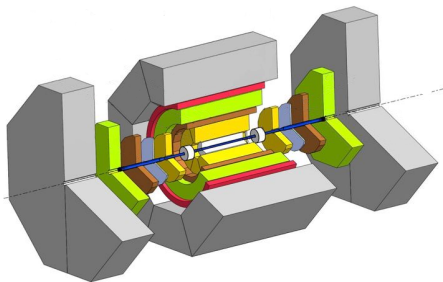


# Databases status update

Fedor Prokoshin

JINR, DLNP

November 7, 2024



- The SPD Experiment have to produce large amounts of data, both collected from the detector and simulated
- The processing of the experimental data requires a wide variety of auxiliary information from many systems
- Huge volumes of the detector condition and management data should be stored in the databases
  - should be used in every stage of data taking, processing and analysis
  - are essential at nearly every stage of data handling
  - for use in number of versatile applications each with its own requirements

- Databases can not be considered as the thing in itself, but as a part of complex information system that include
  - Data collection tools
  - Data transportation tools (messaging services, etc)
  - Application layer between the client and the database server
  - Client software including GUI
  - APIs for access from the production and analysis software
  - Supervisors and monitoring
- Databases and applications should be designed aiming for scalability, long term operation in the varying conditions, data flows and rates.
- Development and deployment would take quite a long period, some technologies may become obsolete or not available, others may emerge
- Information systems are related to each other and to the detector components, this defines the priorities in development

## Data

- Distributed Computing and Data Management
- EventIndex
- Physics Metadata IS

## Collaboration

- Personnel and Publication Databases

## Detector

- Conditions and Calibration Databases
- Monitoring Information Systems
- Logging and Bookkeeping
- Hardware and Mapping Database

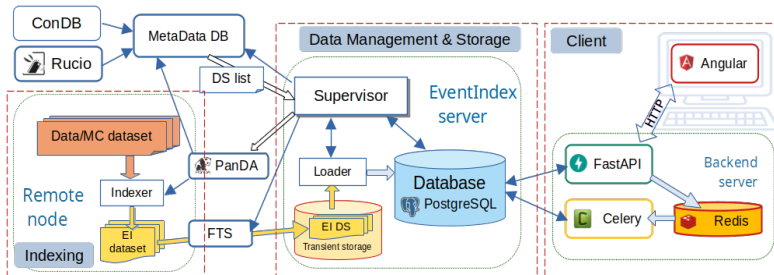
- Complete catalog of SPD events, real and simulated data
- An information system that should provide
  - retrieving event metadata by indexing data files on distributed storage
  - transfer of this information to the EI server
  - verifying of received data and loading to the database
  - access to information via universal API
  - interactive and asynchronous user interfaces.

## Event record content

- Event ID, Online Filter results, UID in the distributed storage of the RAW data file, UID of the dataset containing this file
- UIDs of files and datasets with reconstructed events (AOD)
  - Will be added when ready.
- Important event parameters for classification and selection (TBD).

# EventIndex: architecture

- The EventIndex employs micro-service architecture
  - Each component is a separate service operating independently
  - Each microservice can be updated and deployed independently
  - Provide flexibility, simplifies development and improves scalability
- The system consists of three main blocks:
  - Data indexing: collecting event metadata
  - Data management and storage system: import to the DB
  - The client part: API, Web interface and queries handling



# EventIndex: development progress

PostgreSQL DBMS is used for storage and processing of data

- The estimated data flow from 10s k up to 150 k events per second
- Various optimization methods were investigated to speed up the process of writing data to the DB.
- Using of asynchronous bulk loading showed sufficient performance

A program interface that uses the RESTful API is being developed

- The front-end part was developed using the Angular framework
- Server side employs a fastAPI: a light weighted asynchronous RESTful framework for Python
- For asynchronous task processing Redis and Celery were used.
  - Celery manages tasks that require long-term processing.
  - Redis acts as a message broker and temporary storage for task results.
- Users can launch long queries simultaneously, no need to wait

# Event Index: deploy

- The SPB Event Index project is configured for instant deployment using GitLab CI/CD (Continuous Integration/Continuous Delivery).
- CI/CD allows you to automate assembly, testing and deployment, ensuring fast delivery of updates to production.
- This approach minimizes manual operation time and reduces the probability of errors, making the deployment process faster and more reliable.
- The assembled images generated at the CI/CD stages are stored in the Gitlab Registry, which allows you to centralize management, track versions at the image level and control access rights.

More details on development are presented in paper [▶ Link](#), reports by Zamira Budtueva [▶ Link](#), Alan Gazzaev [▶ Link](#) and Richard Gurtsiev [▶ .docx](#)



# Event Index: further development

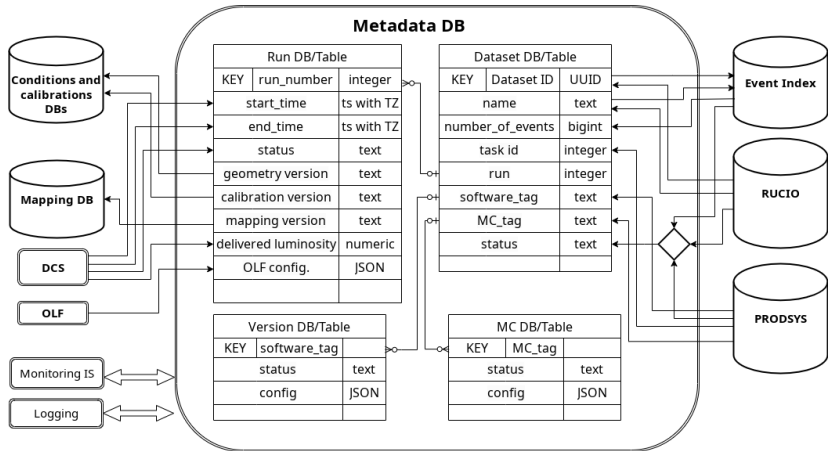
Supervisor manages collecting and import of the EventIndex data:

- Tracks the distributed storage system for a new datasets appearance
  - Launches as panda task PanDA of a indexing program that will go through the events and produce dataset with EventIndex information
  - Manages the transfer of data to the EventIndex server
  - Runs the loader program that fill the data to the database
  - Monitors the processes and collects information for logging
- 
- Datasets will be indexed on the server where they reside
    - The indexing task will be run by PanDA on Supervisor request
    - Alternatively, it may be initiated by production task on its completion
    - The data obtained will be transfered to the EI server via FTS
  - A program for indexing being created, assuming ROOT data format

- Contain information about
  - Datasets and data samples
  - Provenance chains of processed data linked to production tasks info
  - Data-taking runs
  - Cross-sections and configurations used for simulations
  - Online filter and luminosity information for real and simulated data.
- Should collect a great part of its information from other IS's and provide links to data for them
- Conventions for the runs and dataset naming have to be defined, as well as for software and MC configuration versioning
- The amount and scopes of information will be understood in the progress of experiment development

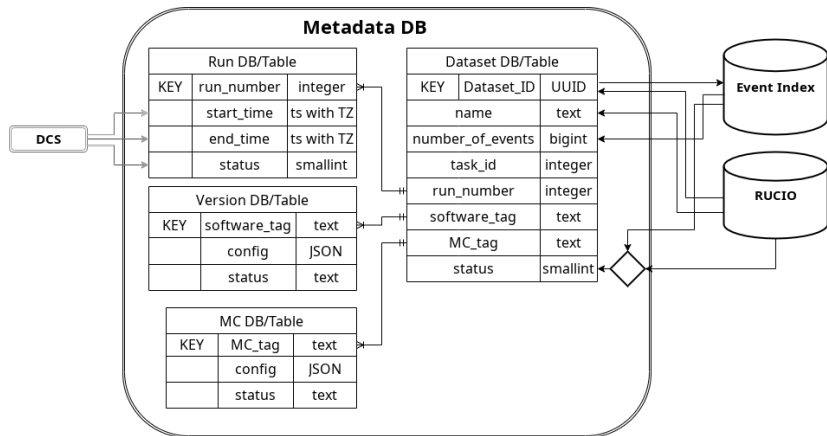
A conceptual prototype has been developed

# Physics Metadata: prototype



The conceptual scheme of Metadata Information System

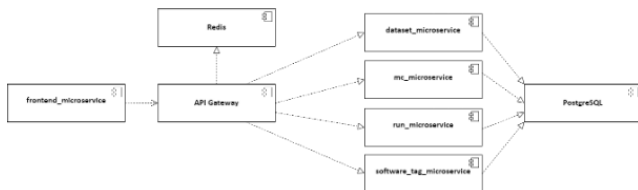
# Physics Metadata: prototype



The prototype of Metadata Information System

# Physics Metadata: prototype architecture

- The three-level microservice architecture is taken as a basis
- For each PostgreSQL schema a type of microservices will be created
- Each microservice consists of a structure of three layers: an API layer, a service layer, and a database layer
- The Gateway API pattern was used, which made it possible to distribute the load between microservices



Details are presented in report by Nikita Shkerin [▶ Link](#) and Rinat Korotkin [▶ Link](#)

# Database of personnel and documents

- About 400 people are currently participating in the SPD project,
  - number of participants is expected to grow close to experiment start
- In order to organize effective cooperation with the shared use of computing and other resources, it is necessary to have IS for
  - handling of a personnel and organizations data
  - support for working groups: membership, access rights,
  - accounting of the contribution (if implemented)
  - generating reports broken down by various parameters
- Procedures for creating, approving and editing related documents
  - Registration and changes of membership in the collaboration
  - Creating and editing lists of groups and privileges
  - Inclusion in the author's lists
- The resources of the working groups should use a single authentication and authorization system
  - An IAM service has been developed and already used for some tasks

# Database of publications and conferences

- SPD already produced some publication and conference reports
- An IS is necessary for preparing and publishing results
  - Tracking the progress of publications,
  - Organizing the exchange of messages between authors, reviewers and curators
  - Searching through documents
  - Tracking of SPD publications in external information systems.
  - Reports on the number of publications, broken down by authors, topics. . .
- Organization of presentations and reports at conferences and meetings:
  - Compiling a list of conferences and available reports
  - Organization of call for speakers and selection
  - Acceptance, review and approval of titles, abstracts and slides
  - Tracking the publication of proceedings

# Conditions and Calibration Data

Conditions data - non-event data representing the detector status

- Detector hardware conditions
- Detector calibrations
- Detector read-out conditions
- Detector alignments
- Physics calibrations
- Luminosity and polarization measurements

## Conditions data usage

- Subsystem calibration
- Online data processing, reconstruction and reprocessing
- User analysis



# Conditions Data organization

- Various pieces of information are heterogeneous both in data type and in time granularity
- The data should be organized by "Intervals of Validity" (IOV), which is the span in time over which that data is valid
- Can be recomputed later if the understanding of the detector behavior improves or the quality of the input data increases.
  - except for the detector and trigger configurations
- Careful versioning of groups of conditions data for production use cases is a critical item to guarantee reproducibility.
- Conditions data are typically written once and read frequently
  - read-rates up to several kHz must be supported for distributed computing use

Development will start later, when appropriate systems will be defined

# Hardware Database

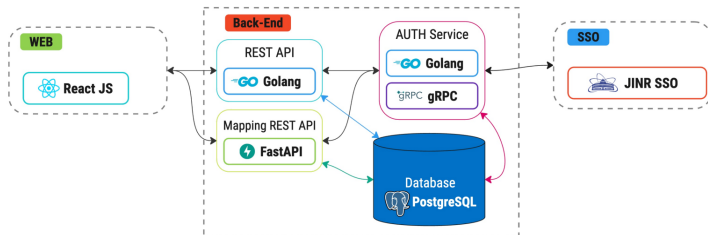
- A catalog of hardware components that SPD detector consist of.
- It should contain the information about the detectors and the electronic parts, cables, racks, and crates, as well as the location history of all items
- It include equipment models, provider, parameters and other (semi)permanent characteristics
- This should help in maintenance of the detector systems and especially helpful in knowledge transfer between team members.
- Filling of the hardware database should take place gradually, and updates will be rare
- The requirements for the speed of recording information in the database are low

# Mapping database

- The construction of the connection diagram and its changes will also be performed rarely (no more than once a week)
- The requirements for the speed of recording information in the database are low
- Mapping information may be required when processing each file.
  - It is possible that tens of thousands of processes will try to simultaneously access the system.
  - It is necessary to ensure their processing, avoiding database overload due to too high frequency of requests
- Due to the large number of elements in the system, it is almost impossible to build mapping manually
- For the elements involved in the transmission of digital signals, an automatic mapping procedure should be implemented
  - The element must issue a HW ID over the data channel in response to a special signal

# Hardware Database: architecture

- A prototype system is being developed
  - PostgreSQL for a data storage
  - Back-end providing endpoints for access through the REST API
  - Web front-end providing interface for fill, search and display data
  - JINR SSO for authorization, access can be requested from web front
- The architecture of the project is built on the basis of microservices
- A stack of modern platforms is used to ensure good performance, flexibility and ease of development and maintenance



# Hardware Database: Web Front

## Hardware Database


prof 

 Home

 All Data ▼

All groups

All devices

 Get Data ▼

NAME	CREATED_AT	UPDATED_AT	ACTIONS
AT5202	2024-10-30 00:53	2024-10-30 00:53	<a href="#">Show props</a>    
DT5202	2024-10-29 22:09	2024-10-29 22:09	<a href="#">Show props</a>    

< 1 >

# Hardware Database: groups

- Similar devices are organized in a groups
- Group typically describe some model of the device
  - For exmple: A5202 front end board
- For group, defined a set of properties, common for its devices
- Each property has data type
  - Data type can be numeric, string, boolean, date-time, IP and enumerated (set of values)
  - More data types can be implemented when needed
- Units of measurement, value ranges and defaults can be specified
- For example:
  - Output voltage. Type: decimal. Range: 20÷85. Default value: 29.0
  - Voltage Range. Type: enum. Options: 4.5, 2.5, DISABLED
  - Manufacture date. Type: date
- Groups can be cloned with all their properties

# Hardware Database: devices

- Device is the specific exemplar from the group
  - For example: A5202 front end board with S/N ABC123456.
- Each device has unique ID assigned to it,
- For the properties defined for the group, values can be specified.
- Values can be within range, or selected from enum.
- If no value specified, used default from group description
- For example:
  - Output voltage: 28.5 V. (29.0 if not specified)
  - Voltage Range: 4.5
- Each device has special property, *component ID*
  - specifies its *geographical* position on SPD
  - for example: SiPM board for 8-th sector of BBC
  - can be changed, as the same device may be used in different subsystems

# Hardware database: Tests and deploy

- Several levels of testing implemented for the REST API service:
  - Unit tests, Integration tests, Functional and E2E tests
- Each component of a hardware database project is isolated and can be developed, deployed, and updated independently of the others.
- For this and to simplify service management, containerization is used.
- Each service is packaged in a separate container with its dependencies and environment, making it independent of the underlying operating system and other elements.
- A working version of the system is installed on VM in the JINR Cloud.
- Also it can be deployed on on your linux machine .

See reports by Boris Dvizov [.docx](#) , Soslan Gussalov [.docx](#) and Rolen Magkaev [.docx](#)

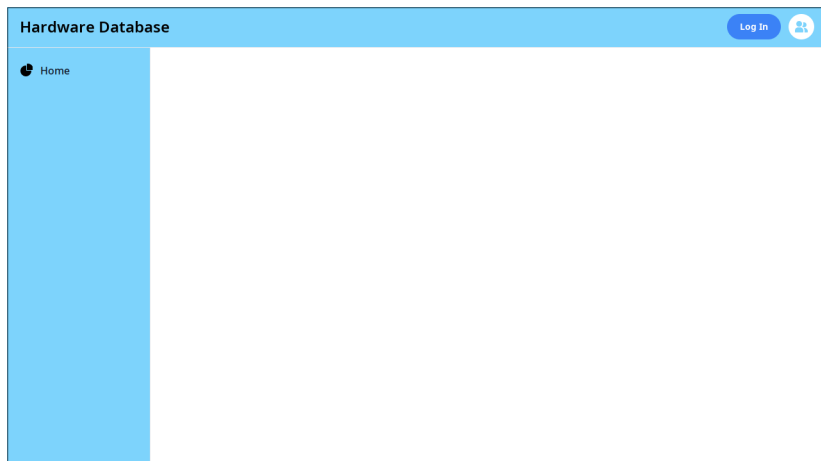




# BACKUP SLIDES

# Hardware Database: Autorisation

- Uses JINR SSO for authorization



# Hardware Database: Autorisation

- Uses JINR SSO for authorization



JINR Single Sign-On

Reminder: you have agreed to comply with the JINR computing rules

Sign in with a JINR account

User name:

Password:

Sign in

[Recovery password](#)

[How to get SSO login for user.](#)  
[Registration SSO service and application.](#)

# Hardware Database: Autorisation

- After authorising with SSO, user can send request for access
  - Currently via e-mail. Use of JINR Help desk proposed for production.

The screenshot displays the 'Hardware Database' application interface. At the top, a light blue header contains the text 'Hardware Database' on the left and 'prof' with a user profile icon on the right. A vertical sidebar on the left has a 'Home' link with a house icon. The main content area features a white box with a red warning icon at the top. Below the icon, the text reads: 'You are currently a user with zero permissions. Contact your administrator to increase the access level.' Underneath this text is the instruction 'Select the access level' followed by a dropdown menu labeled 'Select role'. At the bottom of the white box is a blue button labeled 'Request rights'.

# Hardware Database: Autorisation

- Now there are 4 levels of access
  - **Zero**: only to send request to admin
  - **Restricted**: Read-only, Only view information
  - **Full**: Can both view and fill data
  - **Admin**: Can change access levels for users

The screenshot displays the 'Hardware Database' web interface. At the top left, the title 'Hardware Database' is shown. On the top right, the user is identified as 'prof' with a user icon. A left sidebar contains a 'Home' link with a magnifying glass icon. The main content area features a central warning box with a red exclamation mark icon. The text inside the warning box reads: 'You are currently a user with zero permissions. Contact your administrator to increase the access level.' Below this text is a dropdown menu labeled 'Select the access level' with the option 'Select role' visible. A blue button labeled 'Request rights' is positioned below the dropdown. To the right of the warning box is a user information panel containing the following details: 'id: 6647', 'fio: Prokoshin Fedor Valerevich', 'session: 35', and 'role: Zero access'. This panel includes three buttons: a red 'Logout' button, a blue 'Refresh' button, and a red 'Logout all' button.

# Hardware Database: Autorisation


- You have to wait for Admin grant you access



The screenshot shows a web application interface. At the top, there is a light blue header bar with the text "Hardware Database" on the left and "prof" next to a user profile icon on the right. Below the header, on the left side, is a vertical blue sidebar with a "Home" button. The main content area is white and contains a centered message box with a red exclamation mark icon and the text: "You have sent an enquiry to the administration. Wait for a reply."

# Hardware Database: Autorisation

- When you receive access, you see instructions

The screenshot displays the 'Hardware Database' application interface. On the left is a blue navigation menu with three items: 'Home', 'All Data', and 'Get Data'. The main content area is titled 'How to Use the Hardware database' and contains two sections: 'Navigation Menu' and 'Top Right Information Container'. The 'Navigation Menu' section explains the menu's purpose and lists three options: 'Home', 'All Data', and 'Get Data'. The 'Top Right Information Container' section describes the user information displayed in the top right corner, which includes the user's ID, full name, session number, and role, along with 'Logout', 'Refresh', and 'Logout all' buttons.

**Hardware Database** prof 

**Home**  
**All Data**   
**Get Data** 

## How to Use the Hardware database

### Navigation Menu

On the left side of the screen, you will see the **Navigation Menu**. This menu helps you move the system. It includes the following options:

- **Home:** This option will take you back to the home screen of the service.
- **All Data:** Here you can view a complete list of all available data groups or devices.
- **Get Data:** Select this option to retrieve specific data from existing groups or devices.

### Top Right Information Container

In the top right corner of the screen, you will find a container that displays key user-specific information. This container helps you manage your session and provides quick access to important actions, such as logging out or refreshing your session. The information displayed includes:

- **ID:** The unique identifier for your account.
- **Full Name (FIO):** Displays your full name.
- **Session:** Shows the current session number.
- **Role:** Displays your user role within the system (e.g., Restricted access).

**Additionally, the container has the following options:**

id: 6647  
fio: Prokoshin Fedor Valerevich  
session: 35  
role: Full access

**Logout** **Refresh**

**Logout all**



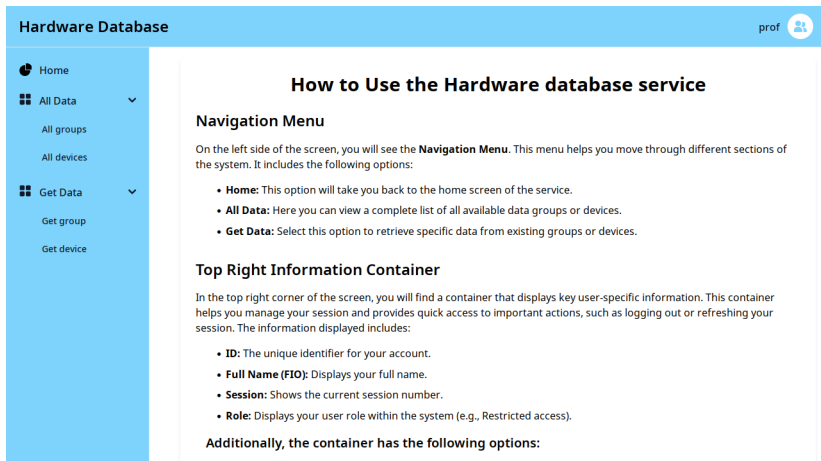
# Hardware Database: workflow

The typical workflow for filling database is the following:


- User creates group, specifies its name and adds properties
  - For each specifies its name and datatype
  - Optionally: units, min, max and default values
  - For enums, specify list of values
- Groups can be cloned, properties can be added or changed
  - This can be used for creating groups for similar models,
  - For example: A5204 front end board
- Groups can be saved to JSON file or read from it
- After creating a group, user can create components based on this group, and fill the values of the parameters.
- Cloning and creation of series of components have to be implemented

# Hardware Database: filling groups

- To see groups information, click on the 'All groups'



The screenshot shows the 'Hardware Database' application interface. At the top, there is a light blue header with the text 'Hardware Database' on the left and 'prof' next to a user profile icon on the right. A vertical navigation menu is on the left side, containing the following items: 'Home' (with a home icon), 'All Data' (with a grid icon and a dropdown arrow), 'All groups' (with a dropdown arrow), 'All devices' (with a dropdown arrow), 'Get Data' (with a grid icon and a dropdown arrow), 'Get group' (with a dropdown arrow), and 'Get device' (with a dropdown arrow). The main content area has a title 'How to Use the Hardware database service' and a sub-section 'Navigation Menu'. Below this, it explains that the navigation menu is on the left and lists three options: 'Home', 'All Data', and 'Get Data'. Another sub-section 'Top Right Information Container' explains that it shows user-specific information and lists three items: 'ID', 'Full Name (FIO)', and 'Role'. Finally, it states that the container has additional options.

**Hardware Database** prof 

## How to Use the Hardware database service

### Navigation Menu

On the left side of the screen, you will see the **Navigation Menu**. This menu helps you move through different sections of the system. It includes the following options:

- **Home:** This option will take you back to the home screen of the service.
- **All Data:** Here you can view a complete list of all available data groups or devices.
- **Get Data:** Select this option to retrieve specific data from existing groups or devices.

### Top Right Information Container

In the top right corner of the screen, you will find a container that displays key user-specific information. This container helps you manage your session and provides quick access to important actions, such as logging out or refreshing your session. The information displayed includes:

- **ID:** The unique identifier for your account.
- **Full Name (FIO):** Displays your full name.
- **Session:** Shows the current session number.
- **Role:** Displays your user role within the system (e.g., Restricted access).

Additionally, the container has the following options:

# Hardware Database: filling groups

- Now no groups, create one

The screenshot displays the 'Hardware Database' application interface. At the top, a light blue header bar contains the title 'Hardware Database' on the left and the user profile 'prof' with a circular icon on the right. A vertical sidebar on the left side of the page lists navigation options: 'Home' (with a home icon), 'All Data' (with a grid icon and a dropdown arrow), 'All groups', 'All devices', 'Get Data' (with a grid icon and a dropdown arrow), 'Get group', and 'Get device'. The main content area on the right is currently empty, displaying the text 'There is no data...' and a grey button labeled 'Create group'.

# Hardware Database: filling groups

- Type in group name

The screenshot shows the 'Hardware Database' application interface. On the left is a dark teal sidebar with navigation options: Home, All Data (with a dropdown arrow), All groups, All devices, Get Data (with a dropdown arrow), Get group, and Get device. The main content area has a dark teal header with the text 'Hardware Database' and a user profile 'prof' with a circular icon. Below the header, the text 'There is no data...' is displayed above a 'Create group' button. A white modal dialog box titled 'Create new group' is centered on the screen. It contains a blue 'Create group' button and a grey 'Upload File' button. Below these is a text input field labeled 'Group Name \*' with a vertical cursor. At the bottom of the dialog are a red 'Cancel' button and a blue 'Create Group' button.

# Hardware Database: filling groups

- lets specify first property: Number of channels, and click Create group

The screenshot shows the 'Hardware Database' application interface. A modal dialog titled 'Group Properties' is open, allowing the user to define a new property for a group. The dialog contains the following fields and controls:


- Property Name \***: A text input field containing the value 'Inputs'.
- Property Type \***: A dropdown menu currently set to 'int'.
- Units**: An empty text input field.
- Min Value**: An empty text input field.
- Max Value**: An empty text input field.
- Default Value**: A text input field containing the value '64'.
- Remove**: A red button located below the Default Value field.
- Add Property**: A green button located below the dialog.
- Cancel**: A red button located at the bottom left of the dialog.
- Create Group**: A blue button located at the bottom right of the dialog.

The background interface includes a sidebar with navigation options: Home, All Data (with a dropdown arrow), All groups, All devices, Get Data (with a dropdown arrow), Get group, and Get device. The top right corner shows a user profile icon labeled 'prof'.

# Hardware Database: filling groups





- Now the group is created. Click Show props

## Hardware Database

prof 

- Home
- All Data ▼
  - All groups
  - All devices
- Get Data ▼
  - Get group
  - Get device

Create group


NAME	CREATED_AT	UPDATED_AT	ACTIONS
DT5202	2024-10-29 22:09	2024-10-29 22:09	<a href="#">Show props</a>    

< 1 >

# Hardware Database: filling groups



- You can see property. Let's add more

## Hardware Database

prof 

- Home
- All Data ▼
  - All groups
  - All devices
- Get Data ▼
  - Get group
  - Get device

← Add a new property 📄 Group as a json

NAME	UNITS	PROPERTY_TYPE	MIN	MAX	DEFAULT_VALUE		
Inputs		int			64		

# Hardware Database: filling groups

- String.

The screenshot shows the 'Hardware Database' application interface. A modal dialog titled 'Add New Property' is open in the center. The dialog contains the following fields and controls:

- Group ID:** A text input field containing the value '8'.
- Name \*:** A text input field containing the value 'S/N'.
- Property Type \*:** A dropdown menu currently set to 'string'.
- Units:** An empty text input field.
- Default Value:** An empty text input field.
- Buttons:** 'Cancel' and 'Save' buttons at the bottom right of the dialog.

The background interface is dimmed and shows a table with columns 'NAME', 'UNIT', 'MAX', and 'DEFAULT\_VALUE'. A row with 'Inputs' is visible under the 'NAME' column. Navigation buttons 'Add a new property' and 'Group as a json' are visible above the table. A sidebar on the left contains navigation options like 'Home', 'All Data', 'All groups', 'All devices', 'Get Data', 'Get group', and 'Get device'. The top right corner shows a user profile 'prof'.



# Hardware Database: filling groups

- Enum. Specify possible values

Hardware Database

Home

← Add a new prop

Log In

### Add New Property

Group ID  
8

Name \* status Property Type \* enum

Units

Default Value

Variants \*

Add Variant

Good Remove

Bad Remove

Ugly Remove

Cancel Save

MAX	DEFAULT_VALUE		
64		✍	🗑
15		✍	🗑
		✍	🗑

# Hardware Database: filling groups

- Integer. Minimal and maximal values specified

The screenshot displays a web application interface for a 'Hardware Database'. A modal window titled 'Add New Property' is open, allowing the user to create a new property. The form contains the following fields:

- Group ID: 8
- Name: HG\_Gain
- Property Type: int
- Units: (empty)
- Min: 1
- Max: 63
- Default Value: (empty)

The background shows a table with the following columns: NAME, MAX, and DEFAULT\_VALUE. The table contains the following data:

NAME	MAX	DEFAULT_VALUE
Inputs	64	
S/N		
Shaping Time Fast	15	
production date		
status		
Shaping Time Slow		

# Hardware Database: filling groups

- After adding some more properties

## Hardware Database

Log In

Home

← Add a new property 📄 Group as a json

NAME	UNITS	PROPERTY_TYPE	MIN	MAX	DEFAULT_VALUE		
Inputs		int			64		
S/N		str					
Shaping Time Fast	ns	float			15		
production date		datetime					
status		enum					
Shaping Time Slow	ns	enum					
HG_Gain		int	1	63			
eth		ip			192.168.50.4		

# Event Index: further development

- User authorization and authentication, group access policies
- API development (REST, Python, C++, ...)
- Further optimization of user request processing, improving priority management depending on the volume of requested data
- Development of communication with other information systems for organising of indexing of data (Rucio, PanDA) and collecting their results (FTS)
- Supervisor - software for managing, collecting and importing data
- Development of component monitoring system, with graphical representation of data based on popular platforms (Grafana, etc.)

---

The implementation of these tasks will be carried out in parallel with the development of other Information Systems of the experiment.

# Monitoring and logging

Monitoring information system yet have to be developed

- Will become necessary as detector components become ready
- Will use various source of data from the subsystem and other databases
- Data can be transferred in JSON format with variable schema
  - Only few mandatory fields required (source id, time stamp, etc..)
- Data transfer through the HTTP requests can be used
- Time series database should be used as a back-end
- Commonly used solution like Grafana for use for visualization of data
- Solutions that worked well on the other NICA experiments should be used

Monitoring and logging can be shared between the NICA experiments