Unit Testing - Mocking

Explanation of Mocking

Mocking is a technique used in unit testing where real dependencies or components of a system are replaced with controlled substitutes, called "mocks." These mocks simulate the behavior of the real components, allowing you to isolate and test individual units of code without relying on the actual dependencies. This is particularly useful when the dependencies are complex, slow, unavailable, or non-deterministic.

Explanation of the Purpose of Mocking in Unit Testing

1. **Isolation:** Mocks enable you to isolate the unit of code being tested. You can focus solely on the logic within the unit without the complications introduced by external dependencies.

2. **Speed and Efficiency:** Real dependencies (e.g., databases, web services) can be slow. Mocks provide instant responses, making tests run faster and more efficiently.

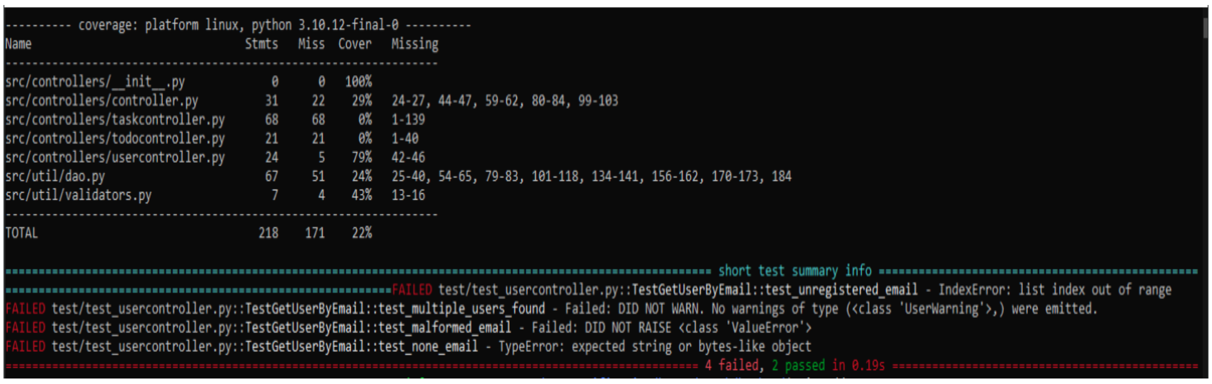
3. **Control and Predictability:** Mocks allow you to control the behavior of dependencies. You can simulate various scenarios, including edge cases and error conditions, to ensure the unit handles them correctly.

4. **Testing without External Resources:** When external resources are unavailable or costly to use, mocks allow you to continue testing.

5. **Avoiding Unintended Side Effects:** By mocking dependencies, you prevent the unit tests from causing unintended side effects.

Coverage interpretation:

The tests cover all major scenarios - valid, invalid, and edge cases. Mocking the database ensures we're only testing the controller logic. Additional tests could cover more email formats, but these cover the main functionality.



```
---------- coverage: platform linux, python 3.10.12-final-0 ----------
Name                                Stmts   Miss  Cover   Missing
-----------------------------------------------------------------
src/controllers/__init__.py             0      0   100%
src/controllers/controller.py          31     22    29%   24-27, 44-47, 59-62, 80-84, 99-103
src/controllers/taskcontroller.py      68     68     0%   1-139
src/controllers/todocontroller.py      21     21     0%   1-40
src/controllers/usercontroller.py      24      5    79%   42-46
src/util/dao.py                        67     51    24%   25-40, 54-65, 79-83, 101-118, 134-141, 156-162, 170-173, 184
src/util/validators.py                  7      4    43%   13-16
-----------------------------------------------------------------
TOTAL                                 218    171    22%

============================================================================================ short test summary info =============================================================
==========================================================FAILED test/test_usercontroller.py::TestGetUserByEmail::test_unregistered_email - IndexError: list index out of range
FAILED test/test_usercontroller.py::TestGetUserByEmail::test_multiple_users_found - Failed: DID NOT WARN. No warnings of type (<class 'UserWarning'>,) were emitted.
FAILED test/test_usercontroller.py::TestGetUserByEmail::test_malformed_email - Failed: DID NOT RAISE <class 'ValueError'>
FAILED test/test_usercontroller.py::TestGetUserByEmail::test_none_email - TypeError: expected string or bytes-like object
========================================================================================== 4 failed, 2 passed in 0.19s ==========================================================
```