INCEpTION Administrator Guide

The INCEpTION Team

Version 0.12.2

Table of Contents

This guide covers handling INCEpTION from an administrator’s perspective.

Installation

You can run INCEpTION on any major platform supporting Java, i.e. Linux, macOS or Windows. However, we do not provide explicit for setting up a production-ready instance of each of these platforms.

This guide assumes Debian 9.1 (Stretch). It may also work on Ubuntu with some modifications, but we do not test this. Instructions for other Linux distributions and other platforms (i.e. macOS and Windows) likely deviate significantly.

It is further assumed that the user **www-data** already exists on the system and that it shall be used to run the application.

All commands assume that you are logged in as the **root** user.

If you cannot log in as root but have to use `sudo` to become root, then the recommended way to do that is using the command `sudo su -`.

System Requirements

Table 1. Requirements for users

Browser	Chrome or Safari

Table 2. Requirements to run the standalone version

Java Runtime Environment	version 8 or higher

Table 3. Requirements run the server version

Java Runtime Environment	version 8 or higher
Apache Tomcat	version 8.5 or higher (Servlet API 3.1.0)
MySQL Server	version 5 or higher

Install Java

You can install an Oracle Java 8 JDK using the following commands.

```
$ apt-get update
$ apt-get install dirmngr
$ echo "deb http://ppa.launchpad.net/webupd8team/java/ubuntu trusty main" | tee
/etc/apt/sources.list.d/webupd8team-java.list
$ echo "deb-src http://ppa.launchpad.net/webupd8team/java/ubuntu trusty main" | tee -a
/etc/apt/sources.list.d/webupd8team-java.list
$ apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys EEA14886
$ apt-get update
$ apt-get install oracle-java8-installer
$ apt-get install oracle-java8-set-default
```

Application home folder

The INCEpTION home folder is the place where INCEpTION's configuration file `settings.properties` resides and where INCEpTION stores its data. Mind that if you are using a MySQL database server (recommended), then INCEpTION also stores some data in the MySQL database. This is important when you plan to perform a backup, as both the home folder and the database content need to be included in the backup.

Now, let's go through the steps of setting up a home folder for INCEpTION and creating a configuration file instructing INCEpTION to access the previously prepared MySQL database.

- Create INCEpTION home folder. This is the directory where INCEpTION settings files and projects (documents, annotations, etc.) are stored

```
$ mkdir /srv/inception
```

- Edit `/srv/inception/settings.properties` to define the database connection as well as internal backup properties:

```
database.dialect=org.hibernate.dialect.MySQL5InnoDBDialect
database.driver=com.mysql.jdbc.Driver
database.url=jdbc:mysql://localhost:3306/inception?useSSL=false&serverTimezone=UTC
database.username=inception
database.password=t0t4llYSecreT

# 60 * 60 * 24 * 30 = 30 days
backup.keep.time=2592000

# 60 * 5 = 5 minutes
backup.interval=300

backup.keep.number=10
```

- Fix permissions in INCEpTION home folder

```
$ chown -R www-data /srv/inception
```

Database

INCEpTION uses an SQL database to store project and user data.

INCEpTION uses by default an embedded HSQLDB database. However, we recommend using the embedded database only for testing purposes. For production use, we recommend using a MySQL server. The reason for this is, that:

- some users have reported that HSQLDB databases may become corrupt when the computer crashes (note that this could probably also happen with MySQL, but we did so far not have any reports about this);
- most INCEpTION developers use MySQL when running INCEpTION on their servers;
- in the past, we had cases where we described in-place upgrade procedures that required performing SQL commands to change the data model as part of the upgrade. We promise to try avoiding this in the future. However, in case we offer advice on fixing anything directly in the database, this advice will refer to a MySQL database.

We try to keep the data model simple, so there should be no significant requirements to the database being used. Theoretically, it should be possible to use any JDBC-compatible database after adding a corresponding driver to the classpath and configuring INCEpTION to use the driver in the `settings.properties` file.

MySQL

For production use of INCEpTION, it is highly recommended to use a MySQL database. In this section, we briefly describe how to install a MySQL server and how to prepare it for use with the application.

Prepare database

- Install MySQL

  ```
  $ apt-get install mysql-server
  ```

- make sure your MySQL server is configured for UTF-8. Check the following line is present in `/etc/mysql/mariadb.conf.d/50-server.cnf` (this is specific to Debian 9; on other systems the relevant file may be `/etc/mysql/my.cnf`):

  ```
  character-set-server = utf8
  collation-server     = utf8_bin
  ```

- also ensure the default settings for client connections to are UTF-8 in `/etc/mysql/mariadb.conf.d/50-client.cnf` (again Debian 9; likely in `/etc/mysql/my.cnf` on other systems)

```
default-character-set = utf8
```

- login to MySQL

```
$ mysql -u root -p
```

- create a database

```
mysql> CREATE DATABASE inception DEFAULT CHARACTER SET utf8 COLLATE utf8_bin ;
```

- create a database user called `inception` with the password `t0t4llYSecreT` which is later used by the application to access the database (instructions for `settings.properties` file below).

```
mysql> CREATE USER 'inception'@'localhost' IDENTIFIED BY 't0t4llYSecreT';
mysql> GRANT ALL PRIVILEGES ON inception.* TO 'inception'@'localhost';
mysql> FLUSH PRIVILEGES;
```

For production use, make sure you choose a different, secret, and secure password.

Configuration options

This section explains some settings that can be added to the `database.url` in the `settings.properties` file when using MySQL. Settings are separated from the host name and database name with a `?` character and multiple settings are separated using the `&` character, e.g.:

```
database.url=jdbc:mysql://localhost:3306/inception?useSSL=false&serverTimezone=UTC
```

To suppress the warning about non-SSL database connections with recent MySQL databases, append the following setting to the `database.url`:

```
useSSL=false
```

Recent MySQL drivers may refuse to work unless a database server timezone has been specified. The easiest way to do this is to add the following setting to the `database.url`:

```
serverTimezone=UTC
```

If you plan to use UTF-8 encoding for project name and tagset/tag name, make sure either of the following settings for MySQL database

- in the `settings.properties` file, make sure that `database.url` includes

```
useUnicode=true&characterEncoding=UTF-8
```

- change the `my.conf` MySQL databse configuration file to include the following line

```
character-set-server = utf8
```

HSQLDB (embedded)

INCEpTION displays a warning in the user interface when an embedded database is being used. It is not recommended to used an embedded database for various reasons:

- HSQLDB databases are known to run a risk of becoming corrupt in case of power failures which may render the application inaccessible and your data difficult to recover.
- In very rare cases it may be necessary to fix the database content which is more inconvenient for embedded databases.

In case that you really want to run INCEpTION with an embedded database in production, you probably want to disable this warning. To do so, please add the following entry to the `settings.properties` file:

```
warnings.embeddedDatabase=false
```

Running via embedded Tomcat (JAR)

The INCEpTION standalone JAR with an embedded Tomcat server and can be easily set up as a UNIX service. This is the recommended way of running INCEpTION on a server.

The instructions below expect a Debian Linux system. Details may vary on other OSes and Linux distributions.

Installing as a service

To set it up as a service, you can do the following steps. For the following example, I assume that you install INCEpTION in `/srv/inception`:

- Copy the standalone JAR file `inception-app-standalone-0.12.2.jar` to `/srv/inception/inception.jar`. Note the change of the filename to `inception.jar`.
- Create the file `/srv/inception/inception.conf` with the following content

  ```
  JAVA_OPTS="-Djava.awt.headless=true -Dinception.home=/srv/inception"
  ```

- In the previous step, you have already created the `/srv/inception/settings.properties` file. You **may optionally** configure the Tomcat port using the following line

  ```
  server.port=18080
  ```

  If you need to do additional configurations of the embedded Tomcat, best refer to the documentation of Spring Boot itself.

- Make sure that the file `/srv/inception/inception.conf` is owned by the root user. If this is not the case, INCEpTION will ignore it and any settings made there will not have any effect. If you start INCEpTION and intead of using the MySQL database, it is using an embedded database, then you should double-check that `/srv/inception/inception.conf` is owned by the root user.

  ```
  $ chown root:root /srv/inception/inception.conf
  ```

- Change the owner/group of `/srv/inception/inception.jar` to `www-data`. When the service is started, it will run with the privileges of the user that owns the JAR file, i.e. in this case INCEpTION will run as under the `www-data` user. **Do NOT run INCEpTION as root.**

  ```
  $ chown www-data:www-data /srv/inception/inception.jar
  ```

- Make the JAR file executable:

```
$ chmod +x /srv/inception/inception.jar
```

- Create a symlink from `/etc/init.d` to the `/srv/inception/inception.jar`:

```
$ ln -s /srv/inception/inception.jar /etc/init.d/inception
```

- Enable the INCEpTION service using

```
$ systemctl enable inception
```

- Start INCEpTION using

```
$ service inception start
```

- Check the log output

```
$ cat /var/log/inception.log
```

- Stop INCEpTION using

```
$ service inception stop
```

Running the standalone behind HTTPD

These are **optional** instructions if you want to run INCEpTION behind an Apache web-server instead of accessing it directly. This assumes that you already have the following packages installed:

- Apache Web Server
- mod_proxy
- mod_proxy_ajp
- Add the following lines to `/srv/inception/settings.properties`:

```
tomcat.ajp.port=18009
server.contextPath=/inception
server.use-forward-headers=true
```

- Edit `/etc/apache2/conf.d/inception.local.conf`

```
ProxyPreserveHost On

<Proxy ajp://localhost/inception >
  Order Deny,Allow
  Deny from none
  Allow from all
</Proxy>

<Location /inception >
  ProxyPass ajp://localhost:18009/inception timeout=1200
  ProxyPassReverse http://localhost/inception
</Location>
```

- Restart Apache web server

```
$ service apache2 restart
```

Running via Docker

Quick start

If you have Docker installed, you can run INCEpTION using

```
$ docker run -it --name inception -p8080:8080 inceptionproject/inception:0.12.2
```

The command downloads INCEpTION from Dockerhub and starts it on port 8080. If this port is not available on your machine, you should provide another port to the `-p` parameter.

The logs will be printed to the console. To stop the container, press `CTRL-C`.

To run the INCEpTION docker in the background use

```
$ docker run -d --name inception -p8080:8080 inceptionproject/inception:0.12.2
```

Logs are accessible by typing

```
$ docker logs inception
```

Use `docker run` only the first time that you run INCEpTION. If you try it a second time, Docker will complain about the name `inception` already being in use. If you follow Docker`s suggestion to delete the container, you will loose all your INCEpTION data. Further below, we explain how you can store your data outside the container in a folder on your host.

When you want to run INCEpTION again later, use the command

```
$ docker start -ai inception
```

or for the background mode

```
$ docker start inception
```

Storing data on the host

If you follow the quick start instructions above, INCEpTION will store all its data inside the docker container. This is normally not what you want because as soon as you delete the container, all data is gone. That means for example that you cannot easily upgrade to a new version of the INCEpTION docker image when one is released.

To store your data on your host computer, first create a folder where you want to store your data. For example, if you are on Linux, you could create a folder `/srv/inception`:

```
$ mkdir /srv/inception
```

When you run INCEpTION via Docker, you then mount this folder into the container:

```
$ docker run -it --name inception -v /srv/inception:/export -p8080:8080
inceptionproject/inception:0.12.2
```

Settings file

The dockerized INCEpTION expects the `settings.properties` file in the `/export` folder. Instead of injecting a custom `settings.properties` file into the container, it is strongly recommender to use the instructions above (Storing data on the host) to mount a folder from the host system to `/export` then to place the into the mounted folder `settings.properties`. Thus, if you follow the instructions above, the settings file would go to `/srv/inception/settings.properties` on the host system.

Connecting to a MySQL database

By default, INCEpTION uses an embedded SQL database to store its metadata (not the texts, annotations and knowledge bases, these are stored in files on disk). For production use, it is highly recommended to use a separate MySQL database instead of the embedded SQL database.

Docker Compose

Using Docker Compose, you can manage multiple related containers. This section illustrates how to use Docker Compose to jointly set up a INCEpTION container as well as a database container (i.e. this one).

The following Compose script sets these containers up.

Docker Compose script

```
##
# docker-compose up [-d]
# docker-compose down
##
version: '2.1'

networks:
  inception-net:

services:
  mysqlserver:
    image: "mysql:5"
```

```
    container_name: inception_mysql
    environment:
      - MYSQL_RANDOM_ROOT_PASSWORD=yes
      - MYSQL_DATABASE=inception
      - MYSQL_USER=${DBUSER}
      - MYSQL_PORT=3306
      - MYSQL_PASSWORD=${DBPASSWORD}
    volumes:
      - ${INCEPTION_HOME}/mysql-data:/var/lib/mysql
    command: ["--character-set-server=utf8", "--collation-server=utf8_bin"]
    healthcheck:
      test: ["CMD", "mysqladmin" ,"ping", "-h", "localhost", "-p${DBPASSWORD}", "-
u${DBUSER}"]
      interval: 20s
      timeout: 10s
      retries: 10
    networks:
      inception-net:

  webserver:
    image: "inceptionproject/inception:0.12.2"
    container_name: inception_webserver
    ports:
      - "${INCEPTION_PORT}:8080"
    environment:
      - INCEPTION_DB_DIALECT=org.hibernate.dialect.MySQL5InnoDBDialect
      - INCEPTION_DB_DRIVER=com.mysql.jdbc.Driver
      -
INCEPTION_DB_URL=jdbc:mysql://mysqlserver:3306/inception?useSSL=false&useUnicode=true&
characterEncoding=UTF-8
      - INCEPTION_DB_USERNAME=${DBUSER}
      - INCEPTION_DB_PASSWORD=${DBPASSWORD}
    volumes:
      - ${INCEPTION_HOME}/server-data:/export
    depends_on:
      mysqlserver:
        condition: service_healthy
    mem_limit: 1g
    memswap_limit: 1g
    restart: unless-stopped
    networks:
      inception-net:
```

Place the script into any folder, change to that folder, and issue the following commands which define the username/password you wish to use for INCEpTION to talk to the database, the folder on the host system where the application data is stored, and the port on which the application will run. The last command starts the containers.

```
$ export DBUSER=<USER_NAME>
$ export DBPASSWORD=<PASSWORD>
$ export INCEPTION_HOME=/srv/inception
$ export INCEPTION_PORT=8080
$ docker-compose -p inception up -d
```

This will start two docker containers: `inception_mysqlserver_1`, and `inception_webserver_1`. You can check the logs of each by running

```
$ docker logs inception_mysqlserver_1
$ docker logs inception_webserver_1
```

The actual name of these containers might vary. A list of running containers can be retrieved by

```
$ docker ps
```

Two directories in your INCEpTION home folder will be created: `mysql-data` and `webserver-data`. No data is stored in the containers themselves, you are safe to delete them with

```
$ docker-compose -p inception down
```

You can also just stop or pause them, please see the docker-compose reference for details.

Upgrading

In general, it is possible to perform an in-place upgrade of the application. However, before doing an upgrade, it is recommender to create a backup of the application and data to allow coming back to a working system if case of a problem during the upgrade. Mind that the upgrade is only completed once the new version has successfully started because during startup, the application may make changes to the database schema or to the data on disk.

Backup your data

- Make a copy of your INCEpTION home folder
- If you are using MySQL, make a backup of your INCEpTION database, e.g. using the mysqldump command.

Upgrading with embedded Tomcat

- Stop the INCEpTION service
- Replace the `inception.jar` file with the new version
- Ensure that the file has the right owner/group (usually `www-data`)
- Start the INCEpTION service again

Remote API

In order to programmatically manage annotation project, a REST-like remote API is offered. This API is disabled by default. In order to enable it, add the setting `remote-api.enabled=true` to the `settings.properties` file.

Once the remote API is enabled, it becomes possible to assign the role `ROLE_REMOTE` to a user. Create a new user, e.g. `remote-api` via the user management page and assign at least the roles `ROLE_USER` and `ROLE_REMOTE`. Most of the actions accessible through the remote API require administrator access, so adding the `ROLE_ADMIN` is usually necessary as well.

Once the remote API has been enabled, it offers a convenient and self-explanatory web-based user interface under `<APPLICATION_URL>/swagger-ui.html` which can be accessed by any user with the role `ROLE_REMOTE`. Here, you can browse the different operations, their parameters, and even try them out directly via a web browser. The actual AERO remote API uses `<APPLICATION_URL/api/aero/v1` as the base URL for its operations.

The API follows the Annotation Editor Remote Operations (AERO) protocol.

Table 4. Remote API settings

Setting	Description	Default	Example
remote-api.enabled	Enable remote API	false	true

Webhooks

Webhooks allow INCEpTION to notify external services about certain events. For example, an external service can be triggered when an annotator marks a document as finished or when all documents in a project have been completely curated.

Webhooks are declared in the `settings.properties` file. For every webhook, it is necessary to specify an URL (`url`) and a set of topics (`topics`) about with the remote service listening at the given URL is notified. If the remote service is accessible via https and the certificate is not known to the JVM running INCEpTION, the certificate verification can be disabled (`verify-certificates`).

The following topics are supported:

- `DOCUMENT_STATE` - events related to the change of a document state such as when any user starts annotating or curating the document.
- `ANNOTATION_STATE` - events related to the change of an annotation state such as when a user starts or completes the annotation of a document.
- `PROJECT_STATE` - events related to the change of an entire project such as when all documents have been curated.

Example webhook configuration

```
webhooks.globalHooks[0].url=http://localhost:3333/
webhooks.globalHooks[0].topics[0]=DOCUMENT_STATE
webhooks.globalHooks[0].topics[1]=ANNOTATION_STATE
webhooks.globalHooks[0].topics[2]=PROJECT_STATE
webhooks.globalHooks[0].verify-certificates=false
```

Settings

Application settings are managed via a file called `settings.properties` which must reside in the application home folder. The file is optional. If it does not exist, default values are assumed.

General Settings

Table 5. General settings

Setting	Description	Default	Example
warnings.unsupportedBrowser	Warn about unsupported browser	true	false
debug.showExceptionPage	Show a page with a stack trace instead of an "Internal error" page. Do not use in production!	false	true
login.message	Custom message to appear on the login page, such as project web-site, annotation guideline link, ... The message can be an HTML content.	*unset*	`<span style="color:red; font-size: 200%;">Use are your own risk.</span>`
user.profile.accessible	Whether regular users can access their own profile to change their password and other profile information. This setting has no effect when running in pre-authentication mode.	false	true
user-selection.hideUsers	Whether the list of users show in the users tab of the project settings is restricted. If this setting is enable, the full name of a user has to be entered into the input field before the user can be added. If this setting is disabled, it is possible to see all enabled users and to add any of them to the project.	false	true

Database connection

Table 6. Database settings in the `settings.properties` file

Setting	Description	Default	Example
database.dialect	Database dialect	org.hibernate.dialect.HSQLDialect	org.hibernate.dialect.MySQL5InnoDBDialect
database.driver	Database driver	org.hsqldb.jdbc.JDBCDriver	com.mysql.jdbc.Driver
database.url	JDBC connection string	*location in application home*	jdbc:mysql://localhost:3306/weblab?useUnicode=true&characterEncoding=UTF-8&serverTimezone=UTC
database.username	Database username	sa	user
database.password	Database password	*unset*	pass
database.initial-pool-size	Initial database connection pool size	4	
database.min-pool-size	Minimum database connection pool size	4	
database.max-pool-size	Maximum database connection pool size	10	
warnings.embeddedDatabase	Warn about using an embedded database	true	false

The basic database connection details can also be configured via environment variables. When these environment variables are present, they are preferred over the `settings.properties` file. The following environment variables can be used:

Table 7. Database configuration via environment variables

Setting	Description	Default	Example
`INCEPTION_DB_DIALECT`	Database dialect	org.hibernate.dialect.HSQLDialect	org.hibernate.dialect.MySQL5InnoDBDialect
`INCEPTION_DB_DRIVER`	Database driver	org.hsqldb.jdbc.JDBCDriver	com.mysql.jdbc.Driver
`INCEPTION_DB_URL`	JDBC connection string	*location in application home*	jdbc:mysql://localhost:3306/inception?useUnicode=true&characterEncoding=UTF-8
`INCEPTION_DB_USERNAME`	Database username	sa	user
`INCEPTION_DB_PASSWORD`	Database password	*unset*	pass

Internal backup

INCEpTION stores its annotations internally in files. Whenever a user performs an action on a document, the file is updated. It is possible to configure INCEpTION to keep internal backups of these files, e.g. to safeguard against crashes or bugs.

The internal backups are controlled through three properties:

Table 8. Database settings in the `settings.properties` file

Setting	Description	Default	Example
backup.interval	Time between backups (seconds)	0 *(disabled)*	300 *(60 * 5 = 5 minutes)*
backup.keep.number	Maximum number of backups to keep	0 *(unlimited)*	5
backup.keep.time	Maximum age of backups to keep (seconds)	0 *(unlimited)*	2592000 *(60 * 60 * 24 * 30 = 30 days)*

By default, backups are disabled (**backup.interval** is set to `0`). Changing this properties to any positive number enables internal backups. The interval controls the minimum time between changes to a document that needs to have elapsed in order for a new backup to be created.

When backups are enabled, either or both of the properties **backup.keep.number** and **backup.keep.time** should be changed as well, because their default values will cause the backups to be stored indefinitely and they will eventually fill up the disk.

The properties **backup.keep.number** and **backup.keep.time** control how long backups are keep and the maximal number of backups to keep. These settings are effective simultaneously.

Example: Make backups every 5 minutes and keep 10 backups irrespective of age

```
backup.interval    = 300
backup.keep.number = 10
backup.keep.time   = 0
```

*Example: Make backups every 5 minutes and all not older than 7 days (60 * 60 * 24 * 7 seconds)*

```
backup.interval    = 300
backup.keep.number = 0
backup.keep.time   = 604800
```

Example: Make backups every 5 minutes and keep at most 10 backups that are not older than 7 days

```
backup.interval    = 300
backup.keep.number = 10
backup.keep.time   = 604800
```

Custom header icons

INCEpTION allows adding custom icons to the page header. You can declare such custom icons in the `settings.properties` file as shown in the example below. Each declaration begins with the prefix `style.header.icon.` followed by an identifier (here `myOrganization` and `mySupport`). The suffixes `.linkUrl` and `.imageUrl` indicate the URL of the target page and of the icon image respectively. Images are automatically resized via CSS. However, to keep loading times low, you should point to a reasonably small image.

The order of the icons is controlled by the ID, not by the order in the configuration file!

Example: Custom header icon

```
style.header.icon.myOrganization.linkUrl=http://my.org
style.header.icon.myOrganization.imageUrl=http://my.org/logo.png
style.header.icon.mySupport.linkUrl=http://my.org/support
style.header.icon.mySupport.imageUrl=http://my.org/help.png
```

Setting	Description	Default	Example
style.logo	Logo image displayed in the upper-right corner	*unset*	*path to an image file*
style.header.icon…	Icons/links to display in the page header. For details, see below.	*unset*	

Annotation editor

Setting	Description	Default	Example
ui.brat.autoScroll	Whether to scroll the annotation being edited into the center of the page	true	
ui.brat.pageSize	The number of sentences to display per page	5	
ui.brat.singleClickSelection	Whether to select annotations with a single click	false	
ui.brat.rememberLayer	Whether "remember layer" is activated by default	false	
annotation.feature-support.string.autoCompleteThreshold	If the tagset is larger than the threshold, an auto-complete field is used instead of a standard combobox.	75	100
annotation.feature-support.string.autoCompleteMaxResults	When an auto-complete field is used, this determines the maximum number of items shown in the dropdown menu.	100	1000

External pre-authentication

INCEpTION can be used in conjunction with header-based external per-authentication. In this mode, the application looks for a special HTTP header (by default `remote_user`) and if that header exists, it is taken for granted that this user has been authenticated. The application will check its internal database if a user by the given name exists, otherwise it will create the user.

Pre-authentication can be enabled by setting the property `auth.mode` to `preauth`. When enabling pre-authentication mode, the default roles for new users can be controlled using the `auth.preauth.newuser.roles` property. The `ROLE_USER` is always added, even if not specified explicitly. Adding also the role `ROLE_PROEJCT_CREATOR` allows all auto-created users also to create their own projects.

Since the default administrator user is not created in pre-authentication, it is useful to also declare at least one user as an administrator. This is done through the property `auth.user.<username>.roles` where `<username>` must be replaced with the name of the user. The example below shows how the user **Franz** is given administrator permissions.

Example: Authenticate using the `remote_user` header, new users can create projects, user **Franz** *is always admin.*

```
auth.mode                     = preauth
auth.preauth.header.principal = remote_user
auth.preauth.newuser.roles    = ROLE_PROJECT_CREATOR
auth.user.Franz.roles         = ROLE_ADMIN
```

The roles specified through `auth.preauth.newuser.roles` are saved in the database when a user logs in for the first time and can be changed after creation through the user interface.

The roles added through `auth.user.<username>.roles` properties are **not** saved in the database and **cannot** be edited through the user interface.

Setting	Description	Default	Example
auth.mode	Authentication mode	database	preauth
auth.preauth.header.principal	Principal header	remote_user	*some other header*
auth.preauth.newuser.roles	Default roles for new users (comma separated)	*<none>*	ROLE_PROJECT_CREATOR
auth.user.*<username>*.roles	Extra roles for user (comma separated)	*<none>*	ROLE_ADMIN

Concept Linking

There are several configurable parameters related to the Concept Linking functionality:

Cache size

This parameter controls the size of the Candidate Cache, which stores a set of candidates for a mention. Increasing the cache size will reduce the number of queries that have to be made against the KB and therefore increase average retrieval time.

Candidate Frequency Threshold

This parameter controls after how many concepts the ranking approach should take into account by selecting the `n` most frequent concepts. Increasing this parameter will lead to a longer ranking time, since more candidates are considered for ranking.

Mention Context Size

This parameter declares the size `k` of the context, where the context is defined as the words included in a window with `k` words to both left and right.

Candidate Retrieval Limit

This parameter defines how many concepts should be retrieved for the Candidate Retrieval step. Increasing this parameter will lead to a longer time to retrieve candidates from the KB.

Semantic Signature Query Limit

This parameter defines how many concepts should be retrieved for the Semantic Signature of a candidate. Increasing this parameter will lead to a longer time to retrieve concepts for constructing the Semantic Signature.

Candidate Display Limit

This parameter regulates how many candidates will be displayed for a mention in the Concept Selector UI.

If no value for a parameter is specified, its default value is used. The default values are shown as examples of how the parameters can be configured below:

Table 9. Concept linking settings overview

Setting	Description	Default	Example
inception.entity-linking.cacheSize	Cache size	1024	-
inception.entity-linking.candidateQueryLimit	Candidate Retrieval Limit	2500	-
inception.entity-linking.mentionContextSize	Mention Context Size	5	-
inception.entity-linking.candidateDisplayLimit	Candidate Display Limit	100	-

Setting	Description	Default	Example
inception.entity-linking.signatureQueryLimit	Semantic Signature Query Limit	2147483647	-

Resources

In order to improve the quality of suggestions, several additional resources can be incorporated. These are to be put into the `.inception/resources` folder. These include:

- properties_with_labels.txt
  - List of properties, each line containing information for one property, tab-separated

ID	Label	Description	Aliases	Data type	Count
P6	head of government	head of the executive power of this town, city, municipality, state, country, or other governmental body	government headed by, executive power headed by, president, chancellor	wikibase-item	17,592

- property_blacklist.txt
  - A list of properties that are filtered when computing the Semantic Signature, one property ID per line, e.g. `P1005`, `P1014`
- stopwords-en.txt
  - A list of stopwords, one stopword per line, e.g. `i`, `me`
- wikidata_entity_freqs.map
  - Each line consists of a the ID of a concept and its frequency in the KB, tab-separated, e.g. `Q4664130 409104, Q30 205747`

Knowledge Base Settings

This section describes the global settings related to the knowledge base module.

Default max results

This parameter determines the default value for the maximum number of results that can be retrieved from a SPARQL query. The queries are used to retrieve concepts, statements, properties, etc. from the knowledge base. The maximum number of results can also be configured separately for each knowledge base in the project settings.

Hard max results

A hard limit for the **Max results** parameter.

If no value for the parameter is specified, its default value is used. The default value is shown as an example of how the parameter can be configured below:

Table 10. Knowledge base settings overview

Setting	Description	Default	Example
inception.knowledge-base.defaultMaxResults	default result limit for SPARQL query	1000	10000
inception.knowledge-base.hardMaxResults	hard limit for the maximum number of results from a query	10000	5000

Scheduler Settings

This section describes the global settings related to the scheduler.

Number of threads

This parameter determines the number of threads the scheduler uses. It should be less than hardware threads available on the machine that runs INCEpTION. The higher the number, the more tasks can be run in parallel.

Queue size

This parameter determines the maximum number of tasks that can be waiting in the scheduler queue. If the queue is full, then no new tasks can be scheduled until running tasks are completed.

If no value for the parameter is specified, its default value is used. The default value is shown as an example of how the parameter can be configured below:

Table 11. Scheduler settings overview

Setting	Description	Default	Example
inception.scheduler.numberOfThreads	Number of threads that run tasks	4	8
inception.scheduler.queueSize	Maximum number of tasks waiting for execution	100	200