Generic Data Module (GDM)

The Generic Data Module is a graphical user interface on top of a MySQL / MariaDB database that allows the users to browse and edit data without any SQL knowledge.

Depending on the data model's complexity the user is expected to have a minimum of knowledge on how tables are connected in relational databases.

User Story 1 – Data migration process

a) Jane is a curator in the herpetological collections in one of the major natural history museums in Europe. She is used to manage her data in several Excel spreadsheets and doesn't know much about databases, although she understands the advantages of having the data of all collections in one central collection management system, e.g. DINA Web which is the preferred system in her institution.
In collaboration with the database manager John she is asked to migrate her data to DINA Web. In preparation of this migration John identifies, that 90% of Jane's data can be accommodated in the DINA system, but special information on the herpetological specimens (approx. 10% of her spreadsheet columns) are not yet covered by the underlying DINA data model.
These 10% of special information cause several issues for John, because Jane doesn't want to lose this portion of data:

1) Before the migration the DINA data model needs additional fields in order to accommodate 100% of the Jane's data. But this modification needs to be approved by the DINA consortium in order to add value to the whole community.
   Unfortunately, the complete migration of Jane's data must wait until this process is done.
2) The accommodation of individual requirements would extend the DINA data model to a vast amount of special fields that are rarely used across the collections and institutions.
3) If John decided to migrate only the 90% of the data, Jane would need to continue managing the rest 10% in Excel spreadsheets. Thus, the data would be disconnected and would raise the risk of data loss, redundancy and/or loss of data integrity.
4) Not least, John would have additional effort to compare and integrate the 10% of data as soon as any system would be available to accommodate them in a later stage.

In order to solve the above-mentioned issues John creates an SQL file that contains the (arbitrary) structure needed to accommodate the 10% of Jane's data. He prepares the data the same way he would do if the DINA system would cover the special requirements. Thus, he doesn't have to redo the data wrangling, structuring and quality checks in the future again.
He deploys an instance of the Generic Data Module (GDM) in which he can simply use his SQL file as an initial Liquibase changelog file. He dumps the 10% of data plus the respective primary keys into the database. Thus, Jane can access the GDM interface to browse and edit her data that could not (yet) be accommodated in the collection management system in a relational database instead of using Excel. As

John used the customizable style sheets in the GDM to modify the interface's layout, Jane doesn't even notice that the GDM is separated from her new collection management system DINA Web.

b) In order to preserve the data integrity, John takes this approach one step further an creates a read-only cross-schema MySQL VIEW that joins the tables from the DINA Web to the GDM data structure. Thus, Jane can see the 10% of the data in the (full) context of the collection data whenever she needs and she doesn't get the feeling that the 10% of her data got "lost" during the migration.

c) Jane wants to print labels with some of the data that is not yet in the DINA core modules, John's decision to use the GDM enables him to realize Jane's requirement as the data can be accessed programmatically with GDM's DINA-compliant JSON-API (e.g. with the DINA Labels Module). Thus, the GDM makes Jane's 10% special data an integrated part in the DINA Web.

d) After a while the systems and data model evolved as many other collections had similar requirements to add some fields that Jane got used to manage in the GDM instead of Excel.
Now, John can move most of the data from Jane's GDM instance to the DINA core data model. He is lucky, that he atomized and normalized the data already in the initial migration from Jane's Excel files. So, John migrates the data from the GDM to the DINA modules by using the JSON API. If there was nothing left in the GDM John could simply remove the GDM instance.

User Story 2 – Digitization workflows & research data

a) Jane wants to digitize some specimens in her collection. She develops a digitization workflow together with a vendor company of 3D-scanners. The workflow is designed to suit the needs for her taxonomic research. During the digitization a diverse metadata and research data (e.g. measurement of the specimen's spatial dimensions) should be recorded in addition to the collection data stored in the DINA web system. The media data and its metadata will be stored in a media database, that is used all over Jane's institution, also outside the collection department. The research data will be stored in a database that was shipped by the company of the 3D-scanner. For the metadata on the digitization workflows (e.g. status, digitization method, calibration) and object treatments (e.g. cleaning, perspective), she doesn't have any database to store the information.

The database manager John will set up a GDM instance for Jane's digitization metadata in order to provide her with a simple database and interface for data entry. Therefore, he talks to Jane to agree on the structures and fields needed to accommodate the information. Besides several text fields for the metadata, they agreed on a dedicated field for the collection number in order to reference the specimens in the workflow metadata, so no direct connection to the DINA web system was needed. During the digitization Jane realizes that three fields need to be added to the table structure in the GDM. So John creates a MySQL dump of the database for backup and uses the dump file as a new Liquibase changeset containing not only the structure, but also all data. In an additional changeset file he adds the three missing columns to the respective tables and updates the database of the existing GDM by applying his Liquibase changesets.

b) After the digitization project ended, Jane exports her research data from the company's software as a CSV file for her analyses, publication, re-use and archiving. Therefore, John sets up another GDM

instance in order to import the CSV file to an appropriate data structure and to make the data read-only accessible. He exposes the URL to the internet, creates a Digital Object Identifier (DOI) and redirects the DOI handle to this URL. Meanwhile, Jane analyzed the data and published a research paper in which she cites John's data publication instead of submitting the data as supplemental material in PDF format to the publisher.

User Story 3 – New database modules

The database manager John was asked to support all kinds of data publications in his institution as he did for the collection manager Jane the other day. He uses the DataCite services in order to assign DOIs to different kinds of data packages that is meant to be published by the scientists in his institution. These data packages consist for example of media files, metadata, collection data, measurements, bibliographic data, time series etc.
However, all data publications have in common, that DataCite requires (at least a minimal set of) metadata about the published data package (e.g. author, abstract, resource type etc.). When John didn't do the DOI assignment too often, he created the DataCite Metadata XML manually in his favorite text editor and uploaded the file to the DataCite Metadata Store. But now he realizes that he needs to manage the metadata locally in order to keep track of the metadata and their updates.

So, John sets up a GDM instance, roles out a Liquibase changelog file in order to create a MySQL data structure that is based on the DataCite Metadata Schema. He includes several relations between tables by defining foreign key constraints in order to cover the controlled vocabulary (e.g. terms for resource types) provided in by DataCite Metadata Schema.

The GDM provides John and his colleagues with a simple interface to enter and update metadata and keep overview of the metadata uploaded to DataCite. By using the GDM API John has several options to develop helper scripts in order to facilitate the metadata upload and the respective DOI registration.

As the GDM is fully compliant to the DINA API guidelines and no DINA module for managing metadata of data publications has been developed yet, John starts programming a DINA user interface for this specialized purpose and uses the GDM as a (temporary standalone) backend in order to create dedicated DINA module.