|  |
| --- |
|  |
| Open Reports System Operations Manual |
|  |

Table of Contents

[Table of Contents 3](#_Toc524538859)

[Wildfly Server 4](#_Toc524538860)

[Server Requirements 4](#_Toc524538861)

[Running Wildfly 4](#_Toc524538862)

[Stopping Wildfly 4](#_Toc524538863)

[Setting up for the First Time 4](#_Toc524538864)

[Deploying a Module 6](#_Toc524538865)

[Undeploying a Module 7](#_Toc524538866)

[Tomcat Server 7](#_Toc524538867)

[Server Requirements 7](#_Toc524538868)

[Setting up for the First Time 7](#_Toc524538869)

[Running Tomcat 7](#_Toc524538870)

[Deploying a Module 7](#_Toc524538871)

[Undeploying a Module 8](#_Toc524538872)

[Testing 8](#_Toc524538873)

[Population of Data DB 8](#_Toc524538874)

[Metadata Table Structure 9](#_Toc524538875)

[Metadata Column Meaning 9](#_Toc524538876)

[Table Descriptions 10](#_Toc524538877)

[Insert Sequence 11](#_Toc524538878)

# Wildfly Server

## Server Requirements

* Apps
  + Jboss Wildfly 11.0.0
  + Redis 3.2.10
* Machine Specs
  + Memory: 11g
  + Disk: 26g
* Misc
  + Mirror disk for finished reports connected to 10.10.2.54 (Tomcat Server)
* Server Connection
  + Connecting to the server via SSH using command (*ssh -P2222 root@10.10.2.69)*
    - Password: St@g1ng2017

## Running Wildfly

1. Go to <wildfly\_home>/bin
2. Run *redis-server & ./standalone.sh -b=0.0.0.0 -bmanagement=0.0.0.0 -c standalone-full.xml &*

## Stopping Wildfly

1. Go to <wildfly\_home>/bin
2. Execute ./jboss-cli.sh --connect command=:shutdown

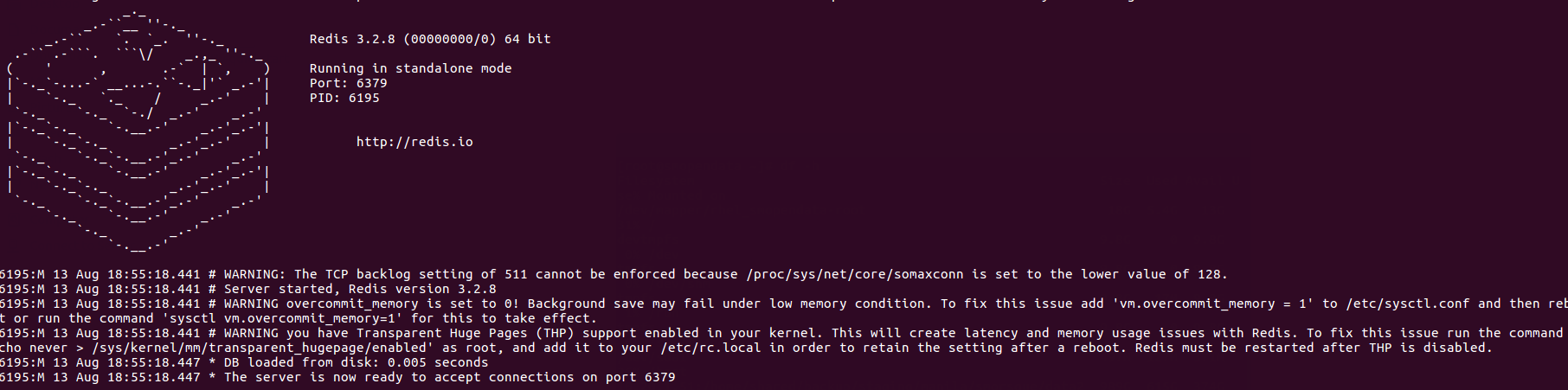
For restart execute ./jboss-cli.sh --connect command=:reload

## Setting up for the First Time

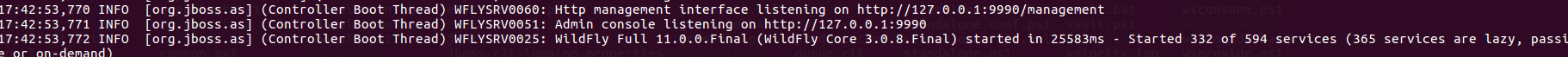
1. Download and install Redis Server 3.2.10 [here](https://redis.io/download). Installation is also included in the link
2. Redis should be running by doing these steps
   1. Execute *redis-cli* and you should see this



* 1. Type in *ping* and the redis server should respond PONG.
  2. If server is not yet running, execute *redis-server &* and you should see



1. Download Jboss Wildfly [here](http://download.jboss.org/wildfly/11.0.0.Final/wildfly-11.0.0.Final.tar.gz)
2. Extract the tar file using *tar -xvf*
3. In your local dev repository, go to <project location>/queuegtw. Copy the following files to Wildfly server via scp to <wildfly\_home>/bin/:
   1. queue.cli
   2. user.sh
4. Go to <wildfly\_home>/bin/
5. Execute *./standalone.sh -b=0.0.0.0 -bmanagement=0.0.0.0 -c standalone-full.xml &*
6. You should see this after the server runs the app

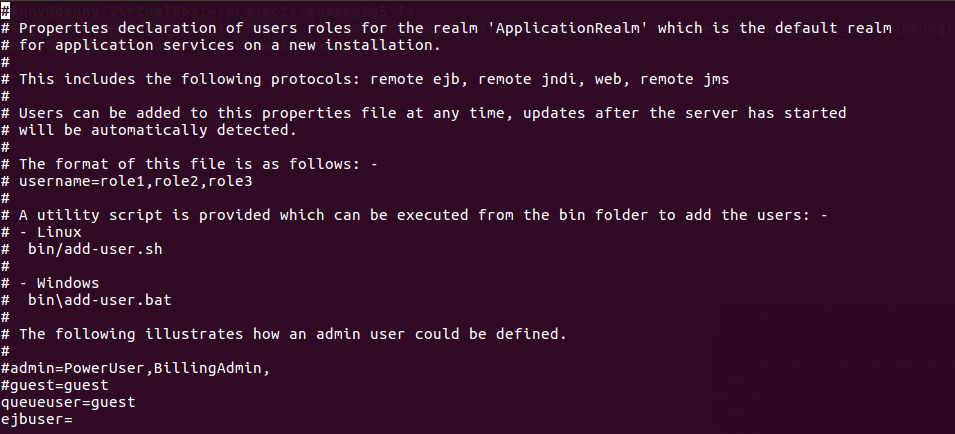


1. Open another console window then connect again to the wildfly server
2. Execute the following
   1. *./jboss-cli --file=queue.cli*
   2. *bash user.sh*
3. Go to <wildfly\_home>/standalone/configuration/ then edit the file application-roles.properties. Add the following lines in the property file:

*queueuser=guest*

*ejbuser=*

It should look something like:



Notes:

* Wildfly will not run successfully if redis server is not running so ensure that redis is running.

## Deploying a Module

*Notes:*

* *Server must be running to deploy a module*
* *Compilation of module is not done in staging/production server but in dev local machine*
* *Compiled modules from dev local machine are copied via SSH*
* *<wildfly\_home>/deployments* should already have existing modules copied from dev local machine (see dev setup section)
* *There are two modules existing in the wildfly server*
  + *queuegtw\_queuelistener-0.0.1-SNAPSHOT.war*
  + *file\_listener-0.0.1-SNAPSHOT.war*

1. Go to <wildfly\_home>/bin/
2. Execute *./jboss-cli.sh --connect* you should see the following after execution



1. Execute *deploy ../deployments/queuegtw\_queuelistener-0.0.1-SNAPSHOT.war*. After issuing the command the server will load the module and wait for it to finish.
2. Execute *deploy ../deployments/file\_listener-0.0.1-SNAPSHOT.war.* After issuing the command the server will load the module and wait for it to finish.

## Undeploying a Module

1. Type in *undeploy ../deployments/queuegtw\_queuelistener-0.0.1-SNAPSHOT.war*
2. Type in *undeploy ../deployments/file\_listener-0.0.1-SNAPSHOT.war*

# Tomcat Server

## Server Requirements

* Apps
  + Tomcat 8
* Machine Specs
  + Memory: 19g
  + Disk: 18g
* Misc
  + Mirror disk connected to 10.10.2.69 (Wildfly Server)
* Server Connection
  + Connecting to the server is via SSH using command (*ssh -p 2222 denny.serrano@10.10.2.54*)
    - Password: ch@ng3me@dennys
* File locations
  + Tomcat\_home: /opt/tomcat/
  + Report mirror disk location: /mnt/data2

## Setting up for the First Time

1. Download Tomcat 8 [here](http://mirror.rise.ph/apache/tomcat/tomcat-8/v8.5.32/bin/apache-tomcat-8.5.32.tar.gz)
2. Extract the tar file using *tar -xvf*
3. Try to run first tomcat to see if its working by going to <tomcat\_home>/bin then execute *./startup.sh && tail -f ../logs/catalina.out*. The server console should look something like when its done:



## Running Tomcat

1. Go to <tomcat\_home>/bin/
2. Execute *./startup.sh && tail -f ../logs/catalina.out*

## Deploying a Module

*Notes:*

* *Compilation of module is done in dev local environment and not in staging/production server*
* *Compiled modules are copied via SSH to the server*
* *There is only one module for Tomcat Server named Root.war and is located in <tomcat\_home>/webapps/*
* *Modules can be deployed either tomcat is running or not*

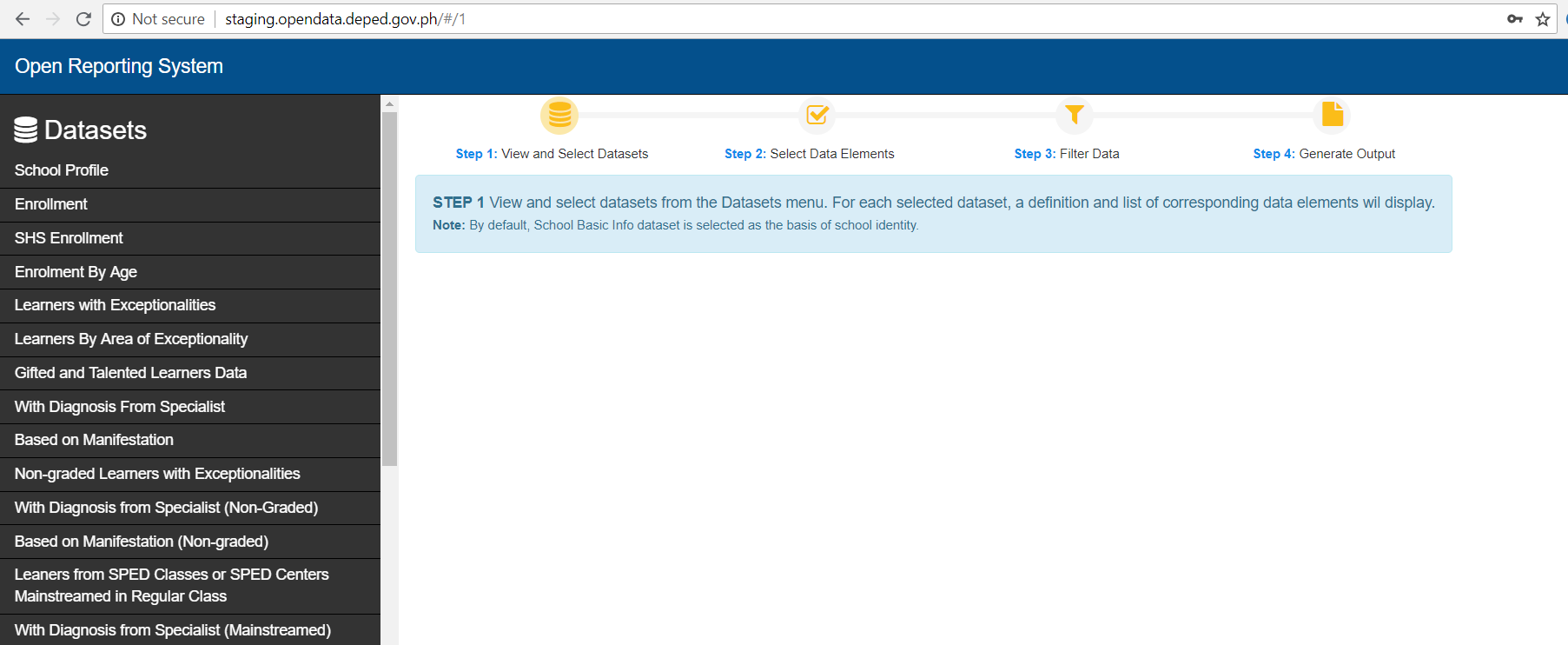
1. If tomcat is running, just copy the war file to <tomcat\_home>/webapps/ then tomcat will automatically deploy the module
2. If tomcat is not running, copy the war file to <tomcat\_home>/webapps/ then follow instructions under (Running the tomcat Server (After first time setup)).

## Undeploying a Module

1. If tomcat is running, go to <tomcat\_home>/webapps/ then remove target war file from this directory. The server will automatically undeploy the module
2. If tomcat is not running, go to <tomcat\_home>/webapps/ then remove target war file from directory then follow instructions under (Running the tomcat Server (After first time setup)).

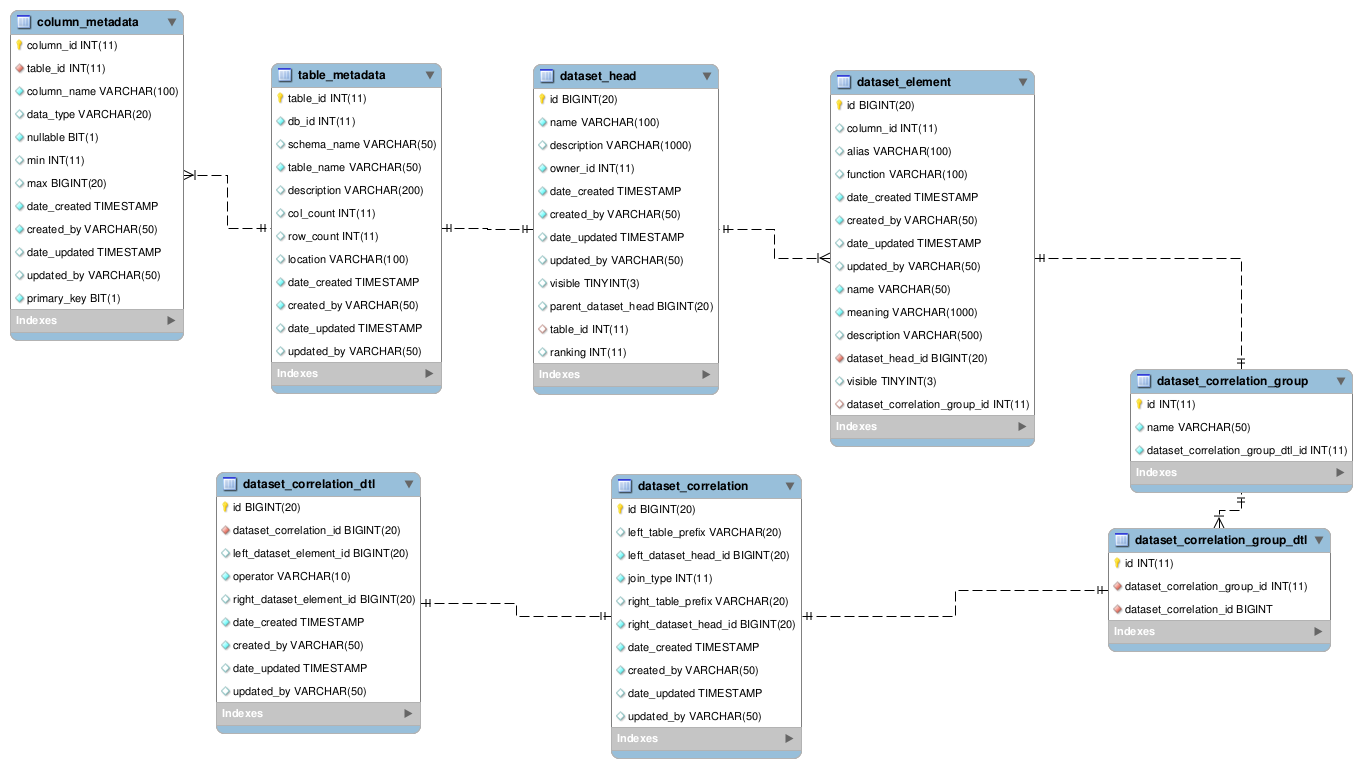
## Testing

1. Access through browser the Tomcat Server (10.10.2.54) or using domain name <http://staging.opendata.deped.gov.ph> the browser should be like:



# Population of Data DB

## Metadata Table Structure



In order for a dataset to appear in the UI, the following tables should be filled up hierarchically.

## Metadata Column Meaning

Below are the important and mandatory fields for metadata and some short description



## Table Descriptions

Table\_metadata – contains info of the physical table. The table\_name should be the same of the table name present in the database because this is the one generated in SQL

Column\_metadata – contains info of the physical fields. The column\_name should be the same of field names present in the table in the database. This would appear in the generated SQL

Dataset\_head – contains info for the virtual datasets in the UI. This is more like the virtual representation of the table\_metadata

Dataset\_correlation\_group – contains different groups used for auto joining tables

Dataset\_correlation\_group\_details – A group could contain one or more group details and each group detail contains one dataset correlation

Dataset\_correlation – defines what dataset heads are joined

Dataset\_correlation\_dtl – defines from the dataset heads, what are the columns joined

## Insert Sequence

* Table Metadata
  + Column Metadata
  + Dataset Head
    - Dataset Element
      * Dataset Correlation Group
        + Dataset Correlation Group Detail

Dataset Correlation

Dataset Correlation Detail