Last updated : October 12, 2024

Detector Construction Hardware Database Requirements

Introduction:

This document is *not meant for* any technical suggestion like database software, HTML, php, host server, mail forwarding, firewall or plotting engine etc. The overall idea is to facilitate the management and tracking of various hardware components for various sub-systems effectively while at the same time ***it shouldn't be a tremendous burden to the users.*** This document presents the necessary features which are felt useful by the ePIC collaborators but it doesn't mean that any other feature can't be added or modified afterwards. Also, it is recognized that maintaining the DB regularly as a "Global Administrator" will require a dedicated personal, and sub-system DSL's will have the responsibility to ensure that the information is up-to-date for their respective DB. This document will serve the purpose of overall expectations and structure of the DB for the software team to get started.

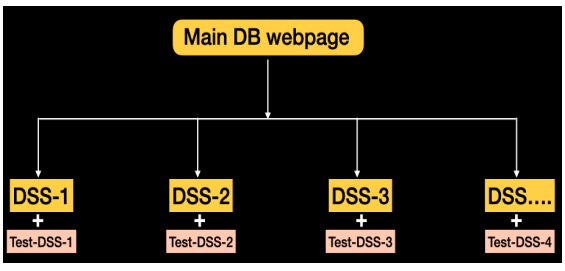
The structure of the database and some of the most salient features are listed below:

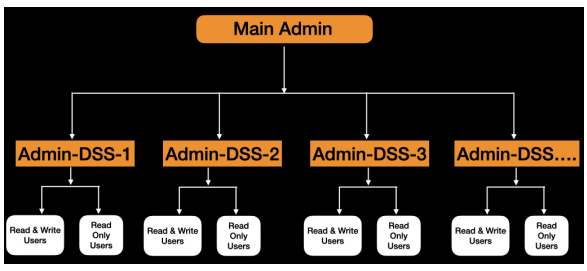
High level structure of web pages:

1) Landing page should have a list of all DB's with links.
2) There should be a "Real database page" and a "Test database page" for each Detector subsystems (DSS's). It's very useful particularly during the early days for new users to practice and get acquainted. Test database would be a clone copy of the real DB at a fixed period of time intervals.
3) Test and real DB pages should be visibly different (e.g color scheme) to avoid accident/confusion.

Administration hierarchy:

1) There should be a Global level admin with access to the main Database, sub DB's and webpages.





2) The detector subsystem leaders (DSL's) should have access to their own DB with privileges to add personals and their rights, e.g. person "A" has only view access but "B" has read and write/submit access.
3) The privilege to change the language of a criteria, to add or subtract some requirement criteria should be only reserved with the DSL not anyone else.
4) If DB is hosted at national labs; we must keep in mind the diverse user base who might not have access behind the lab's firewall.

DB Backup:

All the DB tables and user's data (figures, txt, etc.) should be backed up to a separate storage.

Contact page Info and Email option:

1) DSL's can put in the details of personnels from different institutions who are participating in the construction of their sub-system. It can include full name, email address and the full shipping address. Cell phone and Office phone numbers are optional.
2) It can also have a feature to select personals from the list and send them emails.
3) The content of the email can also be listed on this page very similar to the mailing list archive.

Unique Identification for numbered Items and non-numbered items:

1) The components which are uniquely characterized (e.g HRPPD tiles for pfRICH), should have a unique ID i.e barcode/QRcode or any other permanent unique Number. This ID should be unique, searchable in DB and identifiable on hardware components for one-to-one relationship and history.
2) Items which are not characterized individually e.g. ESR foils for LFHCal, should be only stocked unlabeled to access the need and usage to avoid delays due to shipping etc.

Data upload and Plotting:

1) Depending on the subsystem needs, each item page should have a feature of data file upload (with their log history if it's replaced with another one).
2) DB should also automatically append a timestamp to the file name to avoid mishandling.
3) Item pages should also be compatible with some plotting engine where it can read a data file and display the plot when asked. The file format and display style should be coordinated with the main DB Admin and DSL's.
4) To make a red flag for the users, each entry could have a boolean logic associated with it e.g Humidity > 50% could be flagged with red color.

5) One could think of color coding to sort the finished and unfinished tasks, as an example, if Something is tested good: Green, yet to be tested: Yellow, if it's tested bad: Red

Query and inventory:

1) There should be an inventory of items page which clearly reflects the parts of a particular category which have been used/tested fully and remaining items in that category.
2) This page should also have OR & AND logic for different criteria combinations to sort the query.

Shipping and receiving:

1) It should be treated as a feature not an imposed necessity for each sub-system.
   It definitely has its advantages in terms of log for tracking number, who shipped something and who received it, is your package lost in the busy traffic of items you are sending and receiving? Or you packed something but forgot to ship.

   But, it should be left up to the subsystems how they want to manage it.

Progress plots: stage of preparation with time

1) Every category will have a list of numbered items and steps to be followed with them.

2) Based on when a step is completed on how many items we should be able to make a progress plot. This might be something which needs Project coordination to assess what's the best way to present the progress.

Appendix

Here are just a few example scenarios for a user experience.

User case Example 1:

An aerogel tile is tested at one of the facilities for transmittance measurement at various wavelengths. This tile will have a unique ID and a corresponding individual item page with a predefined list by the DSL. This list of QA also has a range of acceptable values. If the measurement dont fall in this range it flags the entry (e.g with red color). After the user submits all the measurements It gets updated in the inventory as a tested tile with corresponding flag as good or bad.

User case Example 2:

The GEM foils for MPGD detectors are shipped from CERN to the module assembly facility. The user receives them and labels each foil with a unique ID. Then the user login via the web interface for this particular sub-system in the DB. This landing page presents the hierarchy of items. User goes to this particular item and updates it as received. Now anyone who has access to this particular sub-system's interface can see that so many foils have been received from CERN to a particular institute on a particular date by a particular person.

User case Example 3:

A photosensor tile of 10cmx10cm is tested at a particular site for the dark rate of every pixel. These rates are stored in an ASCII file with a predetermined format. The user goes to a particular QA page corresponding to this uniquely ID'ed item and uploads the file in the dark-rate row. Then the user goes to the "Evaluation of current data" tab where it can plot in the desired format. This Evaluation page also contains other 2-D data e.g. quantum efficiency maps, gain maps etc.

User case Example 4:

During the construction phase of the dRICH sub-system, they need to report the progress in terms of the total number of SiPMs available in house, tested good, and tested bad. DSL for dRICH goes to the "Database Activity" page and makes a selection based on the requirement. It pulls up a distribution with horizontal axis as /Days/months/years and vertical axis as the number of items with different colors with an option to compare wrt. the actual expectation by that day.

User case Example 5:

A user just joined a group and is given access to use the DB for a particular sub-system. But the new user is not familiar with this DB and is scared that some information can be rendered or deleted by mistake or simply not sure how to feed the information. In this case the user selects the "test DB" instead of "Real DB" at the login page. Test DB is just a clone of real DB which gets sync at a regular interval say every 2 weeks. In Test DB users can explore different options and try to familiarize themself before using the real DB.

User case Example 6:

Before building a module for a particular sub-system at an institution the user wants to look at the schematic or the instruction to follow a particular protocol. Each record describing a component contains references to relevant documentation (if applicable). This can include schematics, diagrams, testing and QA instructions and any other materials necessary to support the full life cycle of a component. The query and retrieval mechanisms should be flexible for DSL's in order to update them.

User case Example 7:

During the assembly of a sus-system user decided to put a particular component at the 3 O'clock position. User goes to this particular component and feeds the as-built location in the DB. So if the user looks at the location history of this item, it will tell who characterized this part when and where. What are the features and where it is located in the as-built detector.

User case Example 8:

At some point during the operations there is an issue with a particular component and it's hard to guess which item it is. The User reaches to this item and scans the unique id with a scanner or reads the unique code (if applicable). After going to the DB page, the user can enter this id in a search box which presents all the details for this item from procurement till the installation to understand the possible issue.