FM SETUP INSTRUCTIONS

Index

FM Installation

FM installation prerequisites

1. Install docker(https://docs.docker.com/engine/install/)
2. Install docker-compose(https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-compose-on-ubuntu-20-04)

Setup FM with push_fm script

1. Clone fleet manager repository and setup git config for submodule

```
git clone https://github.com/AtiMotors/fleet_manager
cd fleet_manager
git pull
git submodule init

# open and edit .git/config file, add branch=dev to submodule mule
entry
[submodule "mule"]
url = https://<token>@github.com/AtiMotors/mule.git
active = true
branch = dev
```

git submodule update

2. Checkout to release/branch, update mule submodule.

```
git checkout <branch>
git submodule update --remote [Optional]
git submodule update
```

3. Setup cert files - You will need python installed in your machine to carry out this step

   3.1 *Update all_server_ips, http_scheme in static/fleet_config/fleet_config.toml, add wireguard ip of the server as well if wg access is present*

```
for example:
    all_server_ips=["192.168.6.11", "10.9.0.168", "127.0.0.1"]
    http_scheme="https"
```

   3.2 *Install toml, crytography in your machine(not server)- these packages are required to generate cert files*

```
pip install toml cryptography
```

   3.3 *Upon successful installation of the above mentioned packages, run setup_certs.py to generate FM cert file*

```
cd utils && python3 setup_certs.py
../static/fleet_config/fleet_config.toml ../static
```

   3.4 *Copy the cert file(static/certs/fm_rev_proxy_cert.pem) to all the sherpas(/opt/ati/config/fm_rev_proxy_cert.pem)*

4. Update static directory with map_files

   4.1. *Create map folders for each of the fleets*

```
mkdir static/fleet_1/map/
copy all the map files of fleet_1 to static/fleet_1/map/

mkdir static/fleet_2/map/
copy all the map files of fleet_2 to static/fleet_2/map/
```

5. Check all the options available in push_fm script, use them according to your requirements

```
Program to push fleet_manager repo to the FM server!
Args: [-i/W|c|h|v]
options:
i     Give IP address of the FM server, default is localhost
W     Copies the contents of static folder on local machine directly to the
FM server, else the static folder on server will be retained
c     Checksout the local directory static to its current git commit after
the push is successful
b     WILL NOT build the base image
v     Will run docker as host, useful if fm has to communicate with master
fm via vpn connection
h     Display help
```

6. If server has internet, allows you to download open-source packages (Recommended to use step 6 instead of this step)

   a. If you want to setup fm on a remote location, run push_fm script to create all the docker images on the server

   ```
   ./scripts/push_fm.sh -Wi username@ip
   ```

   b. If you want to setup fm on your machine, run push_fm script to create all the docker images on your machine

   ```
   ./scripts/push_fm.sh -W
   ```

7. If server doesn't have internet access, copy built docker images to the server from Ati server(data@192.168.10.21:/atidata/datasets/FM_v<fm_version>_docker_images), run the following commands

   a. Load base images on server/localhost

   ```
   ssh username@ip
   cd FM_v<fm_version>_docker_images
   bash load_docker_images.sh
   exit
   ```

   b. If you want to setup fm on a remote location, run push_fm script from your machine to create all the docker images on the server

   ```
   ./scripts/push_fm.sh -Wbi username@ip
   ```

c. If you want to setup fm on your machine, run push_fm script from your machine to create all the docker images on your machine

```
./scripts/push_fm.sh -Wb
```

8. Setup plugins if any.
9. Setup sherpas.
10. Setup optimal dispatch config
11. Push mule docker image to local docker registry
12. To start using fleet_manager, follow Start or Restart FM

Setup FM by copying built docker images

1. Copy built docker images to the FM server from Ati server(data@192.168.10.21:/atidata/datasets/FM_v<fm_version>_docker_images)
2. Load docker images

```
cd FM_v<fm_version>_docker_images
bash load_docker_images.sh
```

3. Backup the current fleet_config directory present in the FM server. Copy the updated fleet_config directory from FM_v<fm_version>_docker_images folder to the FM server static dir, update it with the info from fleet_config backup . With updates, config parameters change, redoing config will help. Static dir should contain updated fleet_config, mule_config, certs, all the required map_folders etc.
4. Copy docker_compose_host.yml,docker_compose_bridge.yml from <fm_repository>/misc/ or FM_v<fm_version>_docker_images folder to the static folder.
5. Create cert files if not already present by following Setup FM with push_fm script steps 1-3.
6. Copy the cert files generated (fm_rev_proxy_cert.pem, fm_rev_proxy_key.pem) to the static/certs/ directory in the FM server
7. Follow steps 7-10 in Setup FM with push_fm script
8. To start using fleet_manager, follow Start or Restart FM

Start or Restart FM

1. Modify timezone if required by setting environment variables TZ, PGTZ in services fleet_manager, db enlisted in static/<docker_compose_file>, docker_compose_file can be docker_compose_host.yml(to access master_fm via VPN) or docker_compose_bridge.yml based on the conf you choose

2. If docker is going to be run in host network mode, modify below mentioned parameters in static/grafana_config/datasources/default.yml

```
url: localhost:5432
```

3. Start FM

```
cd static
docker-compose -p fm -f <docker_compose_file> down
docker-compose -p fm -f <docker_compose_file> up
```

4. Use FM through UI, if running FM on localhost use ip as 127.0.0.1

```
https://<ip>/fm/
username: <username>
password: <password>
```

5. Addition/Deletion of fleets, sherpas should be done through Configure page on the dashboard. Adding it to fleet_config.toml will have no effect. Fleets names have to be same as map_names. Copy the map files to the static directory on FM server by following Setup FM with push_fm script step 4 before trying to add it through dashboard.

6. Please restart FM using restart_fleet_manager button on the maintenance page, after adding sherpas/fleets.

7. Induct all the sherpas that you want to use
   a. Press enable for trips button from sherpa card
   b. Only those sherpas that has been enabled for trips will get assigned with a trip

8. Follow Fleet maintenance if needs be

Run FM Simulator

a. Follow Setup FM with push_fm script , steps 1-2

b. Set simulate in static/fleet_config/fleet_config.toml

```
[fleet.simulator]
simulate=true
```

c. To get trip bookings done automatically add routes(list of station names), trip booking frequency(seconds) to fleet_config. route1 will be a scheduled trip, route2 would be booked as a normal one time trip

```
[fleet.simulator.routes]
route1 = [["Station A", "Station B"], ["10", "2023-05-31 15:00:00", "2023-
05-31 16:00:00"]]
route2 = [["Station B", "Station A"], ["-1", "", ""]]
```

d. Make sure all the stations mentioned in gmaj file(<fleet_name>/map/grid_map_attributes.json) has only the below mentioned tags. Tags like conveyor, auto_hitch, auto_unhitch will not work in simulator mode.

```
 "station_tags": [
  "parking",
  "dispatch_not_reqd"
 ]
```

e. If you want to start sherpas at particular station add this patch to config

```
[fleet.simulator.initialize_sherpas_at]
sample_sherpa="Station A"
```

f. If you want to simulate transit visas set visa handling in fleet.simulator config

```
[fleet.simulator]
visa_handling=true
```

g.Follow remaining steps in Setup FM with push_fm script, steps 3-7. Setup Sherpa not required for simulation

Setup sherpas

a. Copy fm cert file(fm_rev_proxy_cert.pem) generated in Setup FM with push_fm script step 3 to sherpa's /opt/ati/config directory

b. Add this patch to /opt/ati/config/config.toml in the mule

```
[fleet]
api_key = " "
chassis_number = " "
data_url = "https://<fm_ip_address>:443/api/static"
http_url = "https://<fm_ip_address>:443"
```

```
ws_url = "wss://<fm_ip_address>:443/ws/api/v1/sherpa/"
fm_cert_file="/app/config/fm_rev_proxy_cert.pem"
```

Setup Plugin

a. Setup IES

b. Summon button, conveyor plugins can be configured through UI (maintenance/Plugin Editor). Add the required plugins to static/fleet_config/plugin_config.toml. Restart would be required after you add a new plugin,post restart you would see a plugin editor page in UI.

```
all_plugins=["summon_button", "conveyor"]
```

Setup IES

a. Add IES plugin to static/fleet_config/plugin_config.toml

```
all_plugins=["ies"]
```

b. Modify static/plugin_ies/locationID_station_mapping.json file. Map IES station names to corresponding ati station names as the template indicates.

```
{
    "Warehouse_Pick": "ECFA start",
    "HP02_FA02": "ECFA-2",
    "HP03_FA01": "ECFA-1",
}
```

Setup optimal dispatch config

Optimal dispatch logic tries to allocate the pending trips with the best sherpa available. Choice of best sherpa is made with the paramter Z

$Z=(eta)^a/(priority)^b$
$priority=p1/p2$

```
where,
    eta - expected time of arrival computed for the sherpa to reach the
first station of the trip booked,
    priority - measure of how long a trip has been pending,
    p1 - Time since booking of currrent trip,
    p2 - Minimum of time since booking across all the pending trips,
    a - eta power factor , 0<a<1,
    b - priority power factor , 0<b<1,
```

1. **Maximise number of trips done**: To get maximum number of trips done in a given time frame eta_power_factor can be set to 1, priority_power_factor can be set to 0. This will make the optimal disaptch logic to lean towards trips that can be started faster. The trip booking order will not be followed.

```
[optimal_dispatch]
method="hungarian"
prioritise_waiting_stations=true
eta_power_factor=1.0
priority_power_factor=0.0
```

2. **Fair scheduling**: To configure optimal dispatch logic to take trips in the order they were booked eta_power_factor can be set to 0, priority_power_factor can be set to 1.

```
[optimal_dispatch]
method="hungarian"
prioritise_waiting_stations=true
eta_power_factor=0.0
priority_power_factor=1.0
```

3. **Custom configuration**: There is no ideal combination of eta_power_factor, priority_power_factor. They should be choosen according to the frequecy of trip bookings, route length between the stations to maximise the throughtput.

4. For good takt time, eta power factor should be higher, for fair scheduling priority power factor should be set higher.

5. Sherpas can also be restricted from running on certain routes/station by setting up exclude_stations. Check saved route feature.

6. To reduce computation load due to optimal dispatch,
max_trips_to_consider can be lowered. For a standalone/FM on sherpa,
max_trips_to_consider can be set to a value less than 5 depending on the
use case. Optimal dispatch logic will be run only for the first
max_trips_to_consider number of trips. Default is set to 15.
```markdown
[optimal_dispatch]
max_trips_to_consider=15
```

Push mule docker image to local docker registry

1. Copy mule docker image tar file to fm_server and load the image

```
docker load -i <mule_image tar file>
```

2. Tag mule image with registry ip, tag on fm server

```
docker tag mule:<mule_tag> <fm_ip>:443/mule:fm
```

3. Setup certs for docker push on fm server

```
sudo mkdir /etc/docker/certs.d/<fm_ip>:443
sudo cp <fm_static_dir>/certs/fm_rev_proxy_cert.pem
/etc/docker/certs.d/<fm_ip>:443/domain.crt
```

4. Push mule docker image to FM local registry

```
# auth has been added to docker registry
docker login -u ati_sherpa -p atiCode112  <fm_ip>:443
docker push <fm_ip>:443/mule:fm
```

Fleet maintenance

Update map files

1. Copy all the new map files to <fm_static_directory>/<fleet_name>/all_maps/<map_version_name>/ folder
2. Select the fleet which needs the map update from the webpage header in the dashboard and press update_map button on the webpage header(present along with start/stop fleet , emergency_stop fleet etc.), and choose map_version from drop down
3. Restart of FM after update map button is pressed. FM Pop up would ask for restart.

Swap sherpas between fleets

1. Delete the current sherpa entry and then again add it to a different fleet. Sherpa's can't be swapped directly. FM restart would be required post addition of sherpa to the new fleet.

Generate api keys for sherpas/conveyor/summon_button/any hardware

1. Run utils/api_key_gen.py in utils directory in fleet_manager - You will need fleet_manager repository access, python installed in your machine to run this. Python dependecies required: secrets, click

```
cd <path_to_fleet_manager_repository>/utils
python3 api_key_gen.py --hw_id <unique_hwid>
```

2. To generate api keys for n devices like summon_button

```
cd <path_to_fleet_manager_repository>/utils
python3 utils/gen_api_keys_n_devices.py --num_devices 10
```

Add/Remove frontendusers

1. Run utils/gen_hashed_password.py in utils directory in fleet_manager - You will need fleet_manager repository access, python installed in your machine to run this. Python dependecies required: hashlib, click

```
cd <path_to_fleet_manager_repository>/utils
python3 utils/gen_hashed_password.py --password <password>
```

2. The generated hasshed password can be added to <fm_static_directory>/fleet_config/frontend_users.toml

```
[frontenduser.<new_user>]
hashed_password=<hashed password>
```

3. Remove unwanted entries from <fm_static_directory>/fleet_config/frontend_users.toml if any, restart FM for the changes to take effect.

4. Default FM login credentials

```
username: ati_support
password: atiSupport112
role: support

username: admin
password: 1234
role: support

username: supervisor
password: ati1234
role: supervisor
```

```
username: operator
password: 1234
role: operator
```

Flash summon button firmware

a. Connect summon button to your laptop via USB to flash firmware

b. Copy FlashTool_v2.3.6 from data@192.168.10.21:/atidata/datasets/FM_v<fm_version>_docker_images> to your laptop, run the same.

```
cd FlashTool_v2.3.6
sudo bash ./install.sh
sudo bash ./flashtool_8mb.sh
```

c. Upon flashing, reconnect the summon button usb.

d. Generate unique api key for summon button by using following generate api keys section in Fleet Maintenance

e. Press and hold the button until LED turns blue, connect to summon button via wifi. For instance you would see something like Summon_192049 in the available/known wifi networks. Upon successful connection to summon button wifi, you will see a summon button UI.

f. Press configure WiFi, choose the preferred network and add the wifi password for the same, save it. Wait unitl summon button led turns from yellow to blinking red .

g. Repeat step e, connect to summon button network

h. Now press configure device, add FM plugin url to HOST. PLUGIN_PORT by default would be 8002

```
ws://<FM_IP>:<PLUGIN_PORT>/plugin/ws/api/v1/summon_button
```

i. Set wifi type: WPA/WPA2

j. Set Mode to WiFi-Only

k. Set HEARTBEAT to disable

l. Set APIKEY using api key generated in step d , save.

```
X-API-Key:<api_key>
```

m. Press restart device in summon button UI.

Use Saved routes

1. Enable battery swap trips:

a. Set up conditional trip config

```
[conditional_trips]
trip_types = ["battery_swap"]

[conditional_trips.battery_swap]
book=true
max_trips = 2 # max number of sherpas that can be sent for battery swap at
the same time
threshold = 100 # battery level
priority = 10  # trip priority
```

b. Go to route ops(maintenance) page, select the route you want the sherpa to do when battery level is below threshold, press save and tag it as battery_swap route.

2. Enable idling trips:

a. Set up conditional trip config

```
[conditional_trips]
trip_types = ["idling_sherpa"]

[conditional_trips.idling_sherpa]
book=true
max_trips = 2 # max number of sherpas that can be booked with trips when
found idling at the same time
threshold = 100 # battery level
priority = 1  # trip priority
```

b. Go to route ops(maintenance) page, select the route you want the sherpa is found idling beyond threshold seconds, press save, select sherpa from dropdown and tag it as parking route.

3. Disable sherpa from going to a list of stations

a. Go to route ops(maintenance) page, select the stations that sherpa shouldn't go to, press save, select sherpa from dropdown and tag it as exclude_stations route.

Setup master FM comms

1. Generate api key for the FM server

```
cd <path_to_fleet_manager_repository>/utils
python3 api_key_gen.py --hw_id <customer_name>
```

2. Add customer to master_fm database

```
1.Login to sanjaya.atimotors.com
2.Use add client functionality in client configuration page (requires
customer name, api key generated in the previous step)
```

3. If the FM server has direct access to sanjaya.atimotors.com then make sure mfm_ip, port, cert_files are set as given below in static/fleet_config/master_fm_config.toml

```
mfm_ip="sanjaya.atimotors.com"
mfm_port="443"
mfm_cert_file="/etc/ssl/certs/ca-certificates.crt"
http_scheme="https"
ws_scheme="wss"
```

4. If the FM server doesn't have direct access to sanjaya.atimotors.com but the FM server can be accessed via ssh then set mfm_ip, port, schemes are set as given below in static/fleet_config/master_fm_config.toml. We will have to setup reverse tunnel to sanjaya.atimotors.com

```
mfm_ip="127.0.0.1"
mfm_port="9010"
mfm_cert_file="/etc/ssl/certs/ca-certificates.crt"
http_scheme="http"
ws_scheme="ws"
```

5. To setup reverse tunnel, copy the folder mfm_rev_tunnel from FM_v<fm_version>_docker_images to the machine which has access sanjaya.atimotors.com(pingable) and has ssh access to the FM server.

```
cd mfm_rev_tunnel
bash mfm_rev_tunnel.sh
```

4. Edit params in static/fleet_config/master_fm_config.toml in the FM server and restart the same

```
[master_fm.comms]
send_updates=true
ws_update_freq=60
```

```
update_freq=120
api_key=<api_key>
```

Debug FM

1. Check if there were any queue build ups. The output would show queue build ups if any.

```
docker exec -it fleet_manager bash
inspect
rqi
```

2. Check for occurences of rq errors (rqe) in fleet_manager.log, the output might lead to the issue

```
rqe
```

3. If you are unable to login to FM, Check the docker logs- this should be run outside docker. There might be some errors in the init scripts.

```
docker logs fleet_manager
docker logs fleet_db
```