Upgrading your GSRS from 2.x to 3.0

As of 2 March 2022

# Introduction

GSRS version 3.0 is a major upgrade from earlier versions. The architecture of the system has been updated; the user interface has changed, and new modules that offer vastly new capabilities are available.

## Major factors of the upgrade to GSRS 3.0:

* Microservice deployment. Individual domain services can be added independently.
* The older Play! Framework has been replaced by Spring Boot v 2.4.5
* Added support for Java 11
* Improvements to facet views
* Advanced Search page lets you find your data faster
* New services are available for management of
  + Regulatory Applications
  + Clinical Trials
  + Products
  + Impurities
  + Adverse Events

# The Upgrade Process

The good news is that you can migrate your GSRS installation from 2.x to 3.0 with no loss of data.

The general update process to migrate a 2.x GSRS to be 3.0 requires the following steps:

1. Install GSRS 3.0.
2. Configure GSRS 3.0 to point to the same database as your 2.x instance.
3. Port any configuration customizations that may were active in your 2.x installation to the new 3.0 configuration files (in particular, the substances application.conf file).
4. Perform data maintenance operations.

## Step 1: Install GSRS 3.0

First, install a new GSRS 3.0 instance as explained in the README from this GitHub repo: <https://github.com/ncats/gsrs3-main-deployment>. Note: GSRS 3.0 allows for many different microservices and several deployment pipelines so please consider those parts of the set-up process are relevant to you.

## Step 2: Configure GSRS 3.0 to point to the same database as your 2.x instance

GSRS 3.0 is backwardly compatible with 2.x databases but the syntax used for database connections within configuration files has changed. [2.x used the Ebean ORM technology while 3.0 uses Spring and Hibernate as an ORM.]

The main configuration file used for substances (and other entity microservices) is called application conf, found in the resources folder of the deployed software (WAR file or clone from GitHub). The database connection will be specified within the application.conf file.

Please see the Database Connection requirements document for details.

## Step 3: Port any configuration customizations that may were active in your 2.x installation to the new 3.0 configuration files

Most deployments of GSRS require certain specific configuration properties. Among commonly configured properties are the server hostname, logging level, authentication strategy, search index locations, cache size, Molwitch implementation, chemical rendering style and unique code / “approvalID” generation. In addition, there are a variety of settings related to business logic that extend or modify GSRS functionality.

The syntax used for specifying some configuration property keys and values has changed in 3.0. however, many of the configuration properties supported in 2.x are still supporting in the application.conf files for 3.0. The core expected settings can be found in the core README files as well as the example application.conf files found in the repositories. An overview of some of the configuration changes can be found in the document titled **“Configuration file changes from 2.x to 3.x”.**

## Step 4: Perform data maintenance operations

The data definitions from GSRS 2.x should work with GSRS 3.0 but updating to be compatible with the newest code require a couple of extra steps. After successfully launching the gateway, frontend and substance services and ensuring the substance service can connect to the database, you should perform the following steps:

|  |
| --- |
| Run these 3 Scheduled Jobs from the GSRS Admin Panel’s Schedule Job in this order:   1. *Regenerate structure properties collection for all chemicals in the database*   This job calculates a set of values used when searching chemical structures, which has slightly changed between 2.X and 3.0 releases.   1. *Re-backup all Substance entities*   This job copies full JSON representations of each substance into a ‘backup’ table within the database, and is used to facilitate quick retrieval and indexing.   1. *Reindex all core entities from backup tables*   This job creates the Lucene index used to speed up searches for data and structures.  The index facilitates searching of text as well as structure data |

Once you have done this, do not continue to register data into the same database from GSRS 2.x and 3.0, but exclusively use a single GSRS 3.0 application instance to make database changes.

# Customizations

As noted elsewhere, GSRS may be customized by creating your own

* Index Value Makers – adding fields to the search index to make searches faster
* Validators – custom business rules for your organization’s data
* Entity Processors – data management that takes place automatically when new entities are registered, or existing entities updated
* Exporters – create your own data files for reporting, archiving, etc.
* Scheduled Jobs – created processing that takes place according to a fixed schedule or as needed.

If you have created one or more of the above server-side customizations for GSRS 2.x, some work may be necessary to port your customization to GSRS 3.0. An example of what can be expected from this kind of migration can be found in the guide titled “Migrating Scheduled Tasks from GSRS 2 to GSRS 3”.

If you have created any customizations of the original GSRS Scala front end, a substantial amount of work may be necessary to create a new version of your customization within the Angular front end of GSRS 3.0. Doing this may require both some changes to the frontend configuration file as well as some custom cards or a custom fork from the GSRSFrontend codebase. The frontend code can be compiled and rebuilt as a stand-alone war file based on the information described in the frontend section of the <https://github.com/ncats/gsrs3-main-deployment> repository.