Setup Instructions 20200909_130248

iabic :-) SDNScottie : cc02: ClientServerMgmt_App 20200908_110904
, CodeChallenges 20200827_112125

{{c1::A}} KW 37

#================
Setup :-)
#================

In our python program :

dbsqlite__clientservermgmt_app__tbl__ciud__m.py

Run (Step 1) to create the database.

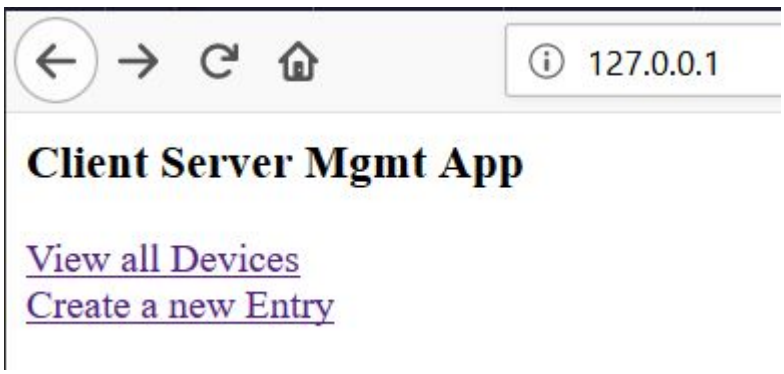
```
# ============================================================
# == Main
# ============================================================
def main():
    result = cSQLite_ClientServerMgmt_App__ciud("")

    #====================
    #==Step 1:
    #====================
    result.create_table__ClientServerMgmt()
```

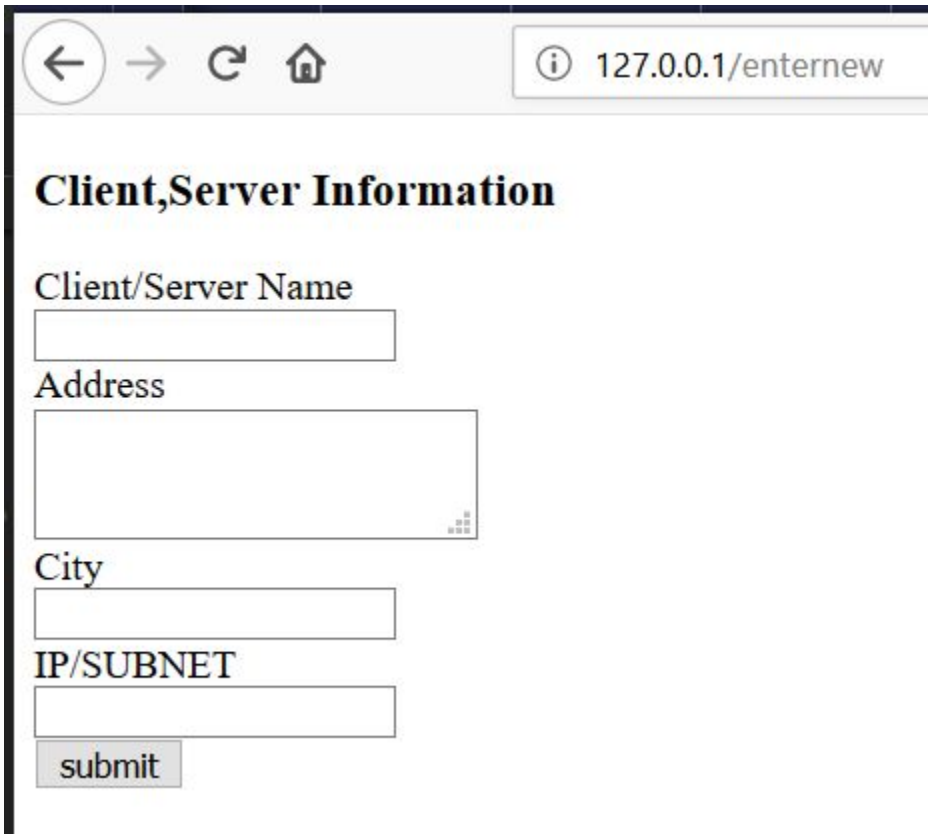
now run the python program, to get our (flask web gui) up and running :

main.py

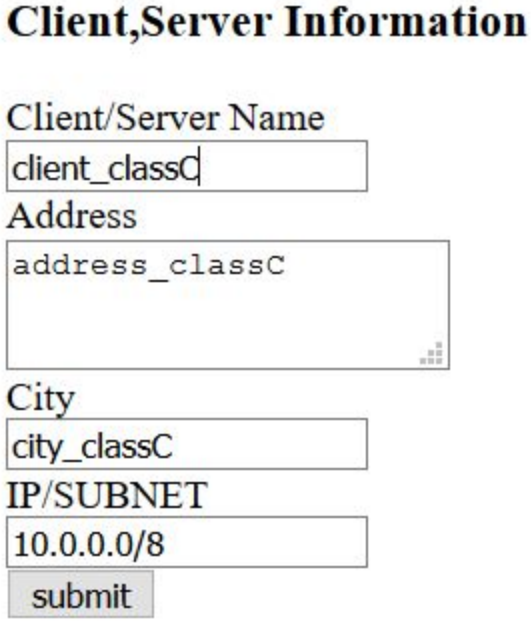
in Browser



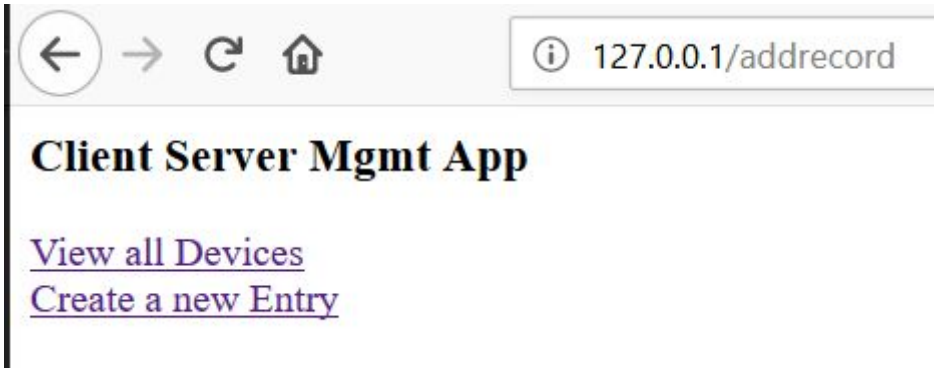
(Create a new Entry)


127.0.0.1/enternew
Client,Server Information
Client/Server Name
Address
City
IP/SUBNET
submit

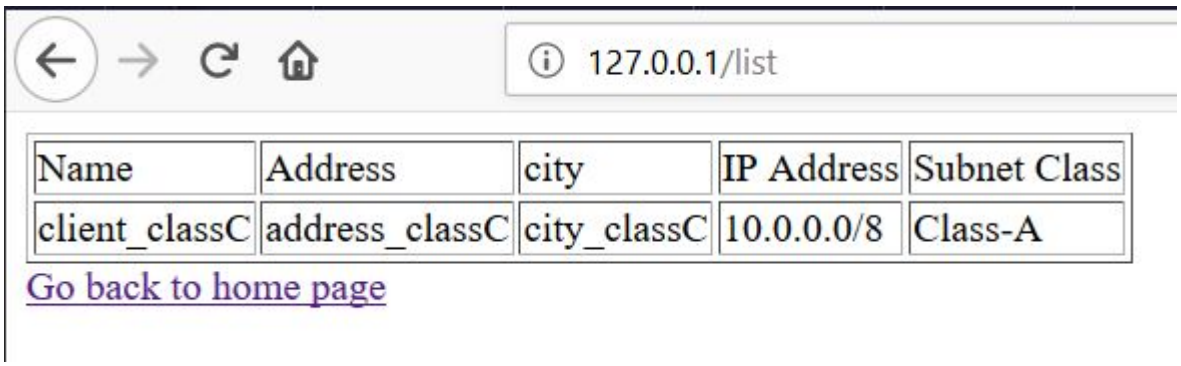
e.g.


Client,Server Information
Client/Server Name
client_classC
Address
address_classC
City
city_classC
IP/SUBNET
10.0.0.0/8
submit

now



(View all Devices)



Ha, Ha, need to fix that ! For now we can use the

dbsqlite_clientservermgmt_app_tbl_ciud_m.py

to (delete a record)

first, comment out (Step 1)

```
#====================
#==Step 1:
#====================
#result.create_table__ClientServerMgmt()
```

and activate (Step 3)

```
#====================
#==Step 3:
#====================
rows = result.select_all_data__ClientServerMgmt()
for row in rows:
    print( str ( row ) )
#    print( str ( row[0] ) )
```

Output shows us

```
(1, 'client_classC', 'address_classC', 'city_classC', '10.0.0.0/8')
```

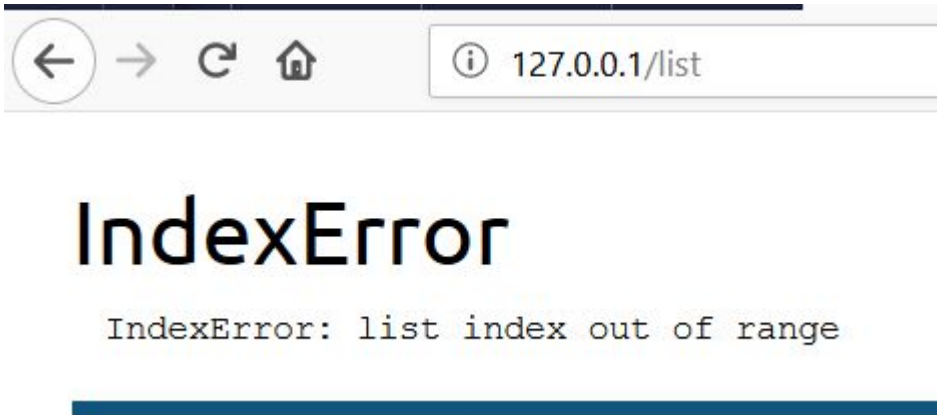
that the (id = 1)

```
#==DELETE, e.g.
result.delete_row_of_data__from__ClientServerMgmt(1)
```

will give us

```
.... cs_id = 1
sql =  DELETE from ClientServerMgmt where cs_id = 1
performed:  DELETE from ClientServerMgmt where cs_id = 1
```

should get




```
    """
    classA = IPv4Network(("10.0.0.0", "255.0.0.0"))  # or IPv4Network("10.0
.0.0/8")
    classB = IPv4Network(("172.16.0.0", "255.240.0.0"))  # or IPv4Network("
172.16.0.0/12")
    classC = IPv4Network(("192.168.0.0", "255.255.0.0"))  # or IPv4Network(
"192.168.0.0/16")
    """
```

So Lets create some Entries for Class A - C :

(View all Devices) should yield :

So, now enter examples for Class B and Class C

Here is where you should use/implement your Class for determining the Subnet Class !

Good Luck.

Kind Regards,
Scottie