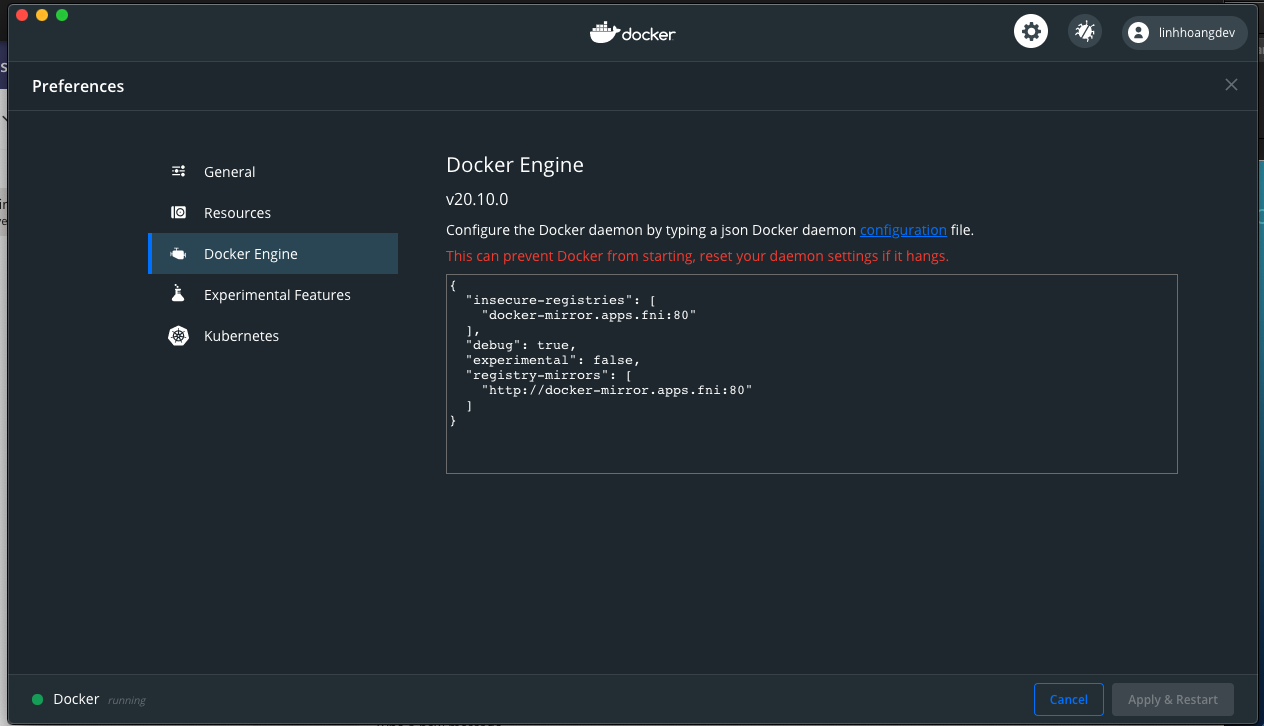
FUGU SETUP LOCAL GUIDELINE:

1. Configuration Git: (For windows: to resolve \r error):
   * git config --global core.autocrlf input
2. Create token from GitHub
   * Click setting on GitHub profile
   * Select 'Development Settings'
   * Select 'Personal access tokens'
   * Generate and save token.
   * Clone code: git clone <https://ghe.coxautoinc.com/SFNI/fni-fugu-app.git>:
     1. User name: GitHub account (Linh-HoangNguyen).
     2. Password: token.
3. Setup Artifactory:
   * Artifactory Enabling: Click the following link to get to your Artifactory profile setting: [Click here](https://artifactory.coxautoinc.com/artifactory/webapp/)
   * Under the Authentication Settings section click the gear wheel button
   * Your API Key is now generated.  Copy the API key
   * Run following command in command line: docker login [https://dtfni-docker.artifactory.coxautoinc.com](https://dtfni-docker.artifactory.coxautoinc.com/)
     1. username is your email address
     2. password is your API Key
4. Add below config in docker engine registry:



{​​​​​​​​

"insecure-registries": [

"docker-mirror.apps.fni:80"

],

"debug": true,

"experimental": false,

"registry-mirrors": [

"[http://docker-mirror.apps.fni:80](http://docker-mirror.apps.fni/)"

]

}​​​​​​​​

1. How to bring up the fugu project in local?

To build the project in local machine

docker-compose build

To bring up the project with local single node mongo

***docker-compose up***

To bring up the project with local replicaset mongo

docker-compose -f docker-compose-mongo-replicaset.yaml up

To bring up the project with devint mongo

docker-compose -f docker-compose-devint.yaml up

1. Test postman API:

Add “***Fugu Event Save.postman\_collection.json”*** to Postman and test call:

Graphical user interface, text, application, email, Teams

Description automatically generated

# Setup Fugu admin at local (using docker-compose-local.yaml)

## Clone [Fugu admin](https://ghe.coxautoinc.com/SFNI/fni-fugu-admin) source code

## Turn off health check condition for mongo services (workaround)

### Lines 32->37 in docker-compose-local.yaml

depends\_on:  
*# mongo1:  
# condition: service\_healthy  
# mongo2:  
# condition: service\_healthy  
# mongo3:  
# condition: service\_healthy* vault:  
 condition: service\_healthy  
 fni-fugu-app:  
 condition: service\_healthy

## Build docker image

$ docker-compose -f docker-compose-local.yaml build

## Turn on mongo services

$ docker-compose -f docker-compose-local.yaml up mongo1 mongo2 mongo3

### Insert default Users into mongo (do this only at 1st time), open another shell, then:

$ docker exec -it localmongo1 /bin/bash

$ mongo

$ rs.initiate({\_id : 'rs0',members: [{ \_id : 0, host : "mongo1:27017" },{ \_id : 1, host : "mongo2:27017" },{ \_id : 2, host : "mongo3:27017" }]})

$ use admin

$ db.createUser({user: "rsroahfcuser", pwd: "rsroahfcpass", roles: [{ role: "readAnyDatabase", db: "admin"}]});

$ db.createUser({user: "rsrodtcomuser", pwd: "rsrodtcompass", roles: [{ role: "readAnyDatabase", db: "admin"}]});

$ db.createUser({user: "rsrwahfcuser", pwd: "rsrwahfcpass", roles: [{ role: "readWriteAnyDatabase", db: "admin"}]});

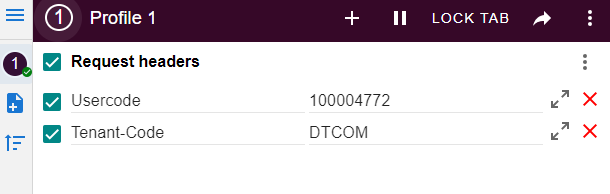
$ db.createUser({user: "rsrwdtcomuser", pwd: "rsrwdtcompass", roles: [{ role: "readWriteAnyDatabase", db: "admin"}]});

## Turn on admin service

open another shell, then

## $ docker-compose -f docker-compose-local.yaml run --service-ports admin Install ModHeader

<https://chrome.google.com/webstore/detail/modheader/idgpnmonknjnojddfkpgkljpfnnfcklj>

then setting: 

## Access to Fugu Admin through <http://localhost:8076/>

# Setup Fugu admin at local (just for tester) (using docker-compose-devint.yaml)

Tại file fugu\_admin/settings.py thay đổi value từ:

else:  
 MULE\_HOST = 'dt2esbint'  
 FUGU\_TENANT\_PORTS = {  
 'AHFC': '7000',  
 'DTCOM': '7100'  
 }

thành:

else:  
 MULE\_HOST = 'dt2esbint.devint1.qts.fni'  
 FUGU\_TENANT\_PORTS = {  
 'AHFC': '7000',  
 'DTCOM': '7100'  
 }

Build docker image and Run:

$ docker-compose -f docker-compose-devint.yaml build

$ docker-compose -f docker-compose-devint.yaml up