Iteration 5 – Beleg für automatische Tests

Daten

Testende Person	Dennis Weinberger
Ausführungszeitpunkt	23. Januar
Getestetes Feature	Chat (erste grundlegende Funktionen)
Ergebnis	Test fehlgeschlagen
Coverage	76%

Testdurchlauf

```
(tsam_venv) C:\Users\denni\PycharmProjects\tsam\backend>pytest testing\test_chat.py --cov --cov-config=.coveragerc
============================================== test session starts ==============================================
platform win32 -- Python 3.7.2, pytest-6.1.2, py-1.9.0, pluggy-0.13.1
rootdir: C:\Users\denni\PycharmProjects\tsam\backend
plugins: cov-2.11.1
collected 5 items

testing\test_chat.py ...F.                                                                              [100%]

=================================================== FAILURES ====================================================
_____________________________________________ test_read_feed _____________________________________________

args = (), kwargs = {}, response = <Response [201]>

    @wraps(func)
    def wrapper(*args, **kwargs):
        response: Response = func(*args, **kwargs)
        assert response is not None
>       assert response.status_code == status_code
E       AssertionError

testing\decorators\assertions.py:24: AssertionError

----------- coverage: platform win32, python 3.7.2-final-0 -----------
Name                  Stmts   Miss  Cover
-----------------------------------------
controllers\chat.py      24      4    83%
models\Chat.py           10      0   100%
routers\chats.py         52     17    67%
-----------------------------------------
TOTAL                    86     21    76%

============================================== short test summary info ==============================================
FAILED testing/test_chat.py::test_read_feed - AssertionError
============================================== 1 failed, 4 passed in 2.90s ==============================================
```

Konsequenzen

Da ein Test nicht erfolgreich durchlief, muss der entsprechende Bug gefunden und dann gefixt werden, bevor die Chat-Implementation auf dem Backend in das Produkt eingebaut werden kann. Der Chat wurde von Max implementiert, und von Dennis getestet. Da der Bug schnell gefunden wurde, und das Frontend auf die Grundfunktionen bereits wartete, wurde der Bug nach dem Testdurchlauf durch Dennis gefixt. Später ist Dennis mit Max den fehlerhaften Code durchgegangen und hat ihm den Fehler erklärt, sodass er in Zukunft nicht mehr auftritt.

Bemerkungen

Beim Betrachten der Fehlermeldung zusammen mit dem fehlgeschlagenen Test ist es naheliegend, dass etwas mit dem zurückgelieferten http-Statuscode nicht stimmt. Tatsächlich ist dies auch der Fall: Die assert-Überprüfungen bis zum „return“ sind alle korrekt durchgelaufen, und nach dem Durchlauf der gesamten Funktion wird mithilfe des „request_assert“-Decorator der zurückgegebene http-Statuscode überprüft.

```python
@request_assert(200)
@oauth2(scopes=['chat_post', 'chat_read_feed'])
def test_read_feed(headers):

    new_chat_data = {
        "sender_id": user_b.user_id,
        "receiver_id": user_a.user_id,
        "content": "Message #2"
    }

    post_response = session.post(root_url + 'chat/', headers=headers, json=new_chat_data)
    assert post_response

    response = session.get(root_url + 'chat/{}/{}'.format(user_a.user_id, user_b.user_id), headers=headers)

    assert response

    json_body = json.loads(response.text)
    assert json_body[0]['content'] == 'Test chat message'
    assert json_body[0]['sender_id'] == user_a.user_id
    assert json_body[0]['receiver_id'] == user_b.user_id

    assert json_body[1]['content'] == 'Message #2'
    assert json_body[1]['sender_id'] == user_b.user_id
    assert json_body[1]['receiver_id'] == user_a.user_id

    return response
```

Fehlgeschlagener Test

Es wurde fälschlicherweise der Code 201 (Created) zurückgegeben, der bedeutet, dass eine neue Ressource erfolgreich erstellt wurde. Da das Abfragen des Chat-Feeds aber nichts erstellt (außer entsprechende Transaction-Logs), sondern nur Daten liest, ist 200 (OK) sinnvoller, und wurde im Test daher erwartet. Unsere API, die das Frontend benutzt, wurde dahingehend von den Entwickler*innen verbessert.