Homework Assignment: Enhancing the `get_drug_group` function

Objective: Refactor the `get_drug_group` function to make it more user-friendly by allowing users to provide their input directly and validate that input.

Introduction to Python's `input` Function:

We are going to build on what we have covered in the last 2 classes. We will be using the `input` function to prompt the user for input and validate it.

Let's get familiar with the `input` function before we integrate it into our code. I suggest you use the Python interpreter to try out the examples below or start a new python file and try them out.

The `input` function in Python allows you to prompt the user for input from the command line. It always returns the input as a string. You can provide an optional message string as its argument, which will be printed as the prompt.

```
user_input = input("Please enter your name: ")
print(f"Hello, {user_input}!")
```

It should look something like this when you run it:

```
$ python3 input_example.py
Please enter your name: Caroline
Hello, Caroline!
```

For a detailed understanding and its nuances, you should refer to the Python documentation. Here's the link: Python docs on `input`

Single Responsibility Principle (SRP):

Here we are going to add another principle to our arsenal of programming principles. We previously discussed DRY (Don't Repeat Yourself). The Single Responsibility Principle, one of the SOLID principles of design, states that a function (or method) should have one and only one reason to change. In simpler terms, a function should do just one thing and do it well. This makes the code easier to understand, modify, and maintain.

In the context of our assignment:

1. The function to validate the input should only concern itself with validation.
2. The function to get the drug group should only deal with fetching the drug group from reference data.

Separating these concerns allows for easier testing, modification, and reuse of code.

Task 1: Allow User Input

Create a new function named `get_user_input` that prompts the user for input and returns the input. This function should accept a single argument - the prompt message.

Check this works by calling the function with a prompt message and printing the returned value.

Task 2: Validate User Input

Write a new function named `validate_input` that ensures the provided input is a string. This function should:

1. Accept a single argument - the user's input.
2. Validate the input to ensure it is a string, and is not empty.
3. Return the validated input.

Extra Credit Tasks (Optional):

1. Ensure that the input prompt is repeated until the user provides valid input. This is called input validation. You will need to use a `while` loop to achieve this. Read about `while` loops here: Python docs on `while` loops
2. Add an extra validation to check that there are no whitespaces in or around the user input. You will need to use the `strip` method to achieve this. Read about `strip` here: Python docs on `strip`. `strip` is another inbuilt function in Python. This should allow input of:

```
 paracetamol
fake drug
```

Task 3: Integrate User Input and Validation with `get_drug_group`

Write your script that integrates the `get_user_input` and `validate_input` functions with the existing `get_drug_group` function. The script should:

1. First, prompt the user for their input.
2. Then, validate the input using the `validate_input` function.
3. Finally, pass the validated input to the existing logic of the `get_drug_group` function.

Tips:

- When writing functions, always keep the Single Responsibility Principle in mind. Ask yourself if the function is doing only one thing.
- Test each function individually before integrating them. This modularity will help you pinpoint issues more easily.