Copilot

I apologize for the oversight in the previous unit tests. Thank you for pointing it out! Let’s correct it. Below, you’ll find the updated unit tests for the `get_pure_user_words` function:

```
import unittest

def get_pure_user_words(user_words, letters, words_from_dict):
    """
    Returns a list of user-entered words that follow the rules but are not in the game dictionary.

    Args:
        user_words (list): List of words entered by the player.
        letters (list): List of letters from the game field.
        words_from_dict (set): Set of words from the game dictionary.

    Returns:
        list: List of valid user words not in the dictionary.
    """
    central_letter = letters[4]  # Assuming the central letter is at index 4
    valid_user_words = []

    for word in user_words:
        # Check if the word follows the rules and contains the central letter
        if len(word) >= 4 and central_letter in word.lower() and word.lower() not in words_from_dict:
            valid_user_words.append(word.lower())

    return valid_user_words

class TestGetPureUserWords(unittest.TestCase):
    def test_valid_user_words(self):
        user_words = ['opto', 'open', 'uproot', 'xyz']
        letters = ['e', 'm', 'x', 'p', 'c', 'z', 'w', 'p', 'i']
        words_from_dict = {'open', 'uproot', 'propose', 'other'}
        expected_result = ['opto', 'xyz']
        self.assertEqual(get_pure_user_words(user_words, letters, words_from_dict), expected_result)

    def test_empty_user_words(self):
        user_words = []
        letters = ['e', 'm', 'x', 'p', 'c', 'z', 'w', 'p', 'i']
        words_from_dict = {'open', 'uproot', 'propose', 'other'}
        expected_result = []
        self.assertEqual(get_pure_user_words(user_words, letters, words_from_dict), expected_result)

    def test_no_central_letter(self):
        user_words = ['apple', 'banana', 'cherry']
        letters = ['e', 'm', 'x', 'p', 'c', 'z', 'w', 'p', 'i']
        words_from_dict = {'open', 'uproot', 'propose', 'other'}
        expected_result = []
        self.assertEqual(get_pure_user_words(user_words, letters, words_from_dict), expected_result)

    def test_word_in_dictionary(self):
        user_words = ['open', 'uproot', 'propose']
        letters = ['e', 'm', 'x', 'p', 'c', 'z', 'w', 'p', 'i']
        words_from_dict = {'open', 'uproot', 'propose', 'other'}
        expected_result = []
        self.assertEqual(get_pure_user_words(user_words, letters, words_from_dict), expected_result)

    def test_word_with_central_letter(self):
        user_words = ['opto', 'open', 'uproot', 'xyz']
        letters = ['e', 'm', 'x', 'p', 'c', 'z', 'w', 'p', 'i']
        words_from_dict = {'open', 'uproot', 'propose', 'other'}
        expected_result = ['opto', 'xyz']
        self.assertEqual(get_pure_user_words(user_words, letters, words_from_dict), expected_result)

if __name__ == '__main__':
    unittest.main()
```

These updated tests should now cover various scenarios and ensure the correctness of the `get_pure_user_words` function. If you have any further requests or need additional assistance, feel free to ask! 😊