MFP Management Application server set up

Table of Contents:

Server Details :

IP : 10.188.101.118

User Name: test

Password: tsip123!@#

App URL : http://10.188.101.118:50777/client/index.html

Pre-Requisite:

Linux System is required to set up the server

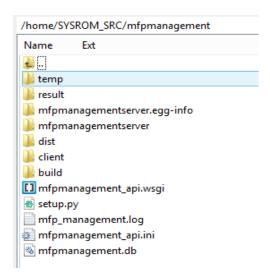## Initial Set up:

Libraries to be installed :

- Install latest python version(version 3.5 and above) s and its dependencies
- Install pyramid, cornice,
- Install subprocess,
- Install inspect
- Install threading, shutil
- Install setuptools
- Install paramiko
- Install pysnmp
- Install zipfile
- Install elementpath
- Install sqlite3
- Install time, sys, os, json, logging

Install its dependent libraries with versions too if necessary

Folder structure:

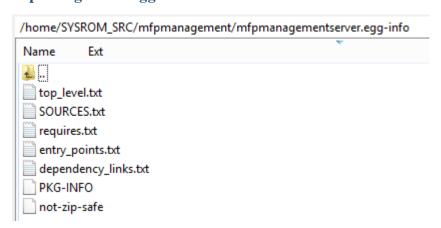Create a folder mfpmanagement and place it in /home/SYSROM_SRC directory

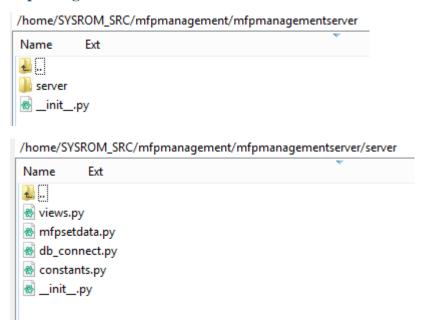Inside mfpmanagement folder place the following files and folders

After installing all the libraries and placing the above shown folders on mfpmanagement directory go to /home/ SYSROM_SRC/mfpmanagement and run "python3 setup.py install" in terminal

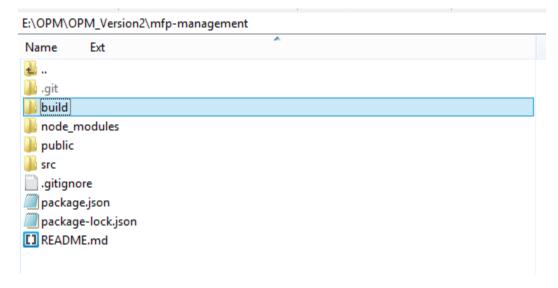**Files and folder view within each folder :**

mfpmanagement.egg-info :

mfpmanagementserver :

/home/SYSROM_SRC/mfpmanagement/mfpmanagementserver

Name	Ext
⬆ ..	
📁 server	
🐍 __init__	.py

/home/SYSROM_SRC/mfpmanagement/mfpmanagementserver/server

Name	Ext
⬆ ..	
🐍 views	.py
🐍 mfpsetdata	.py
🐍 db_connect	.py
🐍 constants	.py
🐍 __init__	.py
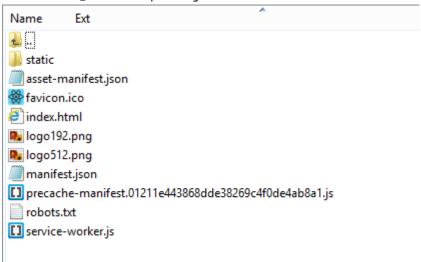
Client :

- Create a build of the react application when changes are made to the application
- Build deployed client files will be present at "build" folder of the application

E:\OPM\OPM_Version2\mfp-management

Name	Ext
⬆ ..	
📁 .git	
📁 build	
📁 node_modules	
📁 public	
📁 src	
📄 .gitignore	
📄 package.json	
📄 package-lock.json	
[] README.md	

- View of client side "build" folder

E:\OPM\OPM_Version2\mfp-management\build

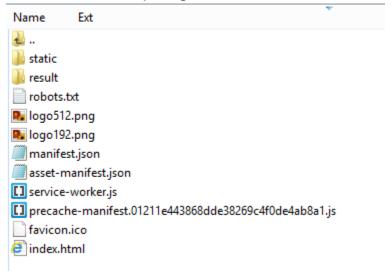| Name	Ext
🔼 ..	
📁 static	
📄 asset-manifest.json	
⚛ favicon.ico	
🅔 index.html	
🖼 logo192.png	
🖼 logo512.png	
📄 manifest.json	
[] precache-manifest.01211e443868dde38269c4f0de4ab8a1.js	
📄 robots.txt	
[] service-worker.js	

- Move all the files present in "build" directory of the react application to the "client" directory in the server
- NOTE : Create a build and deploy in the server only when changes are made to the application else use the existing build files

Client build deployment :

- Steps to create build of the client application (Note: These steps have to be followed by front-end application developers ONLY when changes are made to client side of the application)
  - Go to the application folder and open the terminal (preferably work with visual studio code IDE)
  - Run "npm install" command (Note: this command should be executed only once during the start of the project for installing code dependent libraries if **node_modules** folder is NOT present in the application , this is similar to installing libraries in the system)
  - Run "npm start" to start the server
  - Run "npm run build" command to create build
  - Resolve dependencies if any by installing the packages through terminal
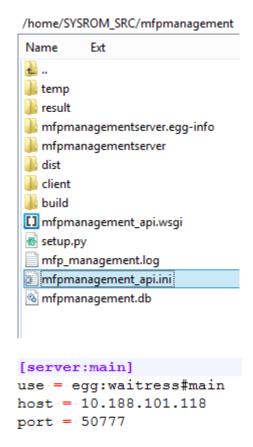
Client folder at server:

/home/SYSROM_SRC/mfpmanagement/client

Name	Ext
..	
static	
result	
robots.txt	
logo512.png	
logo192.png	
manifest.json	
asset-manifest.json	
service-worker.js	
precache-manifest.01211e443868dde38269c4f0de4ab8a1.js	
favicon.ico	
index.html	

- Note : Create a additional "**result**" folder in the "client" folder.

Code changes required while setting up in new machine:
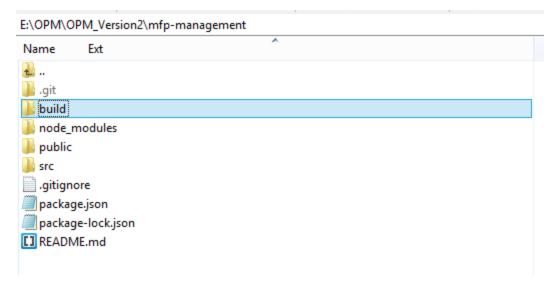
- **Server code changes:**
- At **mfpmanagement_api.ini** file change the host address to server machine's IP address and port number to required port number.

```
/home/SYSROM_SRC/mfpmanagement

Name          Ext
..
temp
result
mfpmanagementserver.egg-info
mfpmanagementserver
dist
client
build
mfpmanagement_api.wsgi
setup.py
mfp_management.log
mfpmanagement_api.ini
mfpmanagement.db
```

```
[server:main]
use = egg:waitress#main
host = 10.188.101.118
port = 50777
```
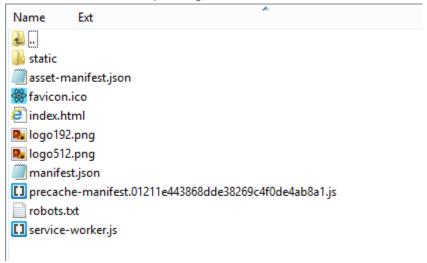
Client code changes:

At client code:

- Search for 50777 (i.e., port number set) in the application files and make changes with the latest server ip address and port number while making api calls
- Save the changes
- Run "npm install" command (Note: this command should be executed only once during the start of the project for installing code dependent libraries if **node_modules** folder is not present in the application, this is similar to installing libraries in the system)
- Run "npm start" command
- Run "npm run build" command
- Build deployed client files will be present at "build" folder of the application

E:\OPM\OPM_Version2\mfp-management

Name	Ext
..	
.git	
build	
node_modules	
public	
src	
.gitignore	
package.json	
package-lock.json	
README.md	

View of client side "build" folder

E:\OPM\OPM_Version2\mfp-management\build

Name	Ext
..	
static	
asset-manifest.json	
favicon.ico	
index.html	
logo192.png	
logo512.png	
manifest.json	
precache-manifest.01211e443868dde38269c4f0de4ab8a1.js	
robots.txt	
service-worker.js	

- Move all the files present in "build" directory of the react application to the "client" directory in the server

Server Restart:

- When any changes are made to the server files, restart the server
- When the below given error message is shown in the browser restart the server

This site can't be reached

10.188.101.118 refused to connect.

Try:

- Checking the connection
- Checking the proxy and the firewall

ERR_CONNECTION_REFUSED

Reload   Details

- Command to restart the server :
    - Login to putty 10.188.101.118
      Username : test
      Password: tsip123!@#
    - Go to /home/SYSROM_SRC/mfpmanagement directory and run
      `pserve mfpmanagement_api.ini`
    - When server is started :

```
test@test-OptiPlex-790:/home/SYSROM_SRC$ cd mfpmanagement/
test@test-OptiPlex-790:/home/SYSROM_SRC/mfpmanagement$ pserve mfpmanagement_api.ini
Starting server in PID 15930.
Serving on http://BhavyaM.TOSHIBA-TSIP.COM:50777
```

PROXY ERROR:

For any proxy related error run the given commands:

```
export http_proxy=http://proxy:8080
export https_proxy=http://proxy:8080
export ftp_proxy=http://proxy:8080
```