Here’s your text re-printed exactly as you provided:

I'll re-read all the test files to check for any remaining logical errors.

[3 tools called]

**Updated Analysis - Logical Errors Found**

After re-examining all test files, here are the **remaining logical errors**:

**1. updateUser.test.js**

**Error 1: Data Mismatch in beforeEach (Lines 154-165)**

**Critical Issue**: The expectedUpdateObject in the beforeEach contains **different data** than what's in req.body:

// req.body (lines 113-122):

email: 'abc001@server.com',

first\_name: 'ABC',

user\_role: 'user',

sms\_comms: true

// expectedUpdateObject (lines 154-165):

email: 'abc002@server.com', // MISMATCH

first\_name: 'ABC2', // MISMATCH

user\_role: 'admin', // MISMATCH

sms\_comms: false // MISMATCH

**Problem**: When \_.omit removes user\_id from req.body, it should return the rest of the fields from req.body, not completely different values. This makes both tests in this describe block logically incorrect.

**Impact**:

* Test on line 171 expects the wrong expectedUpdateObject
* Test on line 192 expects the wrong return value from \_.omit

**Error 2: Missing Mock Setup (Line 171-189)**

**Problem**: The test "should generate the filter with an object containing user\_id" doesn't mock \_.omit, but the updateUser function will call it.

**Why This Fails**: Without mocking \_.omit, Jest will try to call the real lodash function, which may not work as expected in the test environment.

**Error 3: Data Inconsistency in mockUpdatedUser (Line 148)**

**Minor Issue**: In the beforeEach, req.body.sms\_comms is true (line 122), but mockUpdatedUser.sms\_comms is false (line 148).

**Problem**: The mock response doesn't reflect what would actually be in the database after the update.

**2. updateUserPassword.test.js**

**Error 4: Missing Mock Setup (Lines 149-165)**

**Problem**: The test "Should generate the filter with an object containing user\_id" doesn't mock argon2.hash, but the function will call it.

**Why This Fails**: Without mocking argon2.hash, the test will fail when the function tries to hash the password.

**Error 5: Incorrect Mock Return Value (Line 187)**

**Problem**:

argon2.hash.mockResolvedValue(updateObjectWithHashedPassword);

updateObjectWithHashedPassword is an **object**: { password: '$argon2id$hash' }

But argon2.hash should return a **string**, not an object.

**Should be**:

argon2.hash.mockResolvedValue('$argon2id$hash');

**Error 6: Missing Mock Setup (Lines 168-183, 197-209)**

**Problem**: Two tests don't mock argon2.hash:

* Line 168: "Should create the password update object..."
* Line 197: "Should check whether the updated password is successfully sent..."

**Why This Fails**: The function will try to call argon2.hash but it's not mocked, causing test failures.

**3. registerUser.test.js**

**No Critical Errors Found ✅**

All tests in this file appear to be logically correct with proper mock setups.

**Summary of Critical Issues**

| **File** | **Line(s)** | **Issue** | **Severity** |
| --- | --- | --- | --- |
| updateUser.test.js | 154-165 | Data mismatch between req.body and expectedUpdateObject | **CRITICAL** |
| updateUser.test.js | 171-189 | Missing \_.omit mock | **HIGH** |
| updateUser.test.js | 148 | Inconsistent sms\_comms value | **MEDIUM** |
| updateUserPassword.test.js | 149-165 | Missing argon2.hash mock | **HIGH** |
| updateUserPassword.test.js | 187 | Wrong mock return type (object vs string) | **CRITICAL** |
| updateUserPassword.test.js | 168-183, 197-209 | Missing argon2.hash mock in multiple tests | **HIGH** |

These errors will cause tests to fail or produce incorrect results, making the test suite unreliable.