Group Members:

Faiza Gulzar Ahmed (2303.khi.deg.001)

Haseebullah Shaikh (2303.KHI.DEG.015)

Date 10-Apr-2023

Good Software Engineer Practice

Assignment no 1.5:

Refactor following code:

```
from typing import List
import pandas as pd
class User:
   sub: bool
def notify(user: User) -> None:
   pass
def notify_users(x: List[User]) -> None:
 #Filter users with subscription and notify them.
    ▪     for u in x:
    if u.sub:
     # u.notify()
      notify(u)
```

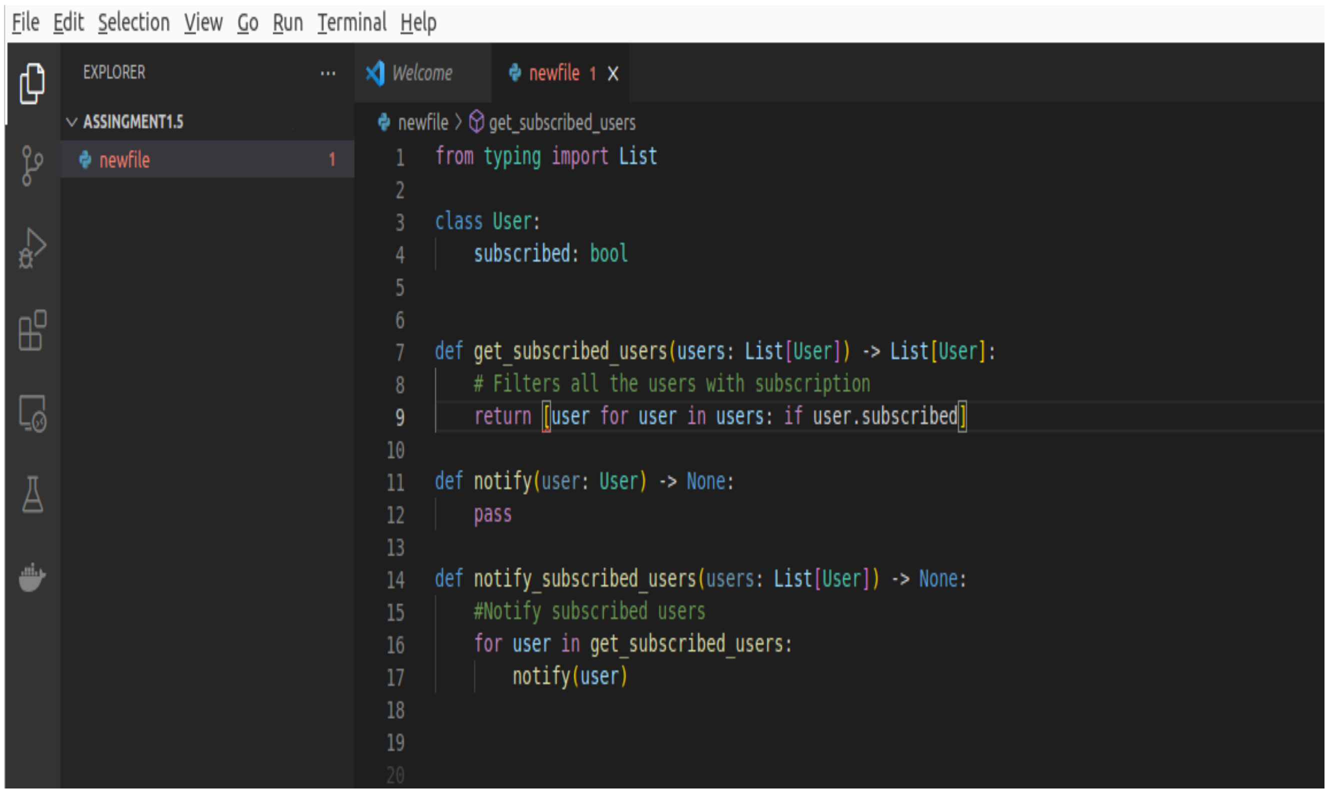
Solution:

Refactored by applying good SE practice which includes:

Yagni
Kiss
Making
Naming Rules

Using Explanatory Variables
Making functions to do one thing
Giving suitable function names

Refactor code:


File Edit Selection View Go Run Terminal Help
EXPLORER
ASSINGMENT1.5
newfile 1
Welcome
newfile 1
newfile > get_subscribed_users

```
from typing import List

class User:
    subscribed: bool


def get_subscribed_users(users: List[User]) -> List[User]:
    # Filters all the users with subscription
    return [user for user in users: if user.subscribed]

def notify(user: User) -> None:
    pass

def notify_subscribed_users(users: List[User]) -> None:
    #Notify subscribed users
    for user in get_subscribed_users:
        notify(user)
```

Solution of the Code:

from typing import List

class User:

subscribed: bool

def get_subscribed_users(users: List[User]) -> List[User]:

Filters all the users with subscription

return [user for user in users: if user.subscribed]

def notify(user: User) -> None:

```
Pass


def notify_subscribed_users(users: List[User]) -> None:

#Notify subscribed users

for user in get_subscribed_users:

notify(user)
```

Explanation of the code:

The explanation of the code Yagni is not a need add an extra functionality of the code and kiss use the don't more complex the code and using the naming rules clearly and using the explanatory variables and make the function do one thing and giving a suitable function name.

I use the library **from typing import List** and Class name define the User and use the attributes **subscribed: bool** and then used the function **def get_subscribed_user** and list the users **(users: List[User]) -> List[User]** and return the value **[user for user in users: if user.subscribed]** if user are subscribed so pass the function **def notify(user: User) -> None:** and notify the function **def notify_subscribed_users(users: List[User]) -> None:** and use the for loops repeat the fixed function and last notify the user.