**BMC Helix backend microservices**

1. **Setting up API Servers & Docker Services**

<https://docs.bmc.com/docs/is221/setting-up-a-local-developer-sandbox-environment-1039589991.html>

A screenshot of a computer program

Description automatically generated

**!IMPORTANT**

**Sandbox-scripts checks for Docker (WSL2) CPU & Memory limits**

Default WSL2 configures limits to:

12 CPU

Memory 7GB (Half of machine physical Memory)

**Docker:**

**To customize WSL2 memory limit up to 12Gb**

1. Go to %UserProfile%

2. Make a **.wslconfig** file in **%UserProfile%**

3. .wslconfig contents:

**[wsl2]**

**memory=12GB**

**processors=8**

Finally, install Docker Desktop (So you don't have to restart it later on)

A screenshot of a computer

Description automatically generated

**!IMPORTANT**

**C:\sandbox\\*.tar.gz**

**C:\sandbox\scripts will NOT have admin access to 'postgres' folder spawned by Docker images**

**Need to Git Bash:**

**cd /c/sandbox;**

**mkdir .\sandbox;**

**// Verify by**

**ls -la /c/sandbox/sandbox;**

**// Give full admin access to /c/sandbox/sandbox**

**chmod 777 /c/sandbox/sandbox;**

**// Move all the files (\*.tar.gz & \*.sh & \*.yml to /c/sandbox/sandbox)**

**// It should be**

**/c/sandbox/sandbox/\*.tar.gz;**

**/c/sandbox/sandbox/\*.yml;**

**/c/sandbox/sandbox/\*.sh;**

**// Edit sandbox.yml to allow full system access to postgres**

**// Locate line 17**

**- PGDATA=/var/lib/postgresql/data:rw**

A screen shot of a computer

Description automatically generated

**Then, run ./setup-sandbox.sh**

**cd /c/sandbox/sandbox;**

**// VIM to edit ./setup-sandbox.sh => Line 350 => Add -ipv6 > /dev/null 2>&1**

**// Otherwise, sandbox cannot recognize the AR server is up or not**

A computer screen with text

Description automatically generated

**// VIM to edit ./start-sandbox.sh too**

**// Line 115 => add -ipv6 > /dev/null**

A computer screen with text

Description automatically generated

**// Run setup-sandbox.sh**

**./setup-sandbox.sh;**

**Enter default password to create instances:**

A black screen with white text

Description automatically generated

**Good sign:**

**Postgresql will start restoring dev db in Git Bash**

A screen shot of a computer program

Description automatically generated

**DB restoration succeeded**

**Setup of database ARSystem is done**

**Database setup completed successfully**

A screenshot of a computer screen

Description automatically generated

**Health check sandbox server:**

**Do health check for localhost:8008, the Jetty layer**

<http://localhost:8008/api/rx/application/healthcheck/ready>

A screenshot of a computer program

Description automatically generated

**JSON returned => Bundles have been loaded successfully**

**Sandbox server:**

<http://localhost:8008/helix/index.html#/com.bmc.arsys.rx.innovationstudio/login>

**Local AR server:**

<http://localhost:8008/helix/index.html#/com.bmc.arsys.rx.innovationstudio>

Default password for AR Server system = arsystem

Default password for AR Server Demo (Admin) user = P@ssw0rd

A screenshot of a login screen

Description automatically generated

**====**

**List all Docker images**

**docker images;**

A screenshot of a computer screen

Description automatically generated

**If AR server cannot start, use Docker to restart it:**

**Get into arserver docker image:**

docker exec -it arserver bash;

A screen shot of a computer

Description automatically generated

**Present working directory:**

pwd;

A screen shot of a computer

Description automatically generated

**Navigate to 'bin' folder to stop the server using:**

cd /opt/bmc/ARSystem/bin;



./arsystem stop;

A screen shot of a computer

Description automatically generated

**In another Git Bash terminal, tail all log files from DB folder (they are also local, as the folder is shared, it should be under /logs/ars/db**

**if memory serves):**

docker exec -it arserver bash;

cd /opt/bmc/ARSystem/db;

A screen shot of a computer

Description automatically generated

tail -f \*.log;

**Start the server (Docker => arserver):**

cd /opt/bmc/ARSystem/bin;

./arsystem start;

**Tones of log will come up**

A screenshot of a computer

Description automatically generated

**Do health check again for localhost:8008, the Jetty layer**

<http://localhost:8008/api/rx/application/healthcheck/ready>

A screenshot of a computer program

Description automatically generated

**JSON returned => Bundles have been loaded successfully**

**Now go checking**

<http://localhost:8008/helix/index.html#/com.bmc.arsys.rx.innovationstudio/login>

A screenshot of a login page

Description automatically generated

**If password is incorrect => ERROR (623)**

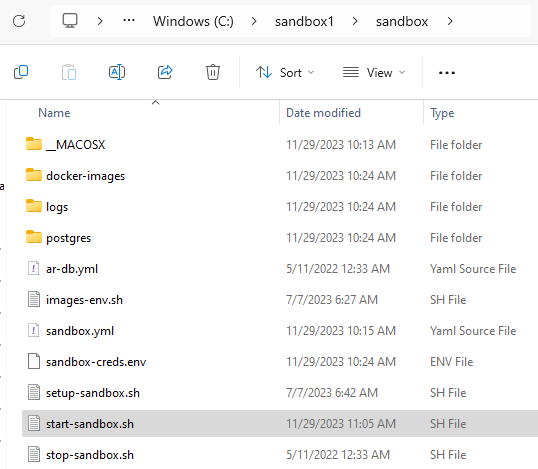
1. **Starting up AR Server**

**// Move to Sandbox**

**Git Bash:**

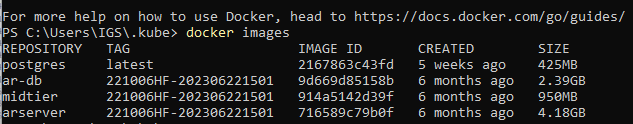
**cd /c/sandbox1/sandbox;**

**bash start-sandbox.sh;**



**List all Docker images**

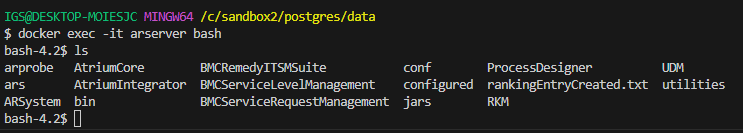
**docker images;**



**If AR server cannot start, use Docker to restart it:**

**Connect to arserver docker image:**

docker exec -it arserver bash;



**Hostname**



**Present working directory:**

pwd;

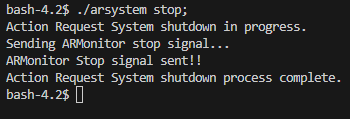


**Navigate to 'bin' folder to stop the server using:**

cd /opt/bmc/ARSystem/bin;



./arsystem stop;

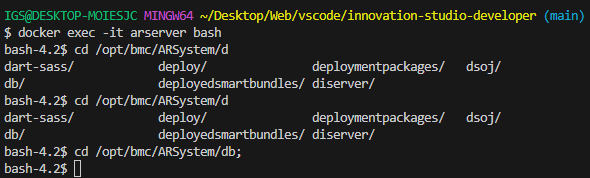


**In another Git Bash terminal, tail all log files from DB folder (they are also local, as the folder is shared, it should be under /logs/ars/db**

**if memory serves):**

docker exec -it arserver bash;

cd /opt/bmc/ARSystem/db;



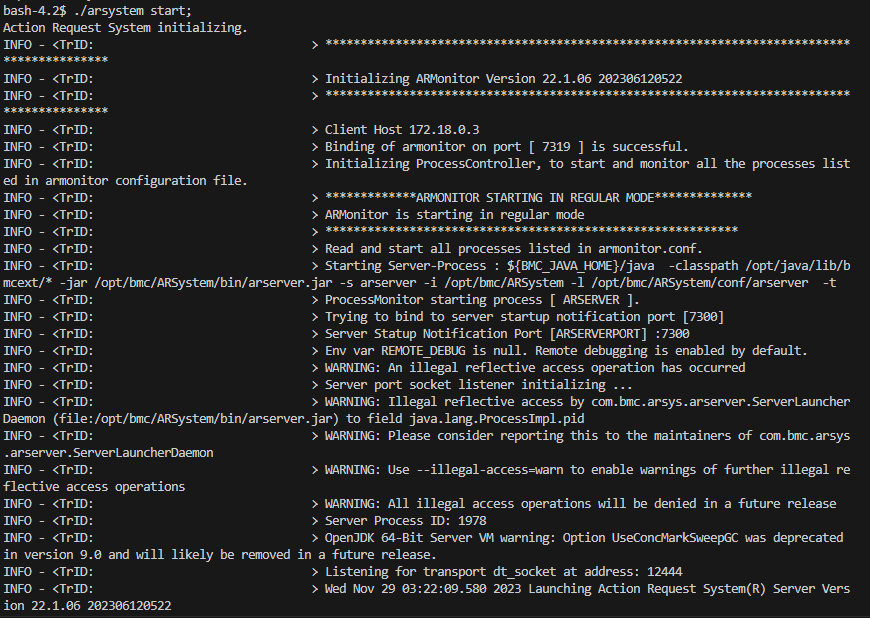
tail -f \*.log;

**Start the server (Docker => arserver):**

cd /opt/bmc/ARSystem/bin;

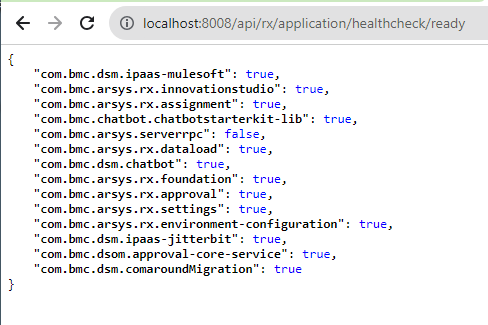
./arsystem start;

**Tones of log will come up**



**Do health check again for localhost:8008, the Jetty layer**

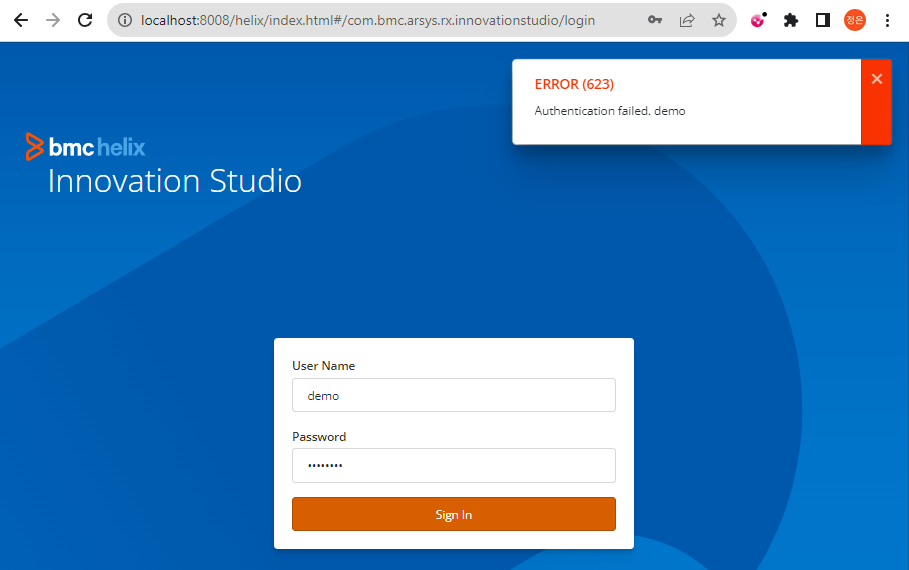
<http://localhost:8008/api/rx/application/healthcheck/ready>



**JSON returned => Bundles have been loaded successfully**

**Now go checking**

<http://localhost:8008/helix/index.html#/com.bmc.arsys.rx.innovationstudio/login>



**If password is incorrect => ERROR (623)**

1. **Starting up Mid-tier Server**

// Once Docker Desktop is running

docker exec -it midtier bash;



hostname



# BMC core path

cd /opt/bmc/bin

A black screen with white text

Description automatically generated

# Interesting core .sh file

vi /bin/mtboot.sh;

# Config files

A computer screen with white text

Description automatically generated

# config properties

/opt/apache-tomat/webapps/arsys/WEB-INF/classes/config.properties

# RSSO agent properties

/opt/apache-tomcat/webapps/arsys/WEB-INF/classes/rsso-agent.properties

# Midtier lib

/opt/apache-tomcat/webapps/arsys/WEB-INF/lib

# Tomcat env

/opt/apache-tomcat/bin/setenv.sh

# Tomcat startup

/opt/apache-tomcat/bin/startup.sh

# arsystem.ccs.host & ports

/opt/apache-tomcat/webapps/arsys/WEB-INF/classes/ccs.properties

# Change Tomcat thread settings

/opt/apache-tomcat/conf/server.xml