**What is TGenBase?**
- Virtual DB engine
- Made for scientists, not programmers
- Rich data visualization in Web
- Language bindings
- Templated code generation

**Key Aspects**
- Visual data description
- Allows storage of user-defined data, ROOT objects, binary data
- Special logics in addition to CRUD (create, read, update, delete)
  - Versioned insert-only - nothing is lost
  - Audited - keeps track of user activity

**Visual data description**
- Describe data classes in Web UI
- Establish relations between classes (one-to-one, one-to-many, many-to-many)
- Many built-in class property types, including arrays, collections, (ROOT-)objects using serialization
- Properties can have custom default values, validation rules, semantic subtypes (URI, LaTeX, DOI)
- Dictionary properties allow for easy ID-Name lookup
- Static data description for initial DB seeding
- Saving and loading of projects, fetching from git

- Same description used to generate the database layout, server to host data, and clients to fetch data objects (C++, Python, LabVIEW)
- Rich data visualization in a full-fledged CMS

**Data description is a Schema**
- Defines the contract between the client and server
- Client and server exchange and check schema versions
- Schema evolution based on migrations

**Schema development workflow**
- Local dev, describe data format, prepare user data export
- Deployment: requires published in central schema store
- Central database server: cmsdbn.chm.gsi.de/TGenBase/SchemataStore

**Data visualization**
- A Content Management System (CMS) is generated from schema
- Allows for navigation between different classes and their relations
- Provides search functionality by single fields or using a dynamic query builder interface
- Data display customization
- Data visualization: ROOT objects (JSROOT), images, PDFs, spreadsheets

- Operator's workspace to create new entries and edit existing, upload files, etc.
- Admin panel to customize server behavior, manage users, etc.

**Server**
- Production server runs inside docker container configured with docker-compose using Laradock as a recipe book
- Provides standard RESTful API to access data by JSON message exchange with clients using token authentication
- Provides configurable role-based user permission management, user import from LDAP directories
- Caches data for frequent access in key-value storages
- Communicates with underlying DB Management Systems
- Ongoing work towards push-to-deploy
- Local development server has minimum requirements and is bootstrapped with a shell script, runs on Linux and macOS

Uses in CBM experiment at future facility FAIR are
- Detector component, logistics, QA data storage
- Parameter, conditions, configuration management

**Production DB server software stack**
- HTTPS Thin Clients
- User Management
- REST API
- Fast Caching
- DB Layer

Currently 6 different instances are hosted for CBM needs