

# **Diagnostic Code Management Application server set up**

## **Table of Contents:**

- 1. Server Details**
- 2. Pre-requisite**
- 3. Initial Set up**
- 4. Folder Structure**
- 5. Client build deployment**
- 6. Code changes required while setting up in new machine**
- 7. Server Restart**
- 8. DB Update**
- 9. Proxy Error**

## Server Details :

IP : 10.188.101.118

User Name: test

Password: tsip123!@#

App URL : <http://10.188.101.118:50666/client/index.html>

## Pre-Requisite:

Linux System is required to set up the server

## Initial Set up:

Libraries to be installed :

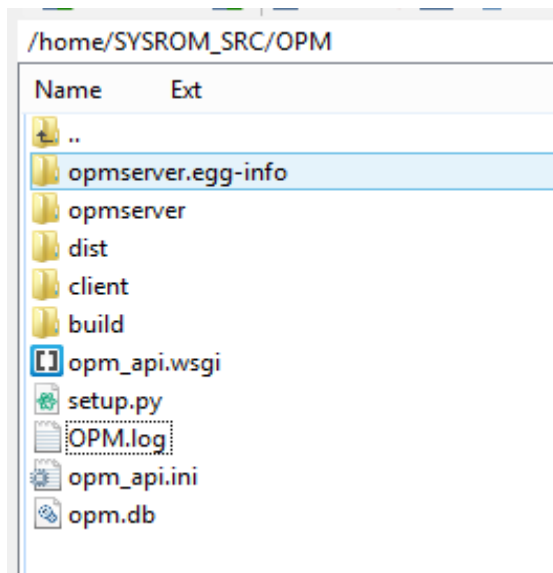
- Install latest python version (version 3.5 and above) and its dependencies
- Install pyramid, cornice, subprocess
- Install setuptools
- Install paramiko
- Install sqlite3
- Install time, sys, os, json, logging, inspect

Install its dependent libraries with versions too if necessary

## Folder structure:

Create a folder OPM and place it in /home/ SYSROM\_SRC directory

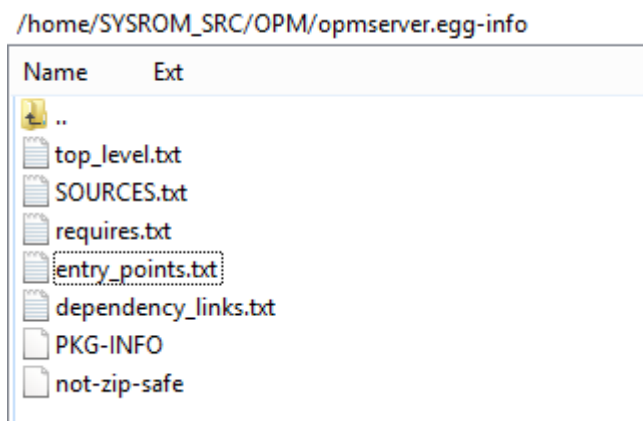
Inside OPM folder place the following files and folders



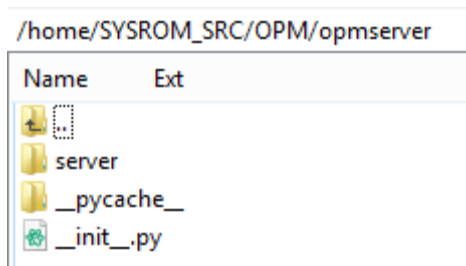
After installing all the libraries and placing the above shown folders on OPM directory go to `/home/ SYSROM_SRC/OPM` and run “python3 setup.py install” in terminal

### Files and folder view within each folder :

#### Opmserver.egg-info :



#### Opmserver :



/home/SYSROM\_SRC/OPM/opmserver/server

Name	Ext
..	
__pycache__	
views.py	
mfpconnect.py	
db_connect.py	
__init__.py	

### Client :

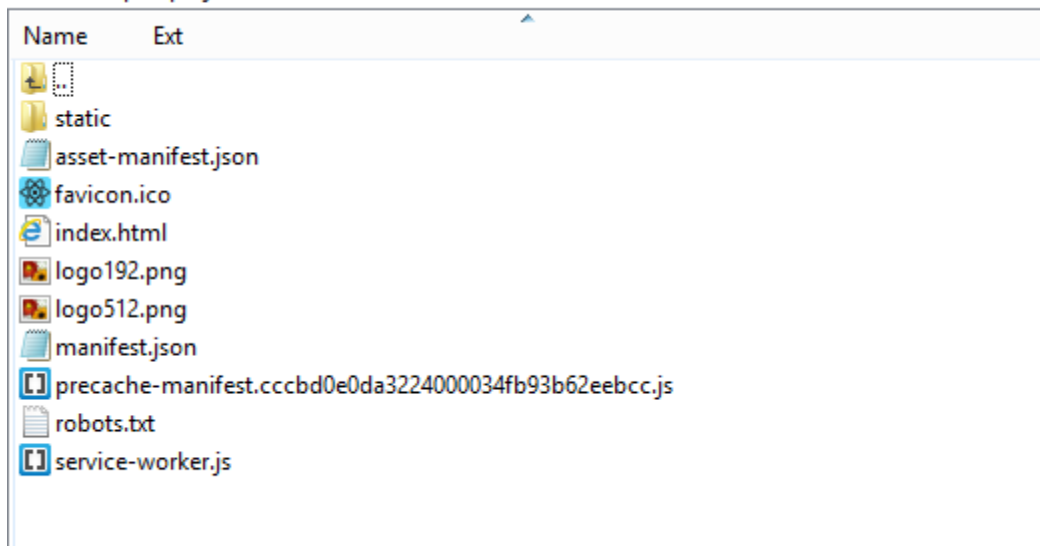
- Create a build of the react application when changes are made to the application
- Build deployed client files will be present at “build” folder of the application

E:\OPM\opm-project

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

- View of client side “build” folder

E:\OPM\opm-project\build

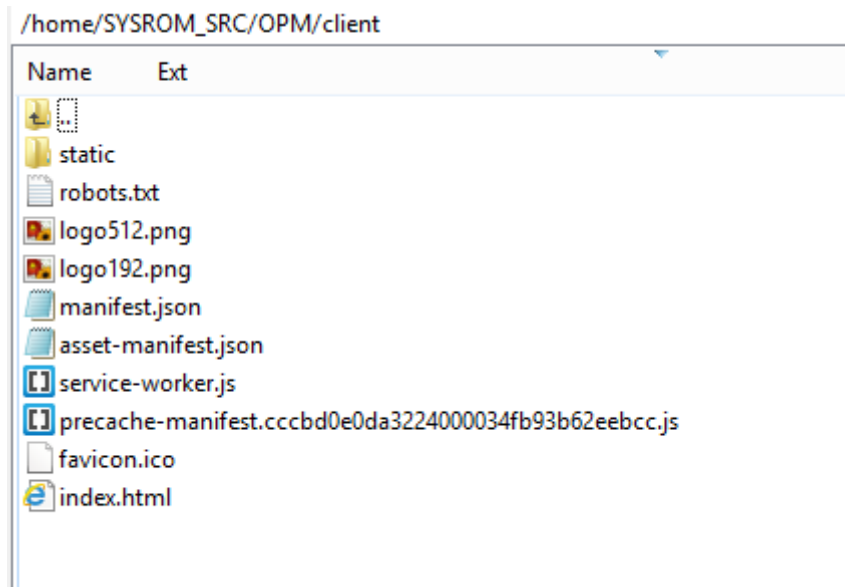


- Move all the files present in “build” directory of the react application to the “client” directory in the server
- NOTE : Create a build and deploy in the server only when changes are made to the application else use the existing build files

## 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

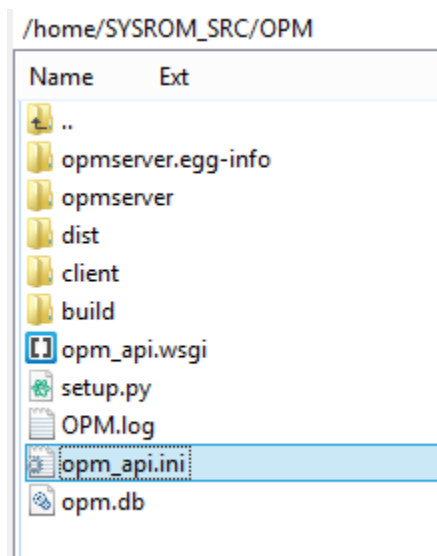
## Client folder at server:



## Code changes required while setting up in new machine :

### Server code changes:

At **opm\_api.ini** file change the host address to server machine's IP address and port number to required port number.

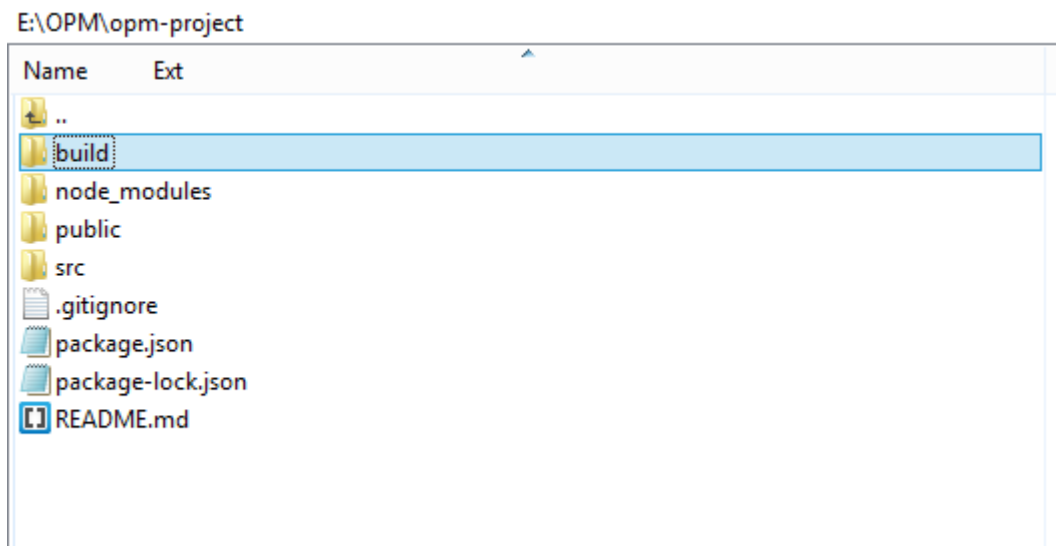


```
[server:main]
use = egg:waitress#main
host = 10.188.101.118
port = 50666
```

### Client code changes:












At client code:

- Search for 50666(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



View of client side “build” folder

E:\OPM\opm-project\build

Name	Ext
	
 static	
 asset-manifest.json	
 favicon.ico	
 index.html	
 logo192.png	
 logo512.png	
 manifest.json	
 precache-manifest.cccbd0e0da3224000034fb93b62eebcc.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/OPM directory and run `pserve opm_api.ini`

- When server is started :

```
test@test-OptiPlex-790:/home/SYSROM_SRC/OPM$ pserve opm_api.ini
Starting server in PID 8780.
Serving on http://BhavyaM.TOSHIBA-TSIP.COM:50666
```

## DB Update:

- Download DB Browser for sqlite application from <https://sqlitebrowser.org>
- Drag and drop the existing db file/open existing DB file from the app
- OPM application schema :

```
CREATE TABLE Sample_08_Code( Mode TEXT, Code INTEGER, Subcode TEXT, Possible_Value TEXT, Description TEXT, Identifier TEXT, PRIMARY KEY(Identifier));
```

- Refer <https://www.w3schools.com/sql/> to learn more about SQL Commands

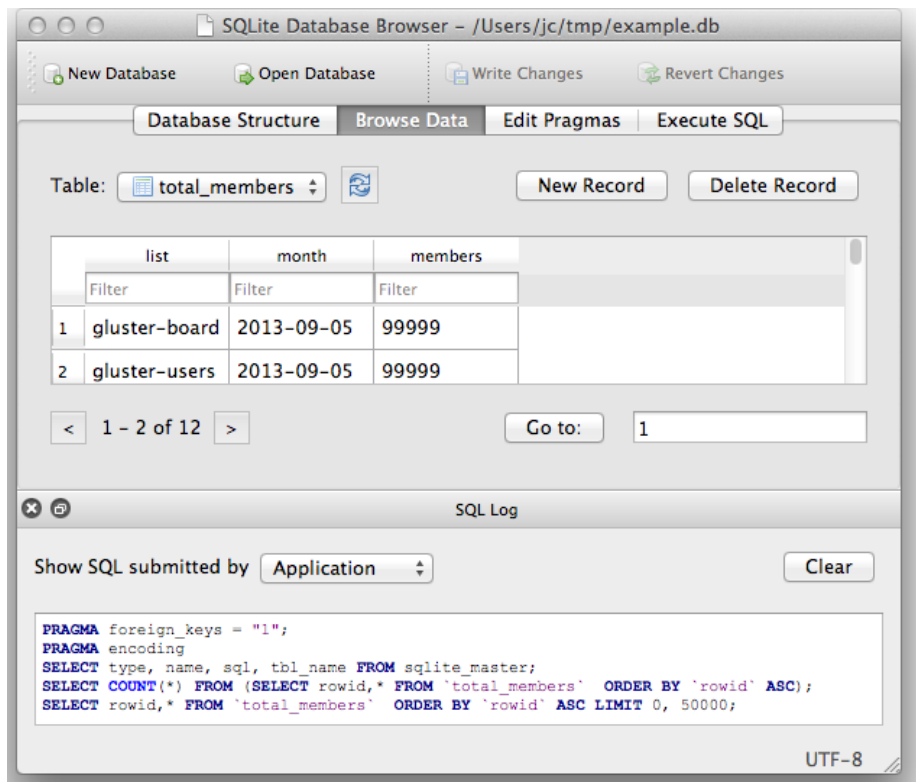
- To add additional rows example :

```
INSERT INTO
```

```
"Sample_08_Code"("Mode","Code","Subcode","Possible_Value","Description","Identifier")
VALUES (NULL,NULL,NULL,NULL,NULL,NULL);
```

- To update existing DB rows example:

```
UPDATE "Sample_08_Code" SET Mode=09;
```



- In DB Browser for sqlite application, click on Browse Data tab to view the table
- Click on Execute SQL tab and execute any of the DB commands, check the changes in the table by clicking on Browse Data tab

## 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
```