CONTENTS

1 INTRODUCTION
   1.1 Purpose of the DDMoRe Model Repository
   1.2 Prerequisites and supported platforms
      1.2.1 Hardware requirements
      1.2.2 Software requirements
   1.3 Support

2 INSTALLATION OF SOFTWARE REQUIREMENTS
   2.1 Core prerequisites for model repository
      2.1.1 Installing Oracle Java 7
         2.1.1.1 Installation of Java 7 on Linux
         2.1.1.2 Installation of Java 7 on Windows
      2.1.2 Checking MySQL installation
      2.1.3 Installing Apache Solr Server
      2.1.4 Creating model repository database
      2.1.5 Mail server installation
      2.1.6 Installation directory structure
      2.1.7 Configuration settings
   2.2 Prerequisites for installation model repository from source code
      2.2.1 Installing Apache Maven
      2.2.2 Installing Grails
      2.2.3 Checking Git
   2.3 Prerequisites for installation model repository from pre-built binaries
   2.4 Prerequisites for deploying model repository
      2.4.1 Setting up the application server
      2.4.2 Running Solr Server

3 INSTALLATION FROM SOURCE CODE
   3.1 Building search indexer
      3.1.1 Clone
      3.1.2 Build
   3.2 Building Model Repository
      3.2.1 Clone the repository
      3.2.4 Database schema and initial content
      3.2.5 Generating the binary archive

4 INSTALLATION FROM PRE-BUILT BINARIES
   4.1 Download pre-built binaries
   4.2 Initialise content for model repository database

5 DEPLOYMENT

6 ADMINISTRATION OF DDMoRe MODEL REPOSITORY
   6.1 Using the administrator account
   6.2 Adding a new user
   6.3 Customisation of access control
      6.3.1 Overview of available roles
1 INTRODUCTION

1.1 Purpose of the DDMoRe Model Repository

The Drug Disease Model Resources project (DDMoRe) is a Europe-wide effort funded by Innovative Medicines Initiative Joint Undertaking that seeks to create an environment governed by standards that facilitates data and knowledge-sharing between pharmaceutical companies, regulatory bodies and academia.

Specifically, the DDMoRe consortium is developing a common definition language for data, models and workflows, as well as a standard for storing and exchanging models, data and associated metadata.

In this context, the Model Repository is a platform that aids collaborative model development by providing a central, versioned storage infrastructure where public and private models and relevant data can be deposited. The Model Repository also allows researchers to perform exploratory model analysis by offering means to browse, search and view the content it stores. The application captures structured information about the models' context, which is then used to power model search. Finally, the models can be downloaded by users either directly, or programmatically.

One key deliverable of DDMoRe is the deployment of a publicly-available instance of the Model Repository, allowing access to pre-competitive models and data. However, this application has been designed with flexibility and modularity in mind, allowing private instances to be installed in corporate environments with a view to provide an appealing alternative to existing proprietary solutions for storing models and data.

For this reason, the project has been designed to be just a model management platform (JUMMP) that allows extensive customisation. This enables it to be tailored to each organisation's requirements and software environment.

1.2 Prerequisites and supported platforms

1.2.1 Hardware requirements

A 64-bit environment is strongly advised, mostly due to the RAM requirements of the application – minimum 8GB, with a recommended amount of 16GB. Take into consideration the total amount of RAM available on the machine when allocating memory to the application, as a value that is too high will result in increased swapping, which will severely impact the server's performance. The amount of disk space that is required by the application depends on the quantity and complexity of the models and data it handles. A minimum of 1GB is required, although you are advised to allow
10GB or more, as lack of disk space will prevent new submissions to the Model Repository, or even its optimal functioning.

1.2.2 Software requirements

Model Repository’s architecture is organised in three separate projects: the main application (Jummp), an Apache Solr-backed component in charge of maintaining the search index (Jummp indexer) and a collection of externalised domain classes shared by the two applications. As the last two projects are Maven-based, an Apache Maven installation is necessary on the system in order to build the application from source code.

The DDMoRe Model Repository has been designed to work on both Linux distributions (kernel version 2.6+) and Windows (XP or newer). The application requires JavaSE version 7, a relational database management system, an application server or a servlet container and Apache Solr.

The application can be customised to work with either MySQL or PostgreSQL.

The Model Repository has been successfully deployed on Tomcat 7 and Glassfish 3, but other servlet containers or application servers may also work. The project has been tested with the Oracle implementation of the JDK\textsuperscript{1}. Unfortunately, due to compatibility issues, OpenJDK is known not to work.

To serve model searching, Solr should be installed and launched before starting the Model Repository. The current code base is tested against Solr version 5.4.0 compatible with Java 7. The downloadable binary Solr version is available at http://archive.apache.org/dist/lucene/solr/5.4.0/.

If it is intended to build the application from source, then Git (version 1.8 or above), Apache Maven 3, and version 2.3.11 of the web development framework Grails are also necessary. The latter is available from http://dist.springframework.org.s3.amazonaws.com/release/GRAILS/grails-2.3.11.zip. The process of compiling the DDMoRe Model Repository need not take place on the server.

The instructions herein describe making manual modifications to environment variables and also starting and stopping processes by hand, as opposed to through an initialisation service like init.d or systemd. Although we warmly encourage the latter approach to managing an installation of the DDMoRe Repository, we have not provided example scripts due to the sheer diversity of the init implementation landscape. What is more, once a system administrator has understood how to manage the installation, it would be very easy to create the appropriate scripts such that they suit the administrator’s needs and preferences.

The availability of a mail server (such as sendmail http://www.sendmail.com/sm/open_source/) capable of sending emails e.g. through SMTP or STARTTLS is also a prerequisite for the successful installation of an instance of the Model Repository.

Offline mode

There are two additional requirements if the Repository is to reside in an environment that does not have internet access.

The Repository features a contextual help, meaning that the help button on the interface takes users to the section of the user guide that is most appropriate for the page the user is on. The user guide itself consists of a collection of HTML pages and it is available for download as an archive. The archive will need to be unpacked and its contents must be served by a dedicated web server, such as Apache.

\textsuperscript{1} The rest of this guide uses Java 7 to refer to Oracle Java Development Kit (JDK) version 1.7
The Repository is capable of capturing annotations about models, which are used to power the search. The annotation editor is underpinned by an RDF Store which describes what annotations should be captured by the Repository. There exists an official DDMoRe RDF Store, but in the case of internal installations of the Repository, it will be necessary to have an internal RDF Store installed and made available to the Repository. Instructions on how to install the RDF Store are outside the scope of this document.

Finally, the Repository uses the RESTful API of Europe PubMed Central (http://europepmc.org/) to automatically extract information about publications, including authorship and abstract. In offline mode, the automatic retrieval will not work, meaning that users will have to enter all publication details manually.

1.3 Support

If you encounter any problems, please first consult the project’s bug tracker: https://bitbucket.org/jummp/jummp/issues to see whether the issue has already been reported.

If not, please fill in a bug report. To facilitate the resolution of the problem, please include information about your environment, the steps to reproduce the issue and the relevant part of the logs. You can also email us at biomodels-net-support@lists.sourceforge.net.

2 INSTALLATION OF SOFTWARE REQUIREMENTS

In this section and throughout the remainder of this document we will use MySQL and Apache Tomcat to showcase the installation procedure. Please consult your vendor’s documentation if you wish to use a different relational database server or servlet container/application server.

2.1 Core prerequisites for model repository

2.1.1 Installing Oracle Java 7

Verify the Java installation by running the following command in a terminal:

java -version

The output should look like so

java version "1.7.0_79"
Java(TM) SE Runtime Environment (build 1.7.0_79-b15)
Java HotSpot(TM) 64-Bit Server VM (build 24.79-b02, mixed mode)

Be sure that the JAVA_HOME environment variable is also set and pointing to the Java installation folder.

Linux

echo $JAVA_HOME

The expected output looks like to

/homes/tung/DevTools/jdk1.7.0_79

Windows

echo %JAVA_HOME%

The result often shows as
C:\Program Files\Java\jdk1.7.0_79

If the commands above don’t produce any output, this may mean that you have not installed Java yet.

2.1.1.1 Installation of Java 7 on Linux

This procedure installs Java 7, using an archive binary file (.tar.gz)

Step 1: Download the file from
jdk-7<version>-linux-i586.tar.gz is the JDK for 32-bit systems. The file jdk-7<version>-linux-x64.tar.gz is the JDK for 64-bit systems.

Step 2: Change directory to the location where you would like Java 7 to be installed. Move the .tar.gz archive binary file to the current directory.

Step 3: Unpack the tarball and install Java.

```
tar zxfv jdk-7u<version>-linux-i586.tar.gz
```

Java 7 is installed in a directory called jdk1.7.0_<version> in the current directory.

Step 4: Create JAVA_HOME environment variable that points to the directory where you have unpacked the JDK files.

Step 5: Add the bin folder within Java installation location to the PATH environment variable.

Step 6: Test the installation procedure using the instructions presented at the beginning of this section.

We suggest you consult
http://docs.oracle.com/javase/7/docs/webnotes/install/linux/linux-jdk.html to get more information of the JDK installation on Linux.

2.1.1.2 Installation of Java 7 on Windows

This procedure installs Java 7, using a self-installing executable file to unpack and install the JDK (.exe)

Step 1: Download the Installer from

The file jdk-7<version>-windows-i586-i.exe is the JDK installer for 32-bit systems. The file jdk-7<version>-windows-x64.exe is the JDK installer for 64-bit systems.

Step 2: Running the JDK installer.

You must have administrative permissions in order to install the JDK on Microsoft Windows. You should follow the installer instructions to complete the procedure of Java installation.

Step 3: Check the directory where the JDK installer unpacks files to C:\Program Files\Java\jdk1.7.0_<version>

Step 4: Create JAVA_HOME environment variable that points to the directory where you have seen the JDK files.

Step 5: Add the bin folder within Java installation location to the PATH environment variable.

Step 6: Test the installation procedure using the instructions presented at the beginning of this section.

We suggest you consult
http://docs.oracle.com/javase/7/docs/webnotes/install/windows/jdk-installation-windows.html to get more information of the JDK installation on Windows.

2.1.2 Checking MySQL installation

The code base has been successfully ran with MySQL Server 5.1, 5.5 and 5.6. Other versions of the database server have not been tested, but may also work.

To ensure that MySQL is installed, execute the following command in a terminal

```bash
mysql -u <db_username> -p -P <db_port> -h <db_host>
```

Password: ****** (typing the password of the username)

If MySQL is running and you have provided the correct credentials, you should see the output similar to the one below

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 74669436
Server version: 5.6.24-log MySQL Community Server (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

mysql>

Alternatively, you may need to install MySQL. This can be performed by manually retrieving and compiling the sources from http://www.mysql.com/downloads/ or, in the case of Linux distributions, from your package manager.

2.1.3 Installing Apache Solr Server

The search capacities of the DDMoRe Model Repository are powered by the Apache Solr search server. The Repository is currently tested against Solr version 5.4.0, although it should also be compatible with Solr 4. The Repository expects to have an instance of the search server configured and running in order for the former to start.

The Repository does not require the Solr installation to be on the same server, but it is strongly recommended to be on the same network due to the fact that the communication happens through unencrypted HTTP connections.

Check that Solr has been installed on the machine by running the command

```
$ solr start
```

If Solr has already been installed, the expected result should be

```
Waiting up to 30 seconds to see Solr running on port 8983 [/]
Started Solr server on port 8983 (pid=21121). Happy searching!
```

Otherwise, the Solr installation process consists of the following steps:
Step 1: Download Solr from http://archive.apache.org/dist/lucene/solr/5.4.0/ and unpack the contents into your favourite directory.

Step 2: Create a SOLR_HOME environment variable that points to the path where you have just extracted the archive. If you extract Solr to /opt/solr, SOLR_HOME will be /opt/solr/server/solr.

Step 3: Append a reference to the bin directory within the Solr installation directory.

Step 4: Verify the installation by running Solr (see Section 5.1 of this document)

2.1.4 Creating model repository database

You can use any program to connect to MySQL Server like phpmyadmin, mysql workbench or even MySQL command-line client to create the database for your installation. The following command is a simple way of creating a MySQL database:

create database database_name;

The database_name has to match with the one you have defined in .jummp.properties file described in the configuration settings section.

By default, newly-created database can be done all of the editing as the root user. However, in the case of security and unnecessary circumstances, you will be recommended not using the root user frequently. Instead you should create a specific database user and grant it privileges on all operations to repository database schema.

Create a user for repository database

CREATE USER 'jummpdbadmin'@'localhost' IDENTIFIED BY 's3cr3t';

Allow the user permissions on the database

GRANT ALL PRIVILEGES ON database_name.* TO 'jummpdbadmin'@'localhost';

FLUSH PRIVILEGES;

2.1.5 Mail server installation

The DDMoRe Model Repository has been tested with a variety of email providers, thus meaning that it is not necessary to install and maintain a dedicated email server for the Repository. Should you wish, however, to install one, please refer to your vendor's documentation.

2.1.6 Installation directory structure

<table>
<thead>
<tr>
<th>Location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/opt/ddmore-repository</td>
<td>Service home directory.</td>
</tr>
<tr>
<td>/home/username/</td>
<td>Service configuration directory.</td>
</tr>
<tr>
<td></td>
<td>(e.g. the settings file .jummp.properties file would be located here)</td>
</tr>
<tr>
<td>/opt/ddmore-</td>
<td>Service exchange and working files.</td>
</tr>
</tbody>
</table>
There are two folders which are of significance to the Model Repository:

- a working directory containing models and data under version control;
- an exchange directory storing e.g. models retrieved from version control;

It is crucial that these folders exist before the application is started and are available to the user running the servlet container or application server process. In particular, no other account should have write access to these locations.

The application is configured through a property file called .jummp.properties that, by default, should reside in the home folder of the user running the Model Repository. The location of the properties file can be customised through the JUMMP_CONFIG environment variable that should be set to point to the full path of the configuration file -- e.g. /var/ddmore/repo.properties. The important thing to note is that the file should be readable and writable by the Model Repository. For security reasons, the file should not be executable, and it is also advisable not to allow any other user or group to access these settings.

A scaffolded configuration file contains the following properties:

```properties
jummp.security.registration.email.subject=[My Model Repository] new account activation
jummp.security.resetPassword.email.body=Dear {{REALNAME}}, \r\n\r\nA password reset request was received for your account. Please reset your password by going to {{URL}}.\r\n\r\nMy Model Repository
jummp.security.resetPassword.email.subject=[My Model Repository] password reset request

## Database settings
jummp.database.type=MYSQL
# use the name you chose when you created a database in Section 3.2.2
jummp.database.database=<database_name>:
jummp.database.password=<database_user_password>
jummp.database.port=<database_port>
jummp.database.server=<database_server_name>
jummp.database.username=<database_username>

## Model identifier settings
jummp.model.id.submission.part1.type=literal

## User preferences
jummp.model.history.maxElements = 10
```

## Example Configuration file ##

## Jummp Configuration

## Security settings
jummp.security.anonymousRegistration=true
jummp.security.authenticationBackend=database
jummp.context.help.notifications=notifications.html

## User preferences
jummp.model.history.maxElements = 10
## Metadata management settings
jummp.security.cms.policy=/path/to/jummp/source/scripts/WeceemSecurity.groovy

## Email settings
jummp.security.mailer.auth=true
jummp.security.mailer.host=<host_address>
jummp.security.mailer.password=<your_password>
jummp.security.mailer.port=<port_number>
jummp.security.mailer.tlsrequired=true
jummp.security.mailer.username=<your_user_name>
jummp.security.curatorByDefault=false
jummp.security.registration.email.adminAddress=<your_email_address>
jummp.security.registration.email.body=Dear {{REALNAME}},

An account to access the Model Repository has been created for you. Your username is:

{{USERNAME}}.

Your automatically generated password is: {{PASSWORD}}
(you can easily change it after your first login).

Good luck!

jummp.security.registration.email.send=true
jummp.security.registration.email.sendToAdmin=false
jummp.security.registration.email.sender=<sender’s_email_address>
p.model.id.submission.part1.suffix=MODEL
jummp.model.id.submission.part2.type=numerical
jummp.model.id.submission.part2.fixed=false
jummp.model.id.submission.part2.width=8

## Solr settings
# For solr-4
#jummp.search.folder=/path/solr/installed/example/solr
# For solr-5
jummp.search.folder=/path/solr/installed/server/solr
jummp.search.pathToIndexerExecutable=/path/to/JummpSolrIndexer.jar

## Server settings
jummp.server.protection=false
# The address from which users will be able to access this instance
jummp.server.url=http://server_name:port_number/model-repository

## Version Control System (VCS) settings
jummp.vcs.plugin=git
jummp.vcs.exchangeDirectory=/path/to/directory/exchange
jummp.vcs.workingDirectory=/path/to/directory/working

## Embedded user guide
# root URL for help
jummp.context.help.root=http://www.ebi.ac.uk/biomodels/ddmore-repo-docs/
# all other pages
jummp.context.help.browse=model_browsing.html
jummp.context.help.search=model_search.html
jummp.context.help.login=login.html
jummp.context.help.display=model_display.html
jummp.context.help.archives=model_archives.html
jummp.context.help.submission=model_submission.html
jummp.context.help.update=model_update.html
jummp.context.help.profile=user_profiles.html
jummp.context.help.sharing=model_sharing.html
jummp.context.help.teams=teams.html
The `.jummp.properties` file basically comprises of the sections of settings: Security, Database, Model, Solr, Server, Version Control System, Context Specific Help Info, and Others. These sections are presented in details in Table 3.1.

### Table 3.1. Jummp configuration

<table>
<thead>
<tr>
<th>Security settings</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>jummp.security.anonymousRegistration</strong></td>
</tr>
<tr>
<td>Enable/disable users to register for a new account. This property accepts a boolean value, true or false.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
</tr>
<tr>
<td>jummp.security.anonymousRegistration=true</td>
</tr>
</tbody>
</table>

| **jummp.security.authenticationBackend**               |
| Indicate the authentication mechanism that should be used by the application. The only supported value at the moment is 'database', but this setting makes provision for developing alternative authentication mechanisms e.g. LDAP in the future. |
| **Example:**                                          |
| jummp.security.authenticationBackend=database         |

| **jummp.security.cms.policy**                         |
| Indicate the path where the application finds WeceemSecurity.groovy -- the security policy for editing the content of some static pages without having to reload the application for the changes to take effect. The application’s source code comes bundled with a preferred policy that is available at [https://bitbucket.org/jummp/jummp/src/b564d1cbee78b845caea9fb56f52d4e8d0a56f78d/scripts/WeceemSecurity.groovy?at=master&fileviewer=file-view-default](https://bitbucket.org/jummp/jummp/src/b564d1cbee78b845caea9fb56f52d4e8d0a56f78d/scripts/WeceemSecurity.groovy?at=master&fileviewer=file-view-default) |
| **Example:**                                          |
| jummp.security.cms.policy=/home/tung/model-repository/WeceemSecurity.groovy |

| **jummp.security.mailer.auth**                        |
| Enable/disable authentication for the mailer functionality. This property accepts a boolean value, true or false. Set to true if the mail server requires authentication, in which case also set jummp.security.mailer.username and jummp.security.mailer.username to the credentials that should be used to connect to the SMTP server. |
| **Example:**                                          |
| jummp.security.mailer.auth=true                       |

<p>| <strong>jummp.security.mailer.host</strong>                        |
| Indicate the host of the SMTP mail server for outgoing messages. |
| <strong>Example:</strong>                                          |
|                                                     |</p>
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>jummp.security.mailer.host</td>
<td>Indicates the SMTP host.</td>
<td>smtp.domain.com</td>
</tr>
<tr>
<td>jummp.security.mailer.port</td>
<td>Indicates the port number of the outgoing mail (SMTP) server being used.</td>
<td>587</td>
</tr>
<tr>
<td>jummp.security.mailer.tlsrequired</td>
<td>Enable/disable usage of STARTTLS. Enable this setting if you chose port 587</td>
<td>true or false</td>
</tr>
<tr>
<td>jummp.security.mailer.username</td>
<td>Indicates the username that should be used for SMTP authentication.</td>
<td><a href="mailto:model-repository@domain.com">model-repository@domain.com</a></td>
</tr>
<tr>
<td>jummp.security.mailer.password</td>
<td>Indicates the password that should be used in order to authenticate with the</td>
<td>s3cr3t</td>
</tr>
<tr>
<td>jummp.security.curatorByDefault</td>
<td>By default, newly-registered users will be allowed to publish models.</td>
<td>false</td>
</tr>
<tr>
<td>jummp.security.certificationRole</td>
<td>Jummp allows models to be marked as certified -- denoting a particular</td>
<td>ROLE_ADMIN,ROLE_CURATOR</td>
</tr>
<tr>
<td>jummp.security.registration.email.adminAddress</td>
<td>Indicates the administrator's email address. Will receive maintenance- and</td>
<td><a href="mailto:modeladmin@domain.com">modeladmin@domain.com</a></td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>jummp.security.registration.email.body</td>
<td>The message template sent to a new user after successful registration. The pre-defined variables, i.e. <code>{{REALNAME}}</code>, <code>{{USERNAME}}</code>, <code>{{PASSWORD}}</code>, will be replaced by their actual values in the email sent to users.</td>
<td></td>
</tr>
</tbody>
</table>
| Example:                                     | `jummp.security.registration.email.body=Dear {{REALNAME}},
\r\n\r\nAn account to access the Model Repository has been created for you. Your username is: {{USERNAME}}.\r\n\r\nYour automatically-generated password is: {{PASSWORD}} (you can easily change it after your first login).\r\n\r\nGood luck!`                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| jummp.security.registration.email.send       | Allow/disallow sending a confirmation email for a new registration. This property accepts a boolean value, true or false. It is recommended to set this to true.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Example:                                     | `jummp.security.registration.email.send=true`                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| jummp.security.registration.email.sendToAdmin| Enable/disble the sending feature to administrator when a new registration is proceeded successfully. This property accepts a boolean value, true or false.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Example:                                     | `jummp.security.registration.email.sendToAdmin=false`                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| jummp.security.registration.email.sender     | Indicate the sender's address when sending users a confirmation about their account being created.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Example:                                     | `jummp.security.registration.email.sender=admin@domain.com`                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| jummp.security.registration.email.subject    | The subject of the message that will be sent to the new user following successful registration.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Example:                                     | `jummp.security.registration.email.subject=[My Model Repository] new account activation`                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| jummp.security.resetPassword.email.body      | The message template will be sent to users wishing to reset their password. The pre-defined variables, i.e. `{{REALNAME}}`, `{{USERNAME}}`, `{{PASSWORD}}`, will be replaced by actual values.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Example:                                     | `jummp.security.resetPassword.email.body=Dear {{REALNAME}}, \r\n\r\nA password reset request was received for your account. Please reset your password by going to `{{URL}}`.\r\n\r\nMy Model Repository`                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| jummp.security.resetPassword.email.subject   | The subject of the resetting password email will be sent to the existing user.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Example:                                     | `jummp.security.resetPassword.email.subject=[My Model Repository]`                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
### Database settings

**jummp.database.type**
The type of database server using for Model Repository. The values the current development supports are MYSQL, POSTGRESQL, MARIADB. To use the latter, please set the value to 'MYSQL'. Oracle Databases are not supported.

**Example:**
```plain
jummp.database.type=MYSQL
```

**jummp.database.server**
Indicate the name of database server where the database server is running.

**Example:**
```plain
jummp.database.server=mysql-test
```

**jummp.database.database**
Indicate the name of the database that should be used by the model repository installation.

**Example:**
```plain
jummp.database.database=model_repository
```

**jummp.database.username**
Indicate the user in charge of performing operations on the database.

**Example:**
```plain
jummp.database.username=dbadmin
```

**jummp.database.password**
Indicate the password corresponding with the database user.

**Example:**
```plain
jummp.database.password=s3cr3t
```

**jummp.database.port**
Indicate the port that should be used to connect to the database server.

**Example:**
```plain
jummp.database.port=3306
```

### Model identifier settings

The repository assigns a unique and immutable identifier to the model as part of the submission process and can, optionally, be modified to assign another identifier once the model has been published. The format of the identification schemes is customisable and described at https://bitbucket.org/jummp/jummp/wiki/Model%20identification%20scheme#markdown-header-envisaged-workflow-for-setting-up-model-identifier-generators

**jummp.model.id.submission.partN.suffix**
Indicate the suffix of a part of the model identifier generated when a new model is submitted to the platform. “partN” could stand for part1, part2 or any other part N, so long as parts 1..(N-1) are also defined.

Example:

```
jummp.model.id.submission.part1.suffix=MODEL
```

**jummp.model.id.submission.partN.type**

Indicate the type of a part of the model identifier that gets assigned during submission. Valid options are numerical, date or literal. “partN” could stand for part1, part2 or any other part N, so long as parts 1..(N-1) are also defined.

Example:

```
jummp.model.id.submission.part4.type=date
```

**jummp.model.id.submission.partN.fixed**

Indicate that the n\textsuperscript{th} part of the model identifier scheme used for submissions is subject to change. In the case of numerical parts, the suffix gets increments after each submission, in the case of date parts, the suffix changes when the date, as specified by the chosen date format, changes.

Example:

```
jummp.model.id.submission.part2.fixed=false
```

**jummp.model.id.submission.partN.width**

Indicate that the n\textsuperscript{th} part of identifiers generated in the submission process, which must be numerical in nature, should be padded to the specified width.

Example:

```
jummp.model.id.submission.part3.width=10
```

**jummp.model.id.submission.partN.format**

Indicate the date format that the n\textsuperscript{th} part of the submission identification scheme should use. partN.type must be of type date. In addition, this format must comply with the URI specification [http://www.ietf.org/rfc/rfc3986.txt](http://www.ietf.org/rfc/rfc3986.txt). See [https://docs.oracle.com/javase/7/docs/api/index.html?java/text/SimpleDateFormat.html](https://docs.oracle.com/javase/7/docs/api/index.html?java/text/SimpleDateFormat.html) for the syntax that can be used to specify the desired date format.

Example:

```
jummp.model.id.submission.part2.format=yyyyymmdd
```

### Solr settings

**jummp.search.url**

The URL to access a running Solr instance. If Model Repository and Solr are resided in the same machine, this property could be localhost.

Example:

```
jummp.search.url = http://localhost:8983/solr
```

**jummp.search.folder**

Indicate the location where the Repository’s Solr core should be stored. The search configuration settings as well as the necessary Solr indices will reside in a subfolder of this path.
### Example:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>jummp.search.folder</td>
<td>/homes/tung/devtools/solr-5.4.0/server/solr</td>
</tr>
<tr>
<td>jummp.search.pathToIndexerExecutable</td>
<td>Indicate the path of the Repository's indexer jar. This binary file can either be built from sources, or downloaded.</td>
</tr>
<tr>
<td>Example:</td>
<td>jummp.search.pathToIndexerExecutable=/homes/tung/repository/indexer.jar</td>
</tr>
</tbody>
</table>

### Server settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jummp.server.protection</td>
<td>Set to true if you require all pages to be protected by authentication.</td>
</tr>
<tr>
<td>Example:</td>
<td>jummp.server.protection=false</td>
</tr>
<tr>
<td>jummp.server.url</td>
<td>Indicate the url that will be used to access the instance of the Model Repository.</td>
</tr>
<tr>
<td>Example:</td>
<td>jummp.server.url=<a href="http://example.com/model-repository">http://example.com/model-repository</a></td>
</tr>
</tbody>
</table>

### Version Control Systems settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jummp.vcs.plugin</td>
<td>Indicate the version control system that should be used to manage models. The only supported value at the moment is git.</td>
</tr>
<tr>
<td>Example:</td>
<td>jummp.vcs.plugin=git</td>
</tr>
<tr>
<td>jummp.vcs.exchangeDirectory</td>
<td>The folder where accessed models should be checked out from the version control system. This folder must exist and the user running the Model Repository must have read and write access to it.</td>
</tr>
<tr>
<td>Example:</td>
<td>jummp.vcs.exchangeDirectory=/home/tnguyen/model-repository/exchange</td>
</tr>
<tr>
<td>jummp.vcs.workingDirectory</td>
<td>The root folder where models should be stored. This folder must exist and the user running the Model Repository must have read and write access to it.</td>
</tr>
<tr>
<td>Example:</td>
<td>jummp.vcs.workingDirectory=/home/tnguyen/model-repository/working</td>
</tr>
</tbody>
</table>

### Context Specific Help settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>These settings indicate the location of the contextual help that is available for the users of the Model Repository.</td>
</tr>
</tbody>
</table>
2.2 Prerequisites for installation model repository from source code

2.2.1 Installing Apache Maven

Download Apache Maven from the link: https://maven.apache.org/download.cgi

The installation of Apache Maven is a simple process of extracting the archive and adding bin folder with the mvn command to the PATH environment variable. The detailed steps are presented as follows:

**Step 1:** Ensure JAVA_HOME environment variable is set and points to your JDK installation.

**Step 2:** Extract Maven’s distribution archive into your favourite directory.

unzip apache-maven-3.3.9-bin.zip

or
tar xzvf apache-maven-3.3.9-bin.tar.gz

Alternatively use your preferred archive extraction tool.

**Step 3:** Add the bin directory inside the just created directory apache-maven-3.3.9 to the PATH environment variable.

**Step 4:** Check the setting up process by running the command mvn -v in a new terminal tab. The expected result should look like to

Apache Maven 3.3.9 (bb52d8502b132ec0a3f4c09453c07478323dc5; 2015-11-10T16:41:47+00:00)
Maven home: /homes/tnguyen/DevTools/apache-maven-3.3.9
Java version: 1.7.0_79, vendor: Oracle Corporation
Java home: /homes/tnguyen/DevTools/jdk1.7.0_79/jre
Default locale: en_GB, platform encoding: UTF-8
OS name: "linux", version: "2.6.32-573.12.1.el6.x86_64", arch: "amd64", family: "unix"
See more details of installing Apache Maven from the link: https://maven.apache.org/install.html

2.2.2 Installing Grails

It is worth noting that the DDMoRe Model Repository has been developed on Grails 2.3.11. To be sure whether Grails 2.3.11 has already installed in your computer, run the following commands in a terminal.

Unix
$ echo $GRAILS_HOME
/home/tnguyen/devtools/grails-2.3.11
$ which grails
/home/tnguyen/devtools/grails-2.3.11/bin/grails

Windows
$ echo %GRAILS_HOME%
C:\Users\Tung\devtools\grails-2.3.11
$ where grails.bat # available on Windows Server 2003 and later
C:\Users\Tung\devtools\grails-2.3.11\bin\grails.bat

If not, it can be downloaded from http://dist.springframework.org.s3.amazonaws.com/release/GRAILS/grails-2.3.11.zip

The installation process of Grails is rather similar to Maven's:

Step 1: Ensure JAVA_HOME environment variable is set and points to your JDK installation.

Step 2: Extract the archive into your preferred directory.
unzip grails-2.3.11.zip

Step 3: Create a GRAILS_HOME environment variable that points to the path where you have unpacked the archive (e.g., /opt/grails on Unix or C:\grails on Windows)

Step 4: Append a reference to the bin directory of the Grails installation to the PATH environment variable (e.g., %GRAILS_HOME%/bin on Unix or %GRAILS_HOME%\bin on Windows)

Step 5: Verify the configuration by running the following command in a new console:
grails -version

If the output shows Grails version: 2.3.11, it means that you have successfully installed Grails.

2.2.3 Checking Git

As the platform’s source code is managed by Git, it is necessary to have a relatively-recent version of the version control system installed on the machine on which the required application binaries will be produced. The lowest Git version the system supports is 1.8. To be sure you have the supportable version, typing the following command to verify it:
git --version

The output looks like git version 2.6.3.windows.1 (in Windows) or git version 1.8.1 (in Linux).
If Git is not installed, or if the available version of Git predates 1.8.1, then install version 1.8 or newer from either http://git-scm.com/ or, if applicable, your Linux distribution’s package manager.

2.3 Prerequisites for installation model repository from pre-built binaries

Currently, you do not have to meet any further requirements if you intend to install Model Repository from pre-compiled binaries.

2.4 Prerequisites for deploying model repository

2.4.1 Setting up the application server

The default installation of Tomcat 7 (downloadable from http://tomcat.apache.org/download-70.cgi) is assumed. This entails the environment variable CATALINA_HOME being set and $CATALINA_HOME/bin being present on the system path. The default memory settings for Tomcat are insufficient for the application's needs and hence need to be modified.

On Unix, this can be done by creating a file named setenv.sh in the $CATALINA_HOME/bin directory. The file should contain the line

```bash
export JAVA_OPTS="-server -noverify -Xms256M -Xmx4G -XX:MaxPermSize=256M -XX:+UseConcMarkSweepGC -XX:+UseParNewGC"
```

Be sure that the new file is executable by running chmod +x setenv.sh

On Windows, simply create %CATALINA_HOME%/setenv.bat with the following contents

```bash
set JAVA_OPTS=-server -noverify -Xms256M -Xmx4G -XX:MaxPermSize=256M -XX:+UseConcMarkSweepGC -XX:+UseParNewGC
```

Note the lack of quotes for the JAVA_OPTS value under Windows.

It is strongly recommended not to run other applications on the Tomcat installation running the Repository.

The Tomcat server would need to be restarted in order for the new settings to be applied.

2.4.2 Running Solr Server

You should ensure Solr is running before deploying the application. If the command 

```
jps -m <machine_where_solr_is_running>
```

yields output similar to

```
1310 jar -XX:OnOutOfMemoryError=/opt/solr/solr-5.4.0/bin/oom_solr.sh 8983 /opt/solr/solr-5.4.0/server/logs --module=http
```

then Solr is already running. Alternatively, please start Solr.

If you already appended a reference to the bin directory within the Solr installation directory to your PATH environment variable, just type the following command in a terminal:

```
solr start
```
If PATH does not contain $SOLR_HOME/bin, you will need to specify the full path to the solr executable, e.g. $SOLR_HOME/bin/solr start

By default Solr will start on port 8983. If you wish to use a different port, simple indicate this as a command line argument

solr start -p <your_desired_port_number>

For the remainder of this document we will assume that Solr is running on localhost:8983. To verify that Solr has started successfully, navigate to the address http://localhost:8983/solr. Below is the screenshot of the Solr Admin interface that you should see.

3 INSTALLATION FROM SOURCE CODE

This section describes the steps required for producing the artefacts required for deploying the Model Repository starting from a fresh checkout from version control. If you wish to use the binary artefacts that are already available, please skip to chapter four.

3.1 Building search indexer

3.1.1 Clone

git clone https://bitbucket.org/jummp/jummpindexer

3.1.2 Build

Step 1: Enter the directory where you have cloned the jummpindexer project.
Step 2: Run the following command:
mvn package

Your target folder should now contain the JummpSolrIndexer-1.1-SNAPSHOT.jar
3.2 Building Model Repository

3.2.1 Clone the repository

```
git clone https://bitbucket.org/jummp/jummp.git model-repository
```

3.2.2 Database schema and initial content

⚠ Make sure Solr is running in advance.

The database structure must be set up prior to running the Model Repository. To achieve this, open `grails-app/conf/DataSource.groovy`, uncomment this line (line number 112):
```
dbCreate = "create"
```
Then, just running the grails command belows:
```
grails>dbm-changelog-sync
```
Finally, go back to `grails-app/conf/DataSource.groovy` and undo/revert the previous modification (i.e. comment that line).

The database structure required by the Repository should now be in place and any future changes to the schema should be ‘replayable’ onto the existing one.

3.2.3 Generating the binary archive

By default, the binary will be called jummp.war, but this can be changed by editing `application.properties` so that the property `app.name` is set to `model-repository` or whatever matching with your mind. Then, using a shell console, run:
```
grails prod war
```
in the newly-created `model-repository` folder. This instructs Grails to resolve the project's dependencies, compile all classes and package them in a web archive that will be created in the target subfolder.

4 INSTALLATION FROM PRE-BUILT BINARIES

It is also possible to install DDMoRe Model Repository from the pre-built binary files.

4.1 Download pre-built binaries

You could download jummp.war, indexer.jar, WeceemSecurity.groovy and the initial sql structure and data, and put them in appropriate directories.

Download `model-repository.war`:
```
https://bitbucket.org/jummp/jummp/downloads/model-repository.war
```

Download `indexer`:
```
https://bitbucket.org/jummp/jummp/downloads/JummpSolrIndexer-1.1-SNAPSHOT.jar
```

Download `WeceemSecurity.groovy`:
```
```
4.2 Initialise content for model repository database

Because of installing model repository from precompiled binaries, you don't have to adjust any lines of source code in order to initialise content for model repository database. Instead you can use any GUI clients, such as phpMyAdmin, MySQL Workbench, Webmin MySQL module, etc., to import the initial data from jummp_dump.sql file into your existing database. We provide here two ways to import those data into the existing database.

**Method 1: Logged mysql shell client in**

Running the following command:

```
mysql>use model_repository_database;
mysql>source path/to/jummp_dump.sql;
```

**Method 2: Not yet logged mysql shell client in**

Running the following command at your favourite console unless you are not sure model repository database credential.

```
mysql -u database_username -p database_name < path/to/jummp_dump.sql
```

5 DEPLOYMENT

We assume your machine complies with requirements outlined in Section 2 and you already have the necessary files that are explained in Sections 3 and 4. This step goes through the procedure of the integration all of them working together smoothly. In fact, deploying Model Repository is a process of transferring war file or source codes to a configured application server and starting the installation procedure. Since Apache Solr is integrated into the application supporting the model searching component, it is required to start first.

As mentioned above, Model Repository is packed in a war file. Assuming that Tomcat is configured with `autoDeployOnStartup`, the deployment of Model Repository could be performed as belows:

- Stop tomcat/application server.
- Copy the war file to webapps folder on Tomcat installation folder.
- Start tomcat and wait for until the application is deployed successfully.


Please note that it is not possible to rely on the Tomcat Windows Service to manage a Tomcat installation hosting the Repository on Windows -- please use the scripts bundled in $CATALINA_HOME/bin instead.
6 ADMINISTRATION OF DDMoRe MODEL REPOSITORY

Open a web browser and type the address that you chose for the `jummp.server.url` configuration setting. To facilitate the rest of the guide, we will assume the model repository home page url: `http://example:8080/model-repository`.

6.1 Using the administrator account

An administrator account is created by default when the application starts for the first time. The account details are:

username: administrator
password: administrator

Visit `http://example:8080/model-repository/login/auth` and enter the credentials in order to authenticate. You will be greeted with the following page:

Following a successful login, the homepage will feature on the right-hand-side the administration panel which includes several links that facilitate the customisation of the Repository installation.

⚠ You are strongly encouraged to change the password with a secure one after the first login. This can be done by clicking on the Profile link in the top right section of the page.
6.2 Adding a new user

From the home page logged in by administrator, point and click to the link on User Administration. The link is http://example:8080/model-repository/userAdministration/index

To add a new user, click the link Add new user (the link is http://example:8080/model-repository/userAdministration/register) at the top of the user table. Simply fill in the form that appears, ensuring that the email address is unique and click Register for the new account to be created. If configured to do so, the Repository will send a confirmation email to the new user.

6.3 Customisation of access control

6.3.1 Overview of available roles

Non-authenticated users can search, browse, view and download published models. These features are also available to logged-in users with any of the roles below.
ROLE_USER
The default role for anyone that has created an account, this grants users permission to submit and update private models. Users can also choose to share their private models with other collaborators. The search results available to users with this role span all models that the user has access to, whether public and private, containing at least one version that matches the user's query.

ROLE_CURATOR
Newly-registered accounts are automatically granted this role unless the setting `jummp.security.curatorByDefault` was set to false. Curators are users that act as content administrators, being able to make models available without the need for authentication.

ROLE_QC_PROVIDER
The QC_PROVIDER role exists for situations when model certification cannot be performed by the same people that publish models, nor by administrators. In this situation, the setting `jummp.security.certificationRole` can be set to 'ROLE_QC_PROVIDER'. See the documentation for `jummp.security.certificationRole` to learn more about model certification. This role need not be used in all installations of the Repository.

ROLE_ADMINISTRATOR
Super-user role that has access to all models, private and public, thus bypassing access-control lists for the models and also to the administrative interface within the Repository. Users with this role are expected to be familiar with computer administration tasks and able to install, customise and maintain a Model Repository installation so that it serves the users’ needs.

6.3.2 Viewing and/or modifying user roles
From the Model Repository’s home page, you can enter the user administration functionality via the right menu item User Administration or by accessing the link `http://example:8080/model-repository/userAdministration/index` directly. The User Administration Interface should be shown as the image in the section 6.2 of this document. Then click on the `Id` field of the user whose permissions should be modified. You can see a screenshot below:
By default, a newly-created user should be assigned the role User. As a demonstration from the below image, the user tnguyen could be assigned the available roles: curator and admin. Just click to the links: Remove Role from User and Add Role to User in order to change which permissions you want to apply to the user tnguyen. Click the button Update User to save the changes.

6.4 Changing the text on the front page of the Model Repository
You are advised to change the default text on the homepage so that it suits your organisation.

To do this, log into the model repository with an administrative account, and click the item Content Management System on the right-hand-side administrative menu of the homepage. Alternatively, you can also navigate to http://example:8080/model-repository/wcm-admin.

In the Weceem content management window (shown as the image below), click on the file called index.
This will present you with an HTML editor allowing you to change the default text to something more suitable to your needs.

**Edit HTML**

This HTML code is used to display a message about the LICENSE file.

Then, click **Save and continue editing** or **Save** to store what you have adjusted.