**Moqui-Multi-Instance**

**Case 1).** **When You have to use Pre-built-in-image of Production**

* Follow **step 1. - > 1** for enabling remote-connect for docker-client.
* Follow **step 1. - > 3 to 7** for enabling nginx-container and configure-docker-client.
* **Follow step 2. - >(\*)** (all) for enabling mysql-configurations.
* Follow **step - >3** for Docker-Images-Pulling-configurations **.**
* And after that just create an instance (System - > instances -> create-app-instance) and select the image-configured.
* And click the INIT button (it will fetch the image in some time).
* Refresh the page for checking if the image is pulled or not , Instance-Exist you got on UI in yellow-highlighted form.

**Case 2). For image-Building Follow**

* **step 1. - > 2**

**Steps :-**

**1)Docker-Side-Configurations-For-Remote-connect & image-building-Process**

moqui.server.instance.InstanceHost

1. **Run Cmd for exposing 2375 ports for remote-docker ,so that from anywhere we can connect to the docker client.**
   * Dockerd -H unix:///var/run/docker.sock -H tcp://127.0.0.1:2375
   * If this cmd create problem in the case of localhost Refer this linux:
   * <https://gist.github.com/styblope/dc55e0ad2a9848f2cc3307d4819d819f>
   * **For Mac Run cmd after installing** :
   * socat TCP-LISTEN:2375,reuseaddr,fork UNIX-CONNECT:/var/run/docker.sock &
2. **If Building Image then needed to follow below steps:-**
   * Elastic search needed and then build war file
   * ./gradlew downloadElasticSearch
   * **mysql jar file must be needed** -> inside runtime/lib folder must otherwise it would not work
   * ./gradlew cleanAll load
   * docker cleanAll build addRuntime
   * cd docker/simple
   * ./docker.sh (for building the docker-image)
   * cd docker
   * #comment down the mysql-container part inside nginx-mysql-compose.yml
   * docker-compose -f nginx-mysql-compose.yml -p moqui up -d
   * ./gradlew -> ./gradlew run
3. **Configure-docker-client Remote/local inside moqui.server.instance.InstanceHost** for connecting remote-docker ,just make a new Entry for Remote-Docker and give the IP of the server.
4. (#comment down the mysql-container part inside )
5. docker-compose -f nginx-mysql-compose.yml -p moqui up -d - > For running nginx-container.
6. **For enabling nginx-https service we have to generate the certs cmd are given already.**
7. **Before making nginx-container run ,always create a moqui-default network and run on that in case of a compose file no need to worry.**

**—--------------------------------------------------------------------------------**

**2)Moqui-Mysql-DB-Configuration**

moqui.server.instance.DatabaseHost

moqui.server.instances.DatabaseType

1. **This can be Remote/local ,so we have to provide the IP of the Mysql in (Instance-Address and Host-address)** moqui.server.instance.DatabaseHost **Entity and also provide the root-user and root-password which has all privileges.**
2. **DatabaseType inside the** moqui.server.instances.DatabaseType mysql8 **must give otherwise error MysqlXADataSourceNotFound.**
3. **And At Mysql-Client side we have to do Remote-configuration steps for enabling remote access in mysql (Given in Extra-Info point-3).**
4. **And Create root-user for remote/local must provide all privileges and also permission (so that it can provide permission to other users also )(Given in Extra-Info point-3).**
5. **Mysql-SSl Must be configured(**[**https://access.redhat.com/solutions/68098**](https://access.redhat.com/solutions/68098)**) follow upto 10 steps.**
6. **For DB thing i have changed this code -: Time-zone-error**
7. serverTimezone="${database\_time\_zone ?: 'US/Pacific'}" useSSL="false" allowPublicKeyRetrieval="true"
8. **Here I am Getting Bitronix error because the mysql-user has not all permissions for whole db ,or might be previous way is deprecated ,issue is resolved by**
9. ec.entity.runSqlUpdateConf("GRANT ALL PRIVILEGES ON \*.\* TO '${envMap.entity\_ds\_user}'", adminMap)

—-----------------------------------------------------------------------------------------

**3)Docker-Images-Pulling-configurations -: In this section we are configuring images ,whose instance we have to create and configuring credentials ,from where we have to fetch/pull images.**

**-And after following things just create an instance and select the image , which you have created.**

**moqui.server.instance.InstanceImage**

* **Instance-Image-Id** : Must-be-unique-name/PK
* **Image-Type-ID -:** it's our choice of categorization
* **Host-Type-Id -:** it's our choice of categorization
* **Image-Name -** full-name of image must be given serveraddress/image-name:tag
  + **Aws :** 118728153888.dkr.ecr.us-east-1.amazonaws.com/tk
  + **Azure** : rohitmkj.azurecr.io/moquil
  + **Docker-Hub** :- rohitroax123/moquil:latest
  + **Docker-container-Private-registry** : [localhost:5000](http://localhost:5000/v2/%22,/%22email/%22:/%22rohitpawar28112000@gmail.com/%22%7D%22)/moquil:latest
* **Registry-Location :** it contains the Registry-server-address now.
  + **Docker-container-private-registry** :<http://localhost:5000/v2/>
  + **Docker-Hub :** https://index.docker.io/v1/
  + **AWS-ECR-registry-server-address** :[https://118728153888.dkr.ecr.us-east-1.amazonaws.com/v2/](https://118728153888.dkr.ecr.us-east-1.amazonaws.com/v2//%22,/%22email/%22:/%22rohitpawar28112000@gmail.com/%22%7D%22)
  + **AZURE -REGISTRY -:** [rohitmkj.azurecr.io](http://rohitmkj.azurecr.io)
* **Username :** Name of user from which we can login to docker , In case of Aws yet it is hard-coded to use process cmd.
* **Password :** password/ command in case of Aws (through which token is generated)

aws ecr get-login --no-include-email --region us-east-1

—----------------------------------------------------------------------------------

**For Extra-Informations :-**

**1).**

**moqui.server.instance.DatabaseHost :- Instance-Address**(It is for the instance which we are creating ,the given ip here is passed as envs in the instance ) and **Host-address** (it is the root mysql-address)

**But i think both are always same because we can’t use one mysql-root-user to provide-privileges to another mysql-client-user.**

**Special-case :** Always give IP in Instance-Address of instance because the instance is inside the docker-client ,and so it is trying to connect remotely ,if you give localhost it will not be able to connect mysql-outside the docker-client

**2).**

**Point to be Remember :**

1. **Moqui-Image** must to be not **containing local-mapping of DB** ,if contain then it will not be able to connect the DB with the given IP , because b/w envs and local declaration of DB precedence goes to local-declaration of DB
2. And also same for **MAIN-INSTANCE** it should not contain the Local-Db mapping it is coming from envs and we want that because it might happen that db might be Remote.

**3).**

**Remote-Mysql-Configurations : -**

**1)For instance-Address field** inside moqui.server.instance.DatabaseHost (always give the IP ,where mysql is placed).

For Local-Mysql setup this thing is enough to allow remote connection.

* And also i have to change mysql->mysql.conf.d->mysqld.cnf file make \* in the

bind-address = \*

mysqlx-bind-address = \*

* systemctl restart mysql

**2)For Creating Remote User-use** this(I would recommend always use Remote-User)

* **For access the mysql from Remote(any ip) we have to create user with (%)**
* **Create User ‘root’@’%’ IDENTIFIED BY ‘123456’** and allow all permission
* **GRANT ALL PRIVILEGES ON \*.\* TO 'root'@'%'**
* **GRANT PROXY ON ``@`` TO 'root'@'%' WITH GRANT OPTION**
* Must above cmd it gives permission to user that it will also give privileges to other user.
* And then use mysql -h <wlan>IP -u<user-name> -p (check by ifconfig)