SHRINE 4.2.0 Chapter 9 - Setting Up the SHRINE User and Other Users in i2b2

This section describes setting up a user in i2b2, and it uses the most recent i2b2 version (1.7.12a). Use this procedure to add users to the i2b2 PM cell's project. The SHRINE adapter must have an account that it uses to run queries in the i2b2 CRC. These instructions specifically describe how to set that up.

Think of a good password. Do not use 'demouser' with real patient data. It is a well-known default password; it protects nothing.

The value for the password field in the i2b2 PM cell's PM_USER_DATA table is a modified MD5 hash of the password. A standard MD5 algorithm will not always generate the correct hash for i2b2. Instead, use the i2b2_password_hash_generator.py script (also in shrine-setup.zip) to generate the hash:

i2b2_password_hash_generator.py

```
import hashlib
import string
import struct
import sys

def i2b2_pass(passwd):
    md5 = hashlib.md5()
    md5.update(passwd)
    digest = md5.digest()
    byte_list = []
    for b in digest:
        byte_list.append("%x" % struct.unpack('B', b))
    return string.join(byte_list, '')

def main():
    if len(sys.argv) != 2:
        print("usage: %s passwd" % sys.argv[0])
        sys.exit(1)
    passwd = sys.argv[1]
    print("%s" % i2b2_pass(passwd))

if __name__ == "__main__":
    main()
```

Run the command as follows. This script will display the hash you need to create the user as indicated above:

running python MD5 hash generator

```
$ python i2b2_password_hash_generator.py <insert_your_human-readable_password>
```

To set up a standard application user, you will need to connect to the "i2b2" database and insert the following table entries. In the default configuration of i2b2 version 1.7.12a these tables are located in the "public" schema of the "i2b2" database:

```
insert into PM_USER_DATA (user_id, full_name, password, status_cd) values ('shrine', 'SHRINE User',
'9117d59a69dc49807671a51f10ab7f', 'A');

-- The password hash you see above is for 'demouser' . Use your own password's MD5 hash provided by the python
script.

insert into PM_PROJECT_USER_ROLES (PROJECT_ID, USER_ID, USER_ROLE_CD, STATUS_CD) values ('SHRINE', 'shrine',
'USER', 'A');

insert into PM_PROJECT_USER_ROLES (PROJECT_ID, USER_ID, USER_ROLE_CD, STATUS_CD) values ('SHRINE', 'shrine',
'DATA_AGG', 'A');
```

Follow the same procedure for each of your researcher users, but for the second USER_ROLE_CD, use

`DATA_OBFSC`

instead of

DATA_AGG :

```
insert into PM_USER_DATA (user_id, full_name, password, status_cd) values ('researcherUserName', 'Researcher
Full Name', 'hashForResearcherPassword', 'A');

insert into PM_PROJECT_USER_ROLES (PROJECT_ID, USER_ID, USER_ROLE_CD, STATUS_CD) values ('SHRINE',
'researcherUserName', 'USER', 'A');

insert into PM_PROJECT_USER_ROLES (PROJECT_ID, USER_ID, USER_ROLE_CD, STATUS_CD) values ('SHRINE',
'researcherUserName', 'DATA_OBFSC', 'A');
```

.