to the record.

By default there is one **PART** associated with the **DESIGN** and is uniquely identified by appending a -# to the **Design Number**, i.e. **1234567-1**. The **Design Number** uniquely identifies the set of information that describes a thing; the **Part Number** uniquely identifies the thing itself that is produced according to the design information. A **DESIGN** may describe multiple distinct **PARTS** that differ slightly from each other and are uniquely identified as **1234567-1** through **1234567-N**.

Sean continues completing the **DESIGN** record; he enters more metadata, some of which is selected from pre-populated lists, e.g. **Material** for each **PART**. Sean enters a “current best estimate” (**CBE**) mass for each **PART** and a **%Uncertainty** on that mass; the database calculates and displays “predicted best estimate” (**PBE**) mass; PBE = CBE\*(1+%Unc)

Sean doesn’t add any **Components** to the **PART** at this point – that’ll be a different user story.

[See Workflows] Once the information in the DESIGN record is complete and correct to the best of Sean’s knowledge and he’s ready to create one or more Widgets he moves the record along its workflow from the **In Work** state to the **Approval** state. While in **Approval**, the record is read-only. One or more pre-specified users with **Approver** permissions can either move the record to **Released** or send it back to In **Work if** edits are needed.

[See Wireframe 3.] Once the **DESIGN** record is **Released**, Sean decides to order ten (10) Widgets, **P/N 1234567-1**. These ten Widgets are associated with each other as a **BUILD** and each Widget is a **PRODUCT**, uniquely identified by appending their **Part Number** with a three-digit **Serial Number, i.e. 001 – 010**. When creating the BUILD, default metadata for the group of PRODUCTs is specified (these may diverge later) and other metadata and file attachments that will apply to the **BUILD** may also be specified.

[See Wireframe 4.] A **PRODUCT** record is created for each item in the **BUILD**. Each **PRODUCT** record is the collection of all information specific to that particular S/N, that particular physical instantiation of a Widget. Each **PRODUCT** record is associated with **Revision A** of the Widget **DESIGN**. Sean is the **Owner** of the **PRODUCT** record and so has write-access to it while its workflow state is **In Work**. Sean can attach documents and images to the record. While inspecting the Widgets Sean finds a fabrication error with one – something about it doesn’t match the **DESIGN** description and must be noted on the record as a **DISCREPANCY**. Sean adds a **DISCREPANCY** (uniquely ID’d) to the record and is able to add a text **Description** and **Justification** and select a **Disposition** from a pull-down list.

When Sean thinks that the **PRODUCT** record is complete and accurately reflects the condition of the Widget he moves the record from **In Work** to **Approval** and the record becomes read-only. One or more **Approvers** can either move the record to **Released** or send it back to In **Work if** edits are needed.

Sean goes to install the Widgets into their **Next-Level-Assembly**, e.g. they, along with bolts and such, are **Components** (children) of another **PART**…and they don’t fit. Crap. Something might be wrong with the design of the Widgets.

[See Wireframe 5.] Sean creates an **ANOMALY** record with some basic metadata; the **ANOMALY** is associated with the particular **DESIGN** and Revision in question. The **ANOMALY** record has a unique five-digit ID, e.g. **A-12345**

[See Wireframe 6.] As Owner of the **ANOMALY** record, Sean has write-access to edit text fields, attach files and images and URL links, and make selections in pull-down lists. When Sean is satisfied that the **ANOMALY** record fully documents the reason and corrective action for the Widget not fitting he transitions the record through its workflow states to **Approval** and the record becomes read-only. One or more **Approver(s)** can then transition it to **Closed** or back to **In-Work**.

[See Wireframe 7.] The corrective action for the **ANOMALY** is to fix/update the design of the Widget. Sean (or any user) uses the **Revise** button on the **DESIGN** record to create a new record, **Revision B** of **1234567**. The new design record starts as a copy of **Rev A** (same metadata, attachments, etc.) but is at the **In Work** state and thus its **Owner** has write-access. Sean edits the metadata, attachments, etc. for **DESIGN** record **1234567 Rev B** until he’s ready to submit it for **Approval**.

However, all the Widgets are still compliant to and associated with **Rev A**; additional work needs to be done to each one to bring it up to compliance with **Rev B**.

[See Wireframe 8.] Sean creates a **PROCEDURE** record in the database with a unique five-digit number, i.e. **DOC-12345**. This record is associated with the Widget **DESIGN**, **1234567**. (A **PROCEDURE** could potentially be associated with multiple **DESIGNS**…) As the **Owner** of this record, Sean is able to edit metadata, attach files and images as needed to fully describe the necessary work instructions to bring the Widgets into compliance with **Rev B**. There is no workflow for a **PROCEDURE**.

[See Wireframe 9.] Now Sean needs to perform the work ten different times, on each of the ten Widget **PRODUCTS**, and capture information for each. Each time he runs the **PROCEDURE** he creates in the database an **AS-RUN** record uniquely identified by appending three digits to the end of the **PROCEDURE** number, e.g**. DOC-12345.001** through **DOC-12345.010**.

Alternatively, Sean could have written the instructions such that all ten Widget **PRODUCTS** could be updated in one **AS-RUN** instance. An **AS-RUN** is associated with one **PROCEDURE** and is also associated with one or more **PRODUCTS** of the **DESIGN(S)** associated with that **PROCEDURE**. Each **PRODUCT** record will include a link to any and all **AS-RUNs** associated with it.

As Sean performs work on each Widget he updates metadata and attaches files/images to each **AS-RUN** record. When Sean completes the work and each record is up to date he submits each record for **Approval**, after which they will be **Released** and read-only.

Sean then goes to each Widget **PRODUCT** record [Wireframe 4.] and **Updates Rev** – this action moves the record back to In Work and associates it with the **Revision B** record, **1234567\_B**.

## User Story – Design and As-Built Lists

Suppose that Sean’s next solution to a set of requirements isn’t one single thing, but a joining of multiple things – two Widgets and a Bracket need to get bolted together to make an Assembly. Just like the Widget, Sean creates a **DESIGN** record for the Assembly, **1000789**, with **PART** **1000789-1**.

Via the **DESIGN** record [see Wireframe 2.] Sean is able to add children **Components** to the Assembly **PART**. Sean is able to choose as **Components** other existing **PARTS** within the database and specify a **Quantity** for each **Component**. The Widget, the Bracket and the Bolts are all **Components** of the Assembly (and already have **DESIGN**/**PART** records in the database). For this narrative he Widget quantity is two (2), the Bracket quantity is one (1) and the Bolt quantity is twelve (12)

If the Assembly is itself a **Component** of another **PART**, or if any of the Assembly’s **Components** themselves have **Components**, then there is a larger implied, multi-tier hierarchical **DESIGN LIST** of **Part Number**s that the database can generate and display.

* Component = child
* Next Level Assembly (NLA) = parent
* A **PART** may have zero or many **Components** and may have zero or many **NLAs**

If a **PART** has children **Components**, then its **Mass** is no longer a value specified by Sean – it is the product sum of the masses and quantities of its **Components**. In this case, **CBE** and **PBE** are rolled up from the Components and %Uncertainty is calculated.

A larger, multi-tier **DESIGN LIST** for, say, an entire spacecraft will include a comprehensive **CBE**, **PBE** and **%Uncertainty** mass roll-up for the top level assembly. This is a critical capability for estimating spacecraft launch mass during the design phase of a project.

Sean completes the **DESIGN** record (**Rev A**) for the Assembly just like he did for the Widget and submits it for approval to become **Released**.

Now Sean is ready make five (5) Assemblies. Sean creates a **Build** [see Wireframe 3.] from the Assembly **DESIGN** record and specifies S/Ns 001 – 005 for the five Assembly **PRODUCTS**. The five physical instances of the Assembly, the **PRODUCTS**, are uniquely identified as **1000789-1.001** through **1000789-1.005**, commonly displayed as **1000789-1** **S/N001** through **S/N005**.

While making the Assemblies, Sean needs the ability to track which specific Widgets get installed in which specific Assemblies; e.g. Assembly S/N001 contains Widget S/Ns 002 & 008.

When Sean opens the **PRODUCT** record for one of the Assemblies, he sees that a **Component** list is automatically generated from the **DESIGN** record. This is the beginning of an **AS-BUILT LIST**. Because the Widget is a quantity two (2) in the Assembly **DESIGN** **LIST**, there are two line entries for it in the **AS-BUILT LIST**.

A **DESIGN LIST** is a hierarchical list of **PARTS** and their quantities.

An **AS-BUILT LIST** is a hierarchical list of **PRODUCTS**.

Sean is able to specify a Serial Number for each **Component** in the **AS-BUILT LIST** (only) from existing **PRODUCTS** in the database. A given Serial Number can only be specified as a **Component** once – Sean can’t install one Widget in two different Assemblies.

The Bolts are different; they are a different type of **PRODUCT**. Sean doesn’t want to have to make separate records for each of the hundreds of those Bolts he has available and doesn’t want to have to serial number them individually.

When Sean made a **Build** from the Bolt **DESIGN** record he specified that the **PRODUCT** would be a **LOT**. The **LOT** has a unique ID just like **Serial Number**, i.e. L001. The only differences between a **LOT** and a **S/N** are:

* A **LOT** may have many **NLA** parents
* A **LOT** is only one line entry in an **AS-BUILT LIST**, regardless of quantity in the **DESIGN LIST**. Sean doesn’t want “Bolt” showing up twelve times on the Assembly as-built list because they’re all interchangeable with the same pedigree and documentation.

**DESIGN LISTS** do not perform mass roll-up calculation. If Sean wants to know the actual mass of an assembled spacecraft he’s going to weigh the spacecraft…and then out of curiosity compare it to the rolled-up PBE on the spacecraft’s **DESIGN** record.

Sean continues building Assemblies and updating their **PRODUCT** records accordingly. During this time he may find reason to **Revise** the Assembly **DESIGN** record – this **Revision** might include changing the quantity of a **Component** in the **DESIGN LIST**. If this happens, and if Sean updates the associated **Revision** of one or more of the Assembly **PRODUCT** records, the **AS-BUILT LIST** for that/those record(s) would update accordingly.

Once Sean is satisfied that an Assembly **PRODUCT** record accurately reflects the state of that physical assembly, he submits the record for **Approval** after which its state is **Released** and it is read-only.

## Vendor Designs / Products

Sean needs to include an off-the-shelf purchased component in one of his **DESIGNS**. This might be, for example, a camera that he’s going to purchase five (5) of and install in the Assemblies (after rev’ing their design, of course). Just like before, Sean needs to collect design information / description of the camera in general as well as information that is specific to the particular cameras he gets (characterizations, calibrations, warranties, etc.)

[See Wireframe 10.] Sean creates a **VENDOR DESIGN** record for the camera. This is identical to a **DESIGN** record except:

* There is no distinction between a design number and part number, no -#. There is only whatever ID number the vendor sells the item by.
* The **Part Number / ID** can be any length alphanumeric string, not just seven digits
* **VENDOR DESIGNS** do not have **Components**.

As the owner of the record, Sean has write-access to edit metadata, attach files and images, etc. and then submit the **VENDOR DESIGN** record for **Approval**.

[See Wireframe 11.] Once the **VENDOR DESIGN** record is Released, Sean is able to create a Build of five (5) **VENDOR PRODUCTS**. A **VENDOR PRODUCT** is identical to a **PRODUCT** except:

* A **VENDOR PRODUCT** serial number may be any length alphanumeric
* **VENDOR PRODUCT** do not have **Components**.

## Specifications and ECOs

Any user is able to create a **SPECIFICATION** record. A spec is just a generic set of instructions or information or standards, etc. that apply to the company as a whole; e.g. how to torque a bolt or solder a wire or operate the coffee maker. A **SPECIFICATION** is uniquely identified by a five-digit number, e.g. **SPEC-12345**. A **SPECIFICATION** record has some basic metadata (title, owner, etc.) and file/image attachments. A **SPECIFICATION** does not apply to any one particular **DESIGN** and does not have a workflow.

In the course of testing a **PRODUCT**, Sean discovers a design flaw and creates an **Anomaly** to document it. The resolution of this **Anomaly** is to fix the design but he wants to document and get approval on the design fix without updating all the **DESIGN** documentation yet because he’s pretty sure there are going to be additional fixes needed and he wants to roll them all up together in one **Revision** effort. But, he wants to fix the existing **PRODUCTS** now.

[See Wireframe 12.] Sean creates an **ECO** record associated with the **DESIGN** record. The **ECO** is uniquely identified with a five-digit number, e.g. **ECO-12345**. Because Sean is the Owner of the **ECO**, he is able to edit metadata for it and attach files/images. When ready, Sean submits the **ECO** for **Approval**, after which it will be **Released** and read-only. (an ECO is very similar in structure and behavior to an Anomaly)

## Search

Half the reason PRI needs this database is so that Sean can find information on someone else’s work and vice versa. A simple Search Tool allows Sean to search by unique key for any of the major database items; Design, Product, Anomaly, ECO, Procedure, Specification.

A more Advanced Search Tool [see Wireframe 13.] allows Sean to Boolean search with multiple metadata fields – which fields are displayed/searchable depends on what type of item is being searched for.

With either Search Tool, results are displayed in table form with TBD columns of relevant/interesting metadata displayed.

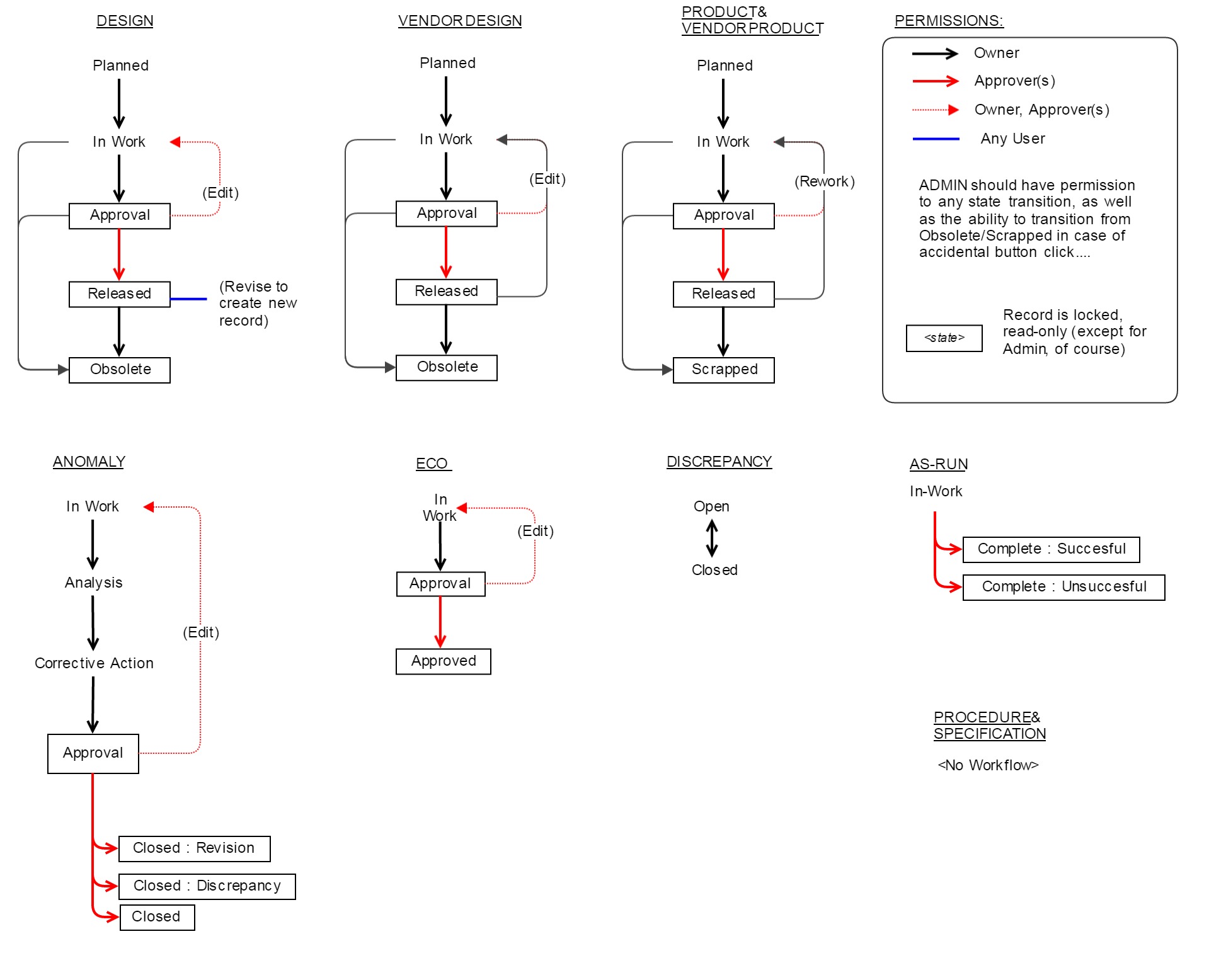
## Future Features

Sean probably has a lot of desires for database functionality and features that he just hasn’t thought of yet; but some he has.

1. Design Lists and As-Built Lists should be exportable to Matlab, Excel
2. Email notifications: when a record is submitted for Approval, the relevant Approvers should get an email notification that they have a task waiting
3. Dashboards: when a user logs in to the database they should have a (customizable?) dashboard that displays their pending tasks – maybe the refreshed results of a particular search

# WORKFLOWS

Most of the database items will have a Workflow – a sequential series of States that are used to denote status/condition of the item and in some cases limit write-access and/or permissions to transition from one State to another.



# TOOLS

Due to in-house expertise, we want the web application to be written in the latest version of Python (currently 3.5.2), preferably using the Flask microframework with extensions.

We wish to avoid the use of Javascript frameworks such AngularJS and ReactJS, as we feel these are unnecessary for this application, but expect jQuery to be part of the implementation, as well as Bootstrap.

The application should be able to run on for instance Gunicorn with an NGINX proxy. The database we wish to use in the background is the latest release of PostgreSQL (9.6 as of writing).

The application must be able to authenticate users via Active Directory.

# SCHEDULE

TBD

# MILESTONES / REVIEWS

TBD

# DELIVERABLES

TBD

# EXAMPLE USE CASES and SCENARIOS

Below are some examples of necessary capabilities; these lists are by no means comprehensive.

## The Database Will Be Used To:

* Create uniquely ID’d Design and Product records
  + collect and display associated data for each record (e.g. part name, project, owner, etc.)
  + capture revision-specific design data
* Organize Part Numbers into hierarchical Design Trees
  + display Design Trees with additional data columns (e.g. mass, part name, etc.)
* Organize Serial Numbers into hierarchical As-Built lists
  + display as-built lists with additional data columns (e.g. mass, part name, etc.)
* Collect component CBE mass and mass uncertainty, calculate component PBE mass and assembly mass roll-ups.
* Archive, or link to, all design documents
  + Released drawings, schematics, layouts, STEP, PDF, etc.
* Archive, or link to, all as-built documents
  + As-run procedures, data sheets, calibration/characterization data, pictures, etc.
* Create and track Discrepancies, Anomalies and Engineering Change Orders
* Create Procedure numbers and records for archiving Procedure and As-Run documents
* Create Specification numbers and create records for archiving Specifications
* Create Vendor Part records for archiving, or linking to, vendor part documentation and data

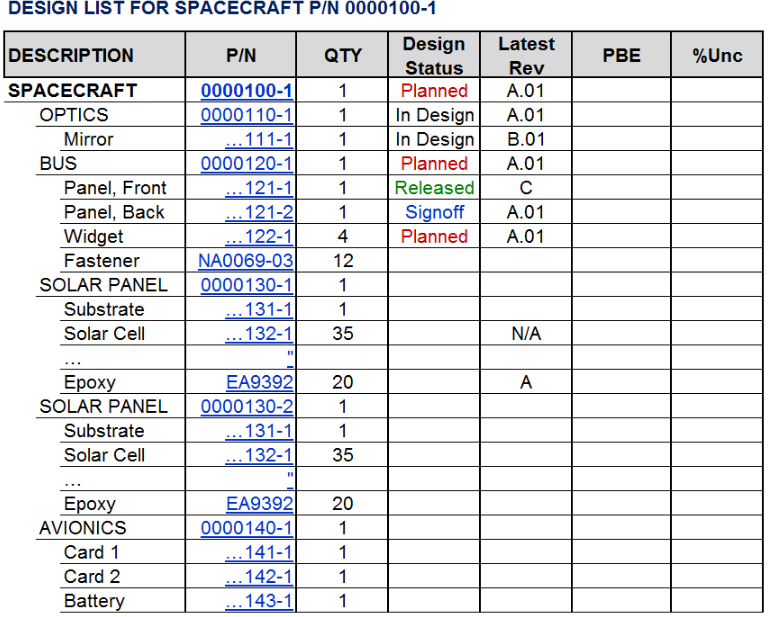
## A Database User Will Be Able To:

* Pull up the *hierarchical*Design List for A6 (A6 is a spacecraft)
  + or a Design List that includes estimated mass, latest rev, design status, etc.
* Pull up the *hierarchical*As-Built List for A6 S/N002
* Start w/ the drawing number of the top-level spacecraft assembly and be able to find out which S/N avionics boards went into spacecraft S/N 003 (and vice versa)
* Get the interface drawing / STEP file for the A100 Power Board
* Get the as-built documentation / test data for a particular Reaction Wheel
* Find out what S/Ns exist of 1001230-2
* Find out what Aanomalies exist against the Star Tracker design
* Find out what Discrepancies exist against the Star Trackers in either A6 spacecraft
* Find out who is responsible for any given component
* Find out the design status of a thing
* Find out the build status of a thing
* Any one person should be able to pull up all information to at least fabricate / purchase everything that went into, for example, A3

# EXAMPLE DESIGN and AS-BUILT LISTS

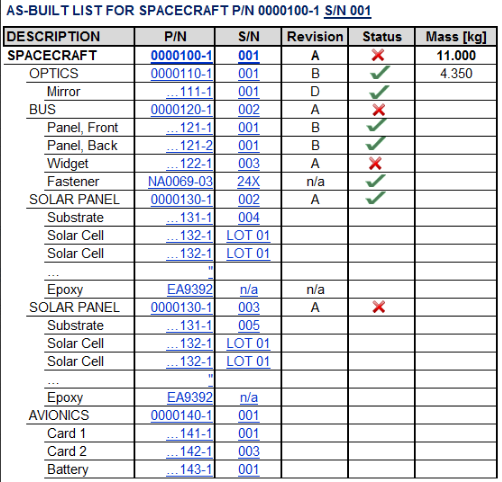
## Design List

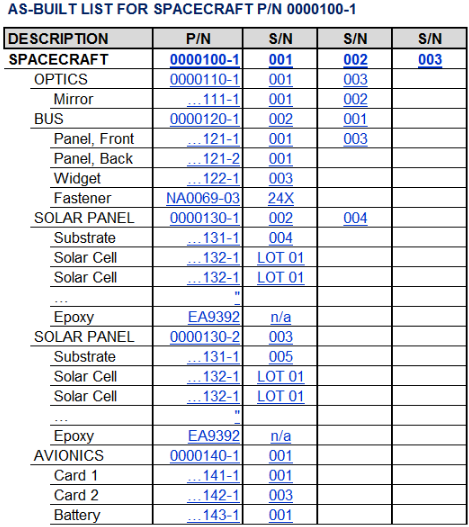
A Design List of a particular Design must include Part Numbers and Quantities of subcomponents. It may also include additional information such as design status or estimated mass roll-up, etc. as deemed useful.



## As-Built List

An As-Built List of a particular Product must include the Serial / Lot number(s) of installed subcomponents. It may also include design revision or as-measured mass, etc. as deemed useful.



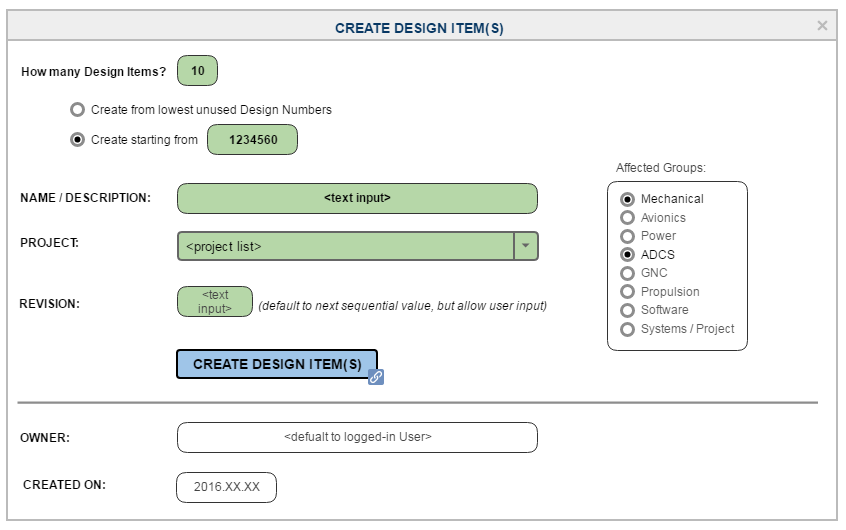


# WIREFRAMES

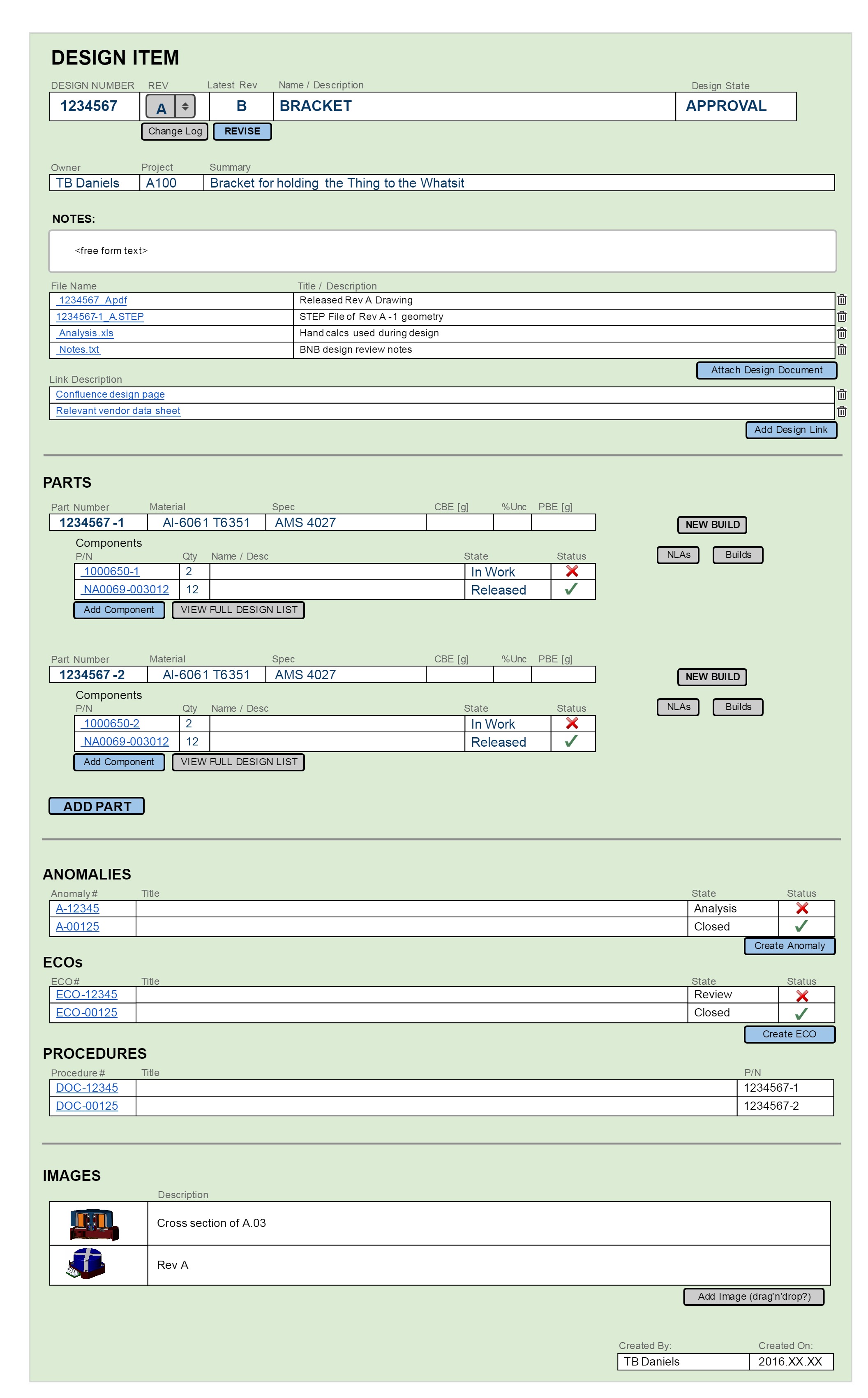
**NOTE**: these wireframes are extremely notional – they are not intended to convey anything other than the types of data to display and the types of interaction a user would have with the database.

Taking away any style or design intent from these is very strongly discouraged.

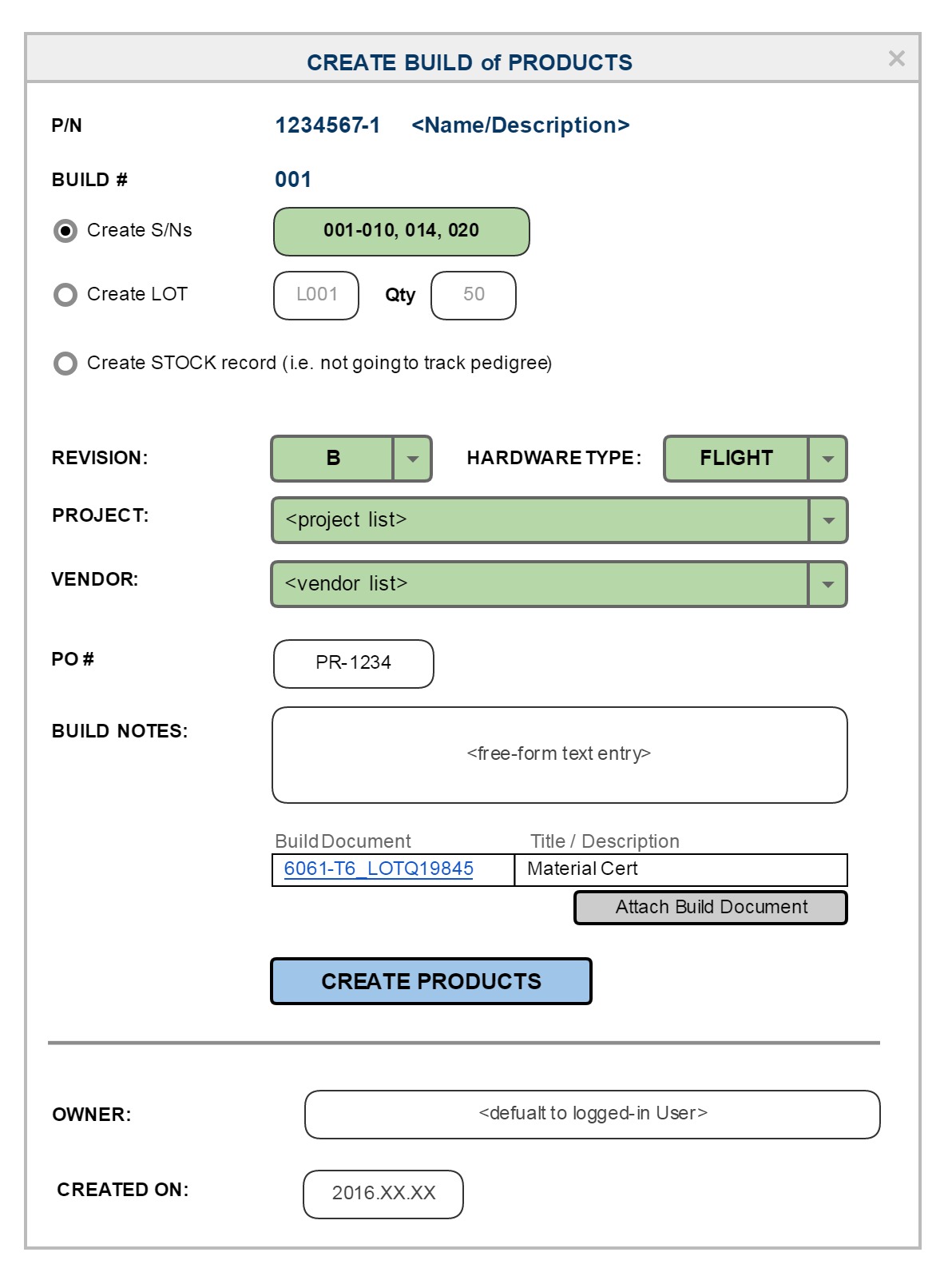
## Create Design Item



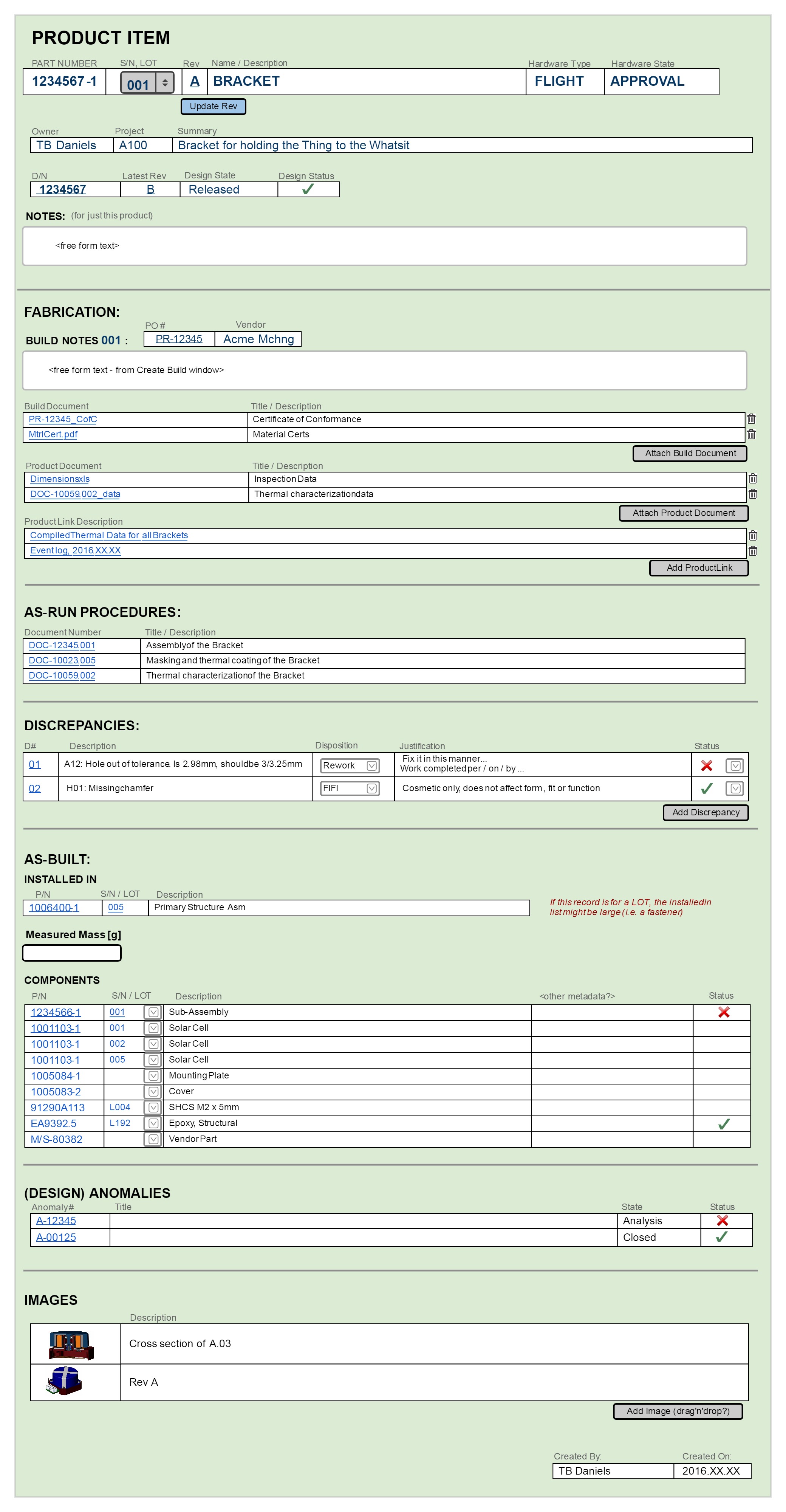
## DESIGN ITEM



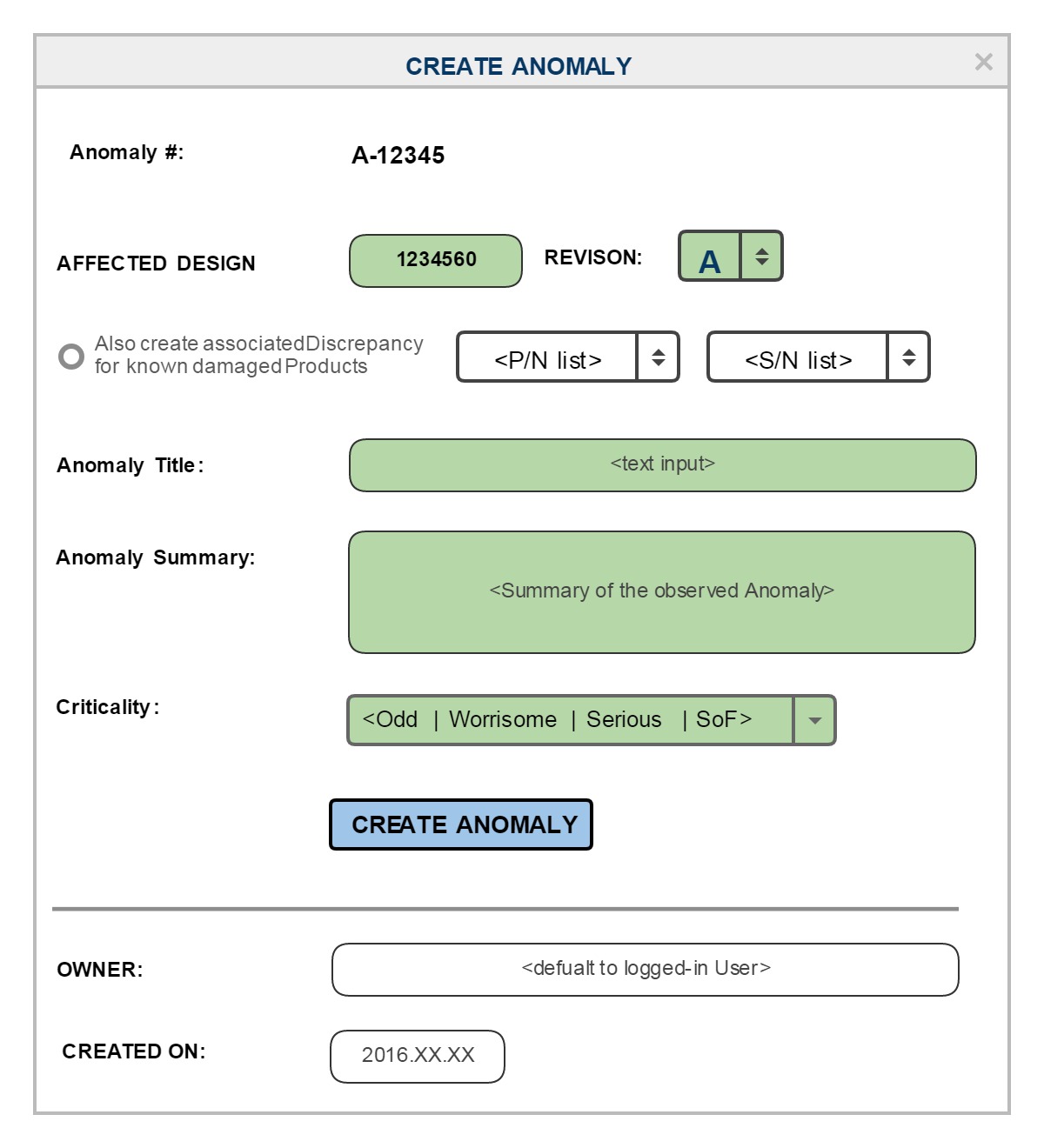
## New Build



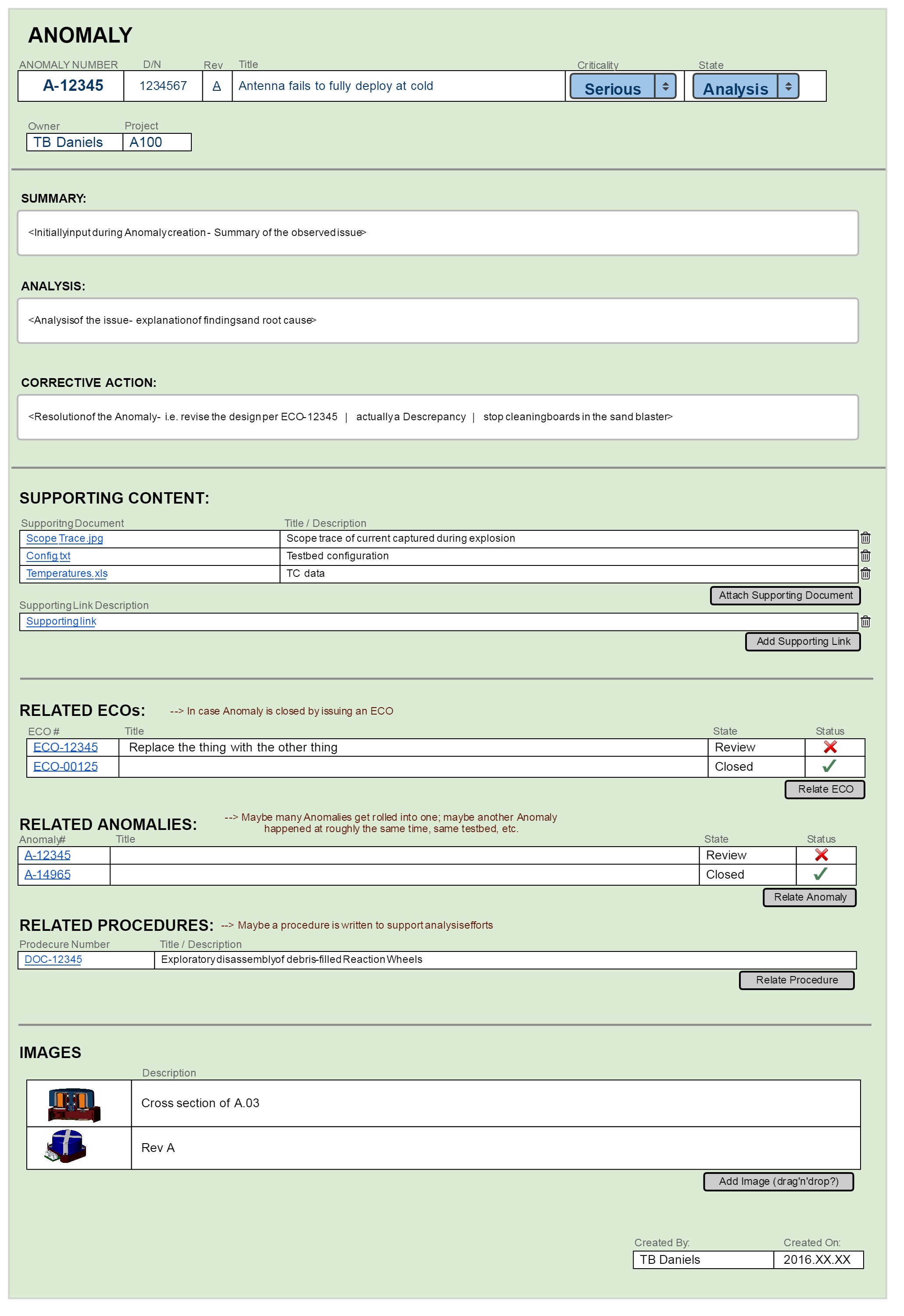
## PRODUCT



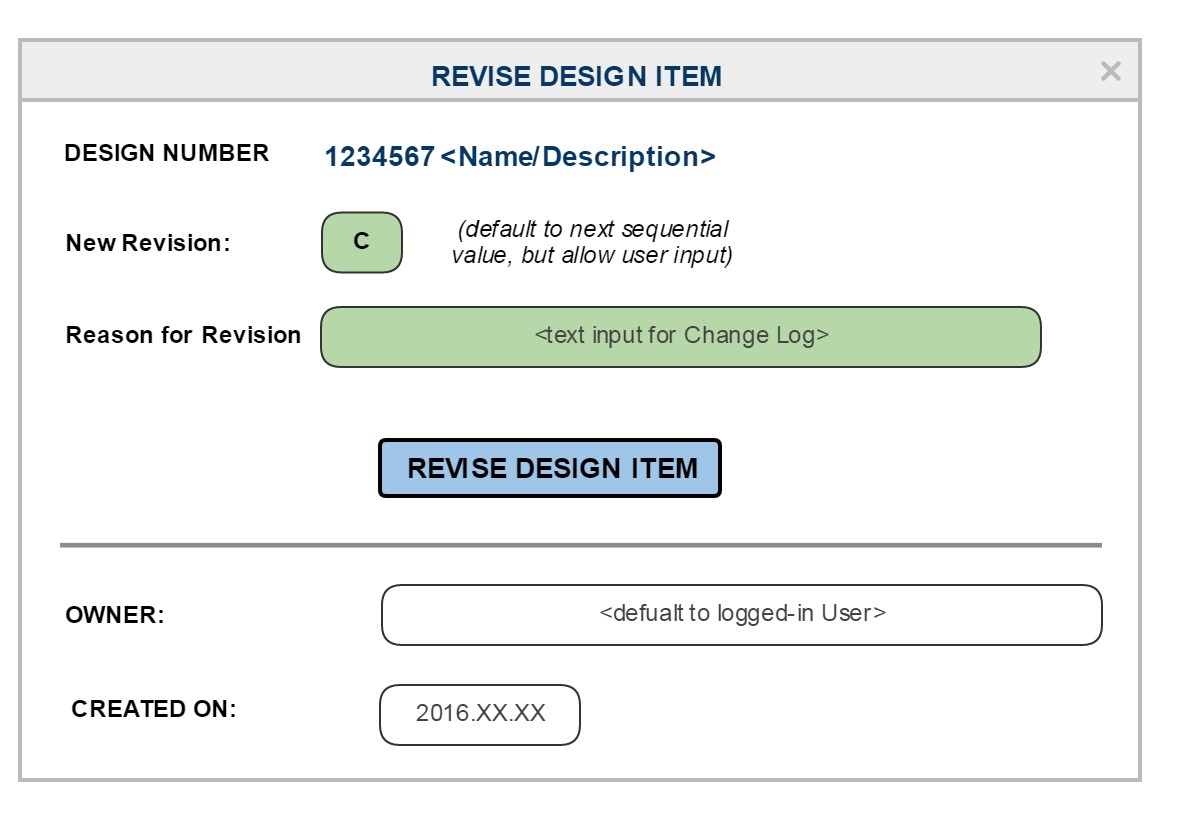
## Create Anomaly



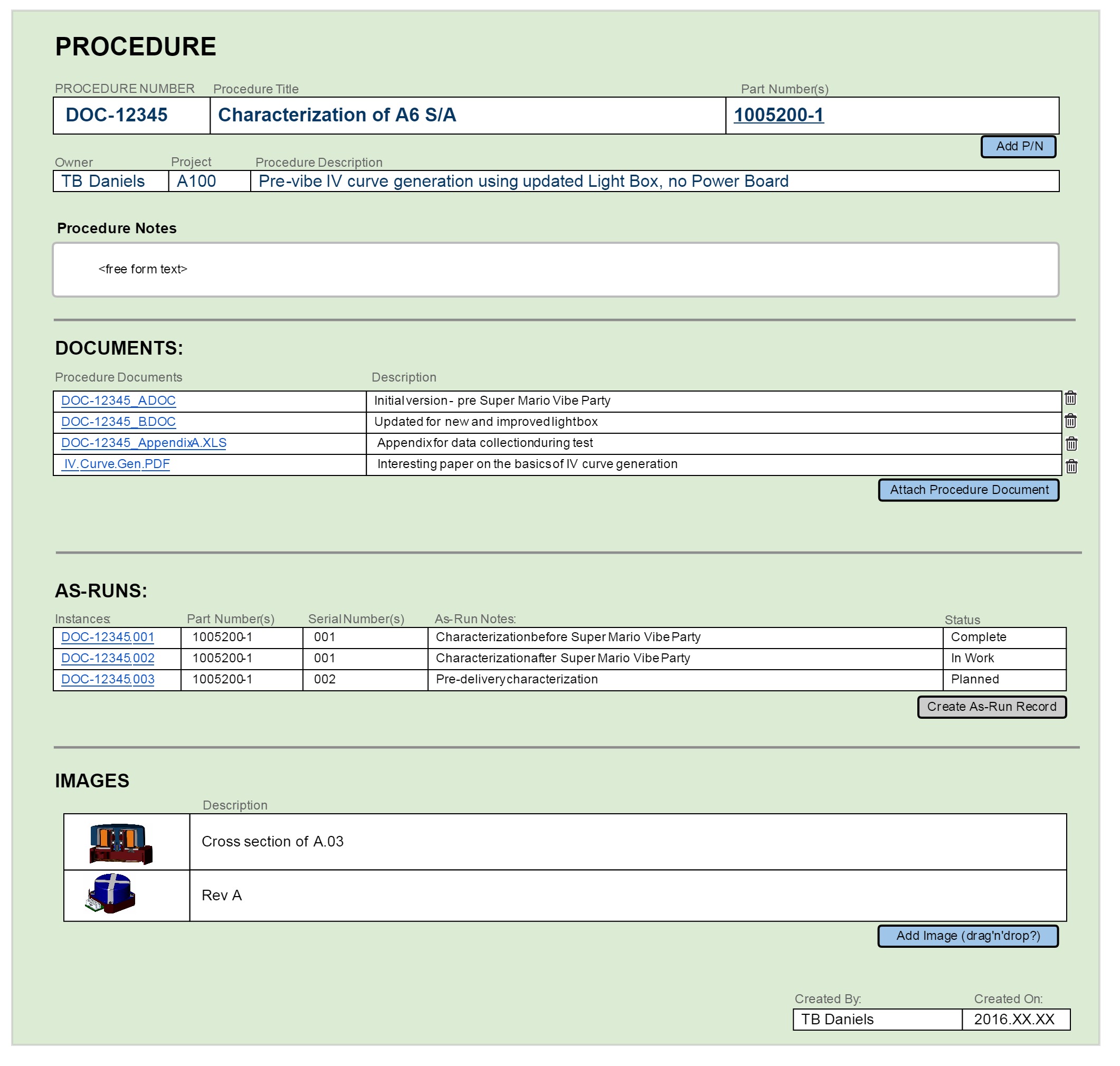
## ANOMALY



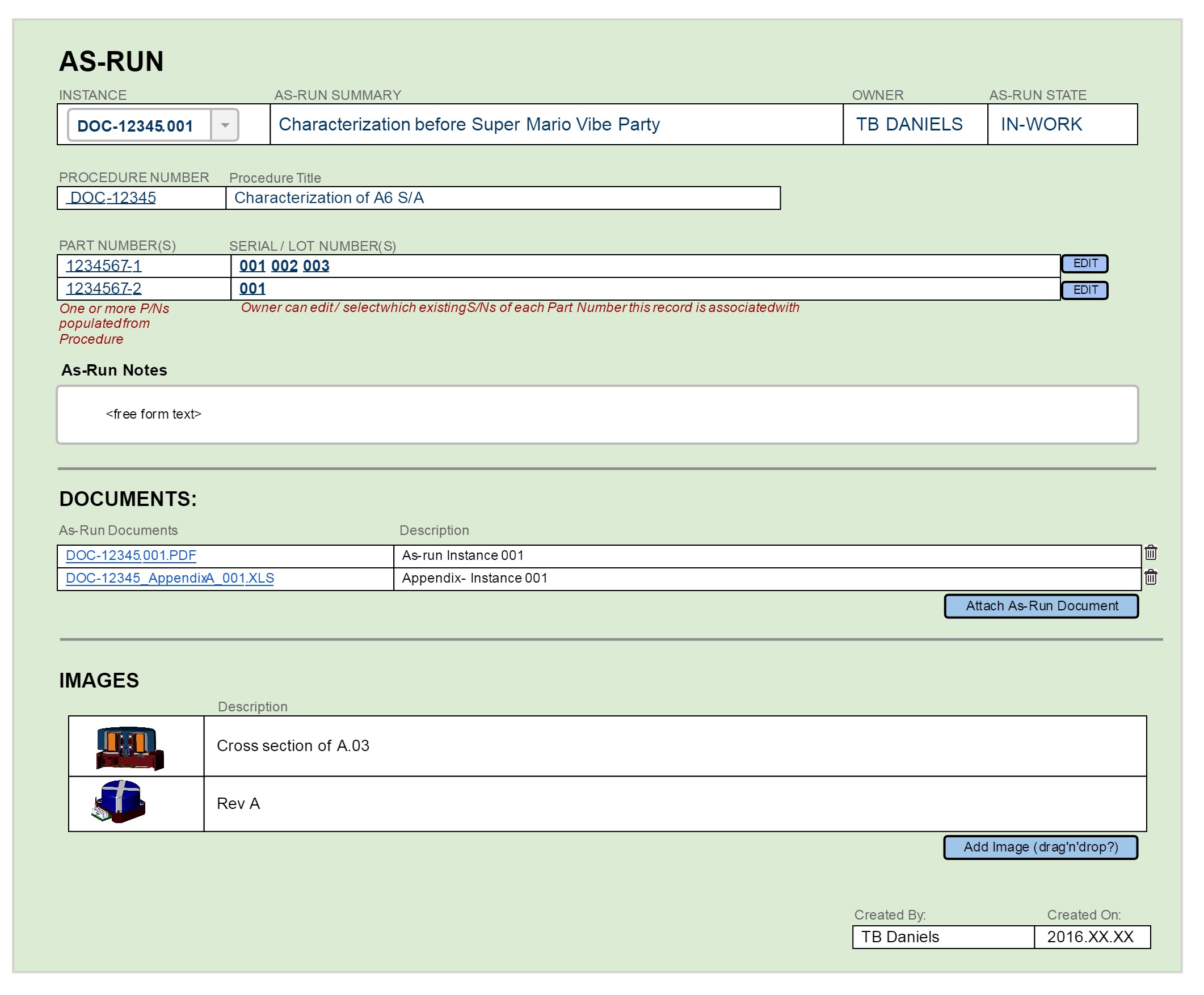
## Revise



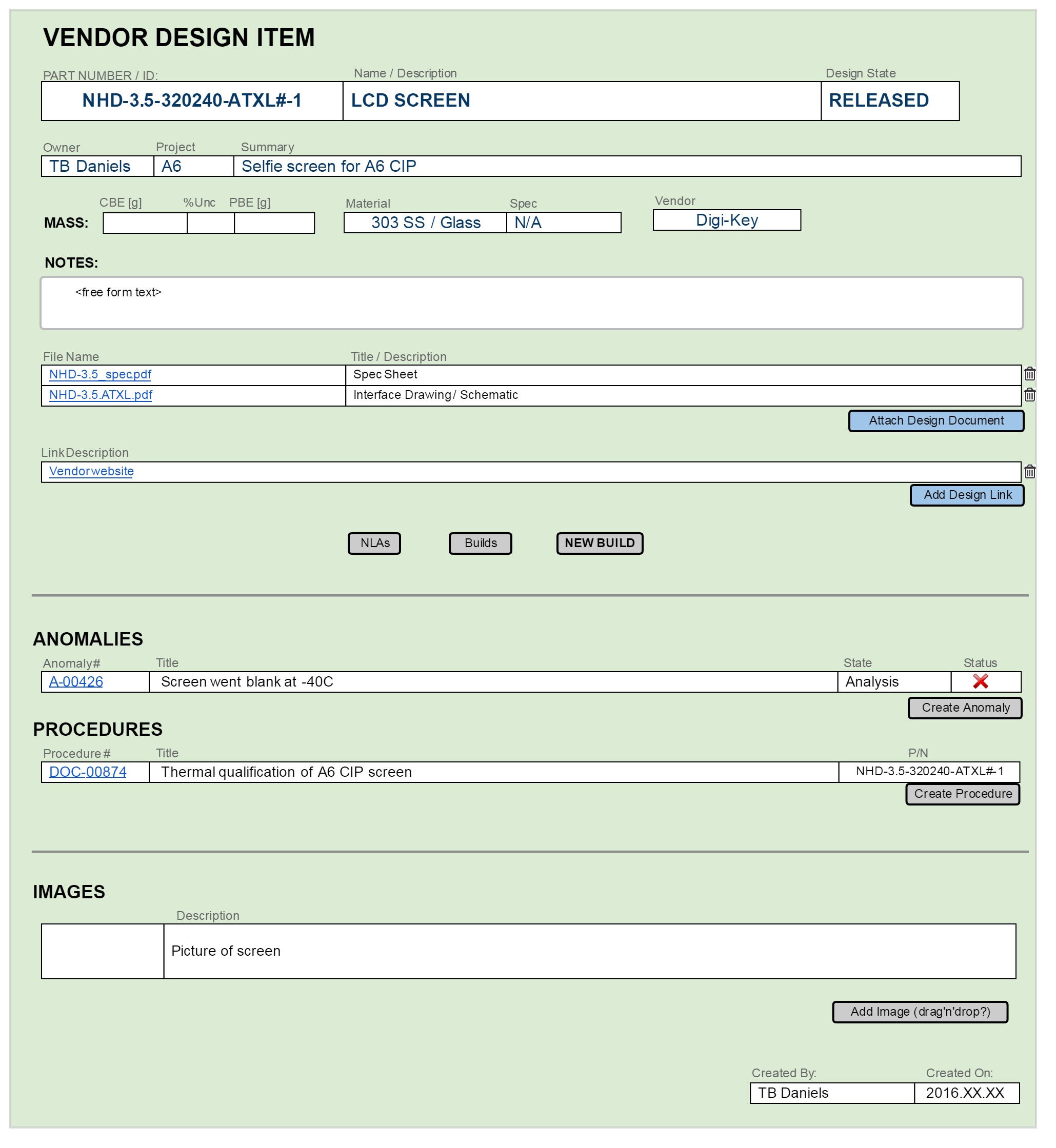
## PROCEDURE



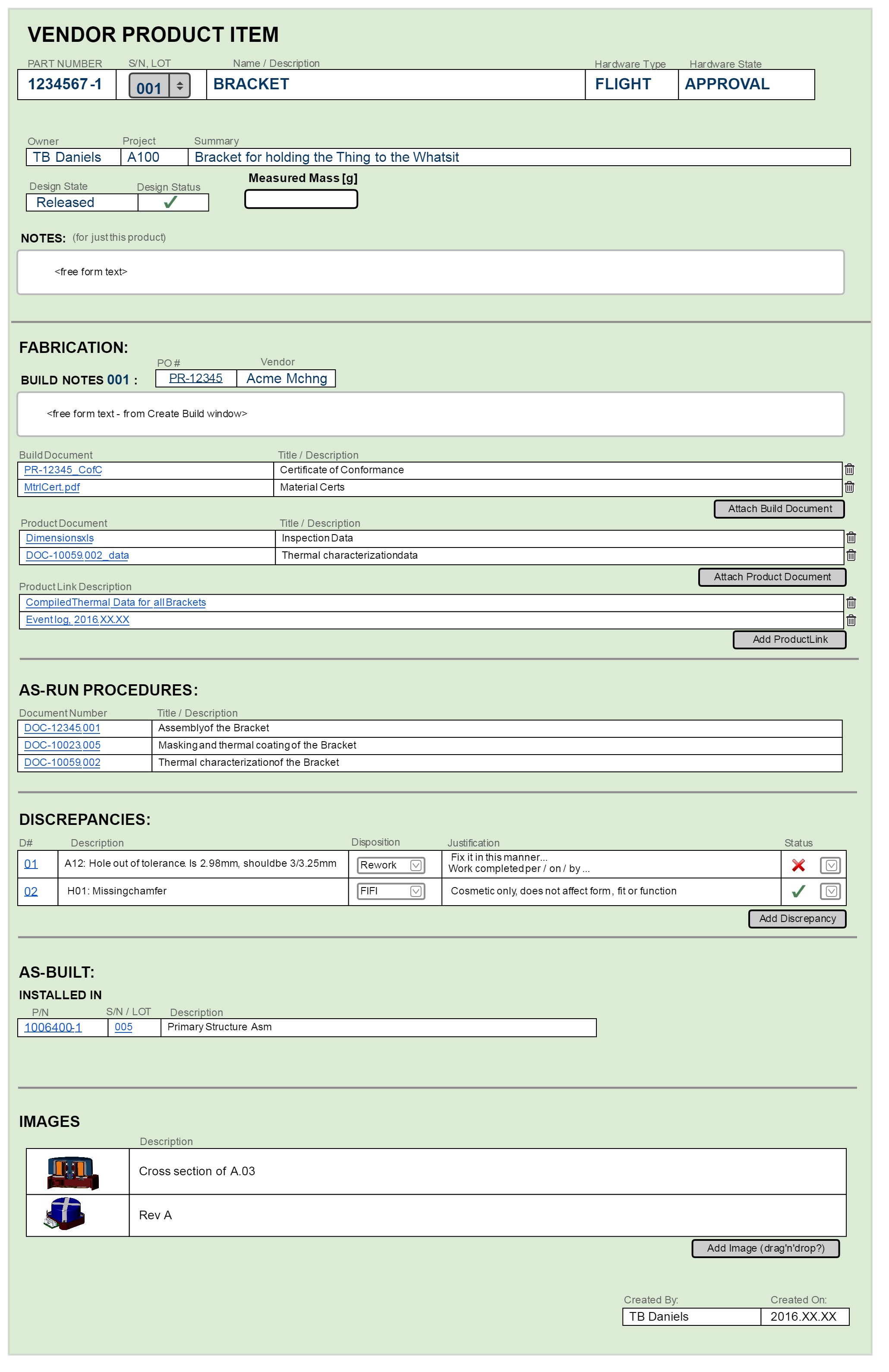
## AS-RUN



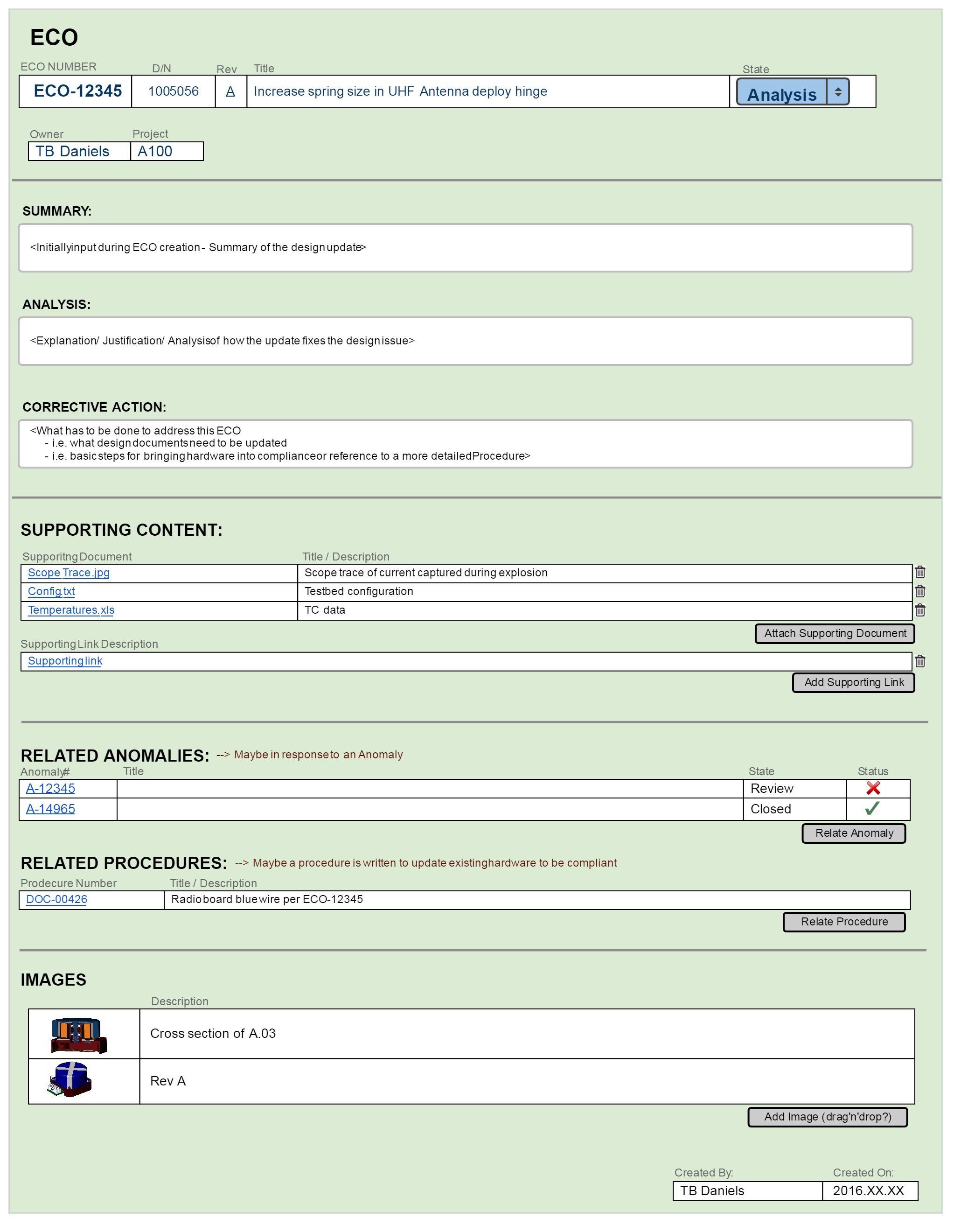
## VENDOR DESIGN



## VENDOR PRODUCT



## ECO



## SEARCH TOOL (partial)

