Installation Guide

1 Installation and Setup of the Application Server

The Application was developed using the open source server Apache Tomcat (version 8.0.32.0). To get a copy of the software and information on how to install and configure it refer to http://tomcat.apache.org/.

2 Setup and Configuration of the Runtime Environment

2.1 Directory Structure

```
$CATHALINA_BASE/
|--- bin/
|--- conf/
|--- lib/
|--- logs/
|--- temp/
|--- webapps/
     |--- application/
          |--- META-INF/
          |--- VAADIN/
               |--- addons.css
               |--- styles.css
               |--- application.css
          |--- WEB-INF/
               |--- classes/
               |--- lib/
               |--- web.xml
|--- work
|--- config.properties
|--- shiro.ini
```

The directory structure of your runtime environment should be similar to the one outlined above. `$CATHALINA_BASE` references to the root reference of your Apache Tomcat installation. The application will reside in its own folder (`application`) inside of `webapps`. The stylesheets of the application can be found inside the `VAADIN` directory. The compiled application classes and

dependencies are located in the `classes` and `lib` folders respectively. The `shiro.ini` file is used to configure user authentication and authorization. The `web.xml` file is the deployment descriptor of the application. For more information on the deployment descriptor refer to the Java Servlet Specification.
Instead of holding a copy of the directory structure inside your application server you can pack the `application` folder in a web application archive (WAR) file and put a copy of it inside the `webapps` folder.

2.2 User Management, Authentication and Authorization

The application uses Apache Shiro for user management, authentication, authorization and session management. For detailed information about this framework refer to http://shiro.apache.org/.
Apache Shiro uses a text based configuration for web applications. The `shiro.ini` file has the following structure:

```
[main]
filter.property = value
[users]
username = password, role
[roles]
role = privilege [, ...]
[urls]
/path = filter [, ...]
```

To add a user create a new line under the `[users]` section according to the format shown above. You can assign a role, i.e. a set of privileges to a user. If you don’t assign a role to the user he/she will have no special privileges.
The roles are defined under the `[roles]` section. The application has two predefined roles: admin and user. The admin role holds the privileges write:add, write:edit and report. The user role holds the privileges write:add and report. The write:add privilege allows to add new entries or property-value pairs to the database or edit property-value pairs with unassigned values (cf. User’s Guide). The write:edit privilege allows to edit property-value pairs with assigned values of the database. The report privilege allows a user to report errors and make suggestions for a correction. You can add new role by adding a new line under `[roles]` section according to the format shown above.
The access restriction and login/logout behavior is defined under the `[urls]` section using filters. The filters are configured under the `[main]` section. For detailed information on filters refer to http://shiro.apache.org/. The default configuration is as follows:

```
[main]
authc.loginUrl = /login.jsp
logout.redirectUrl = /
...
[urls]
/admin = authc, roles[admin]
/logout = logout
```

The `/admin = authc, roles[admin]` line says that access to the admin page (cf. User's Manual) requires authentication and is restricted to users with the role admin. The line `authc.loginUrl = /login.jsp` says that an unauthenticated user trying to access the admin page will be redirected to the specified login page.
The `/logout = logout` and `logout.redirectUrl = /` line specify that a user shall be redirected to the root page on logout.
You can extend the authentication and authorization behavior by adding new urls and configuring filters according to the formats specified above.

The text based configuration does not allow dynamic configuration at runtime. To apply any changes you have to restart the server.

2.3 Configuring the Application

The application is configured via the `config.properties` file (using the Java Properties File Format). You can set three parameters:

`TDB_PATH`	specifies the full path to your triple store location on filesystem
`TTL_PATH`	specifies the full path to your Turtle files on filesystem
`MESSAGES_PATH`	specifies the full path to admin messages directory on filesystem
`BACKUP_PATH`	specifies the full path to the backup directory on filesystem
`BACKUP_INTEVAL`	specifies the time interval for backup .ttl export in miliseconds

The text based configuration does not allow dynamic configuration at runtime. To apply any changes you have to restart the server.

2.4 Initializing the Database

On server startup the application will check if the configured `TDB_PATH` is empty. If so, the Database will be initialized using the Turtle files at location specified by `TTL_PATH`. If the TDB directory is not empty initialization will be skipped. To reinitialize the database just empty the directory.