### **Overview**

LoginApp is a Java Swing-based GUI application that authenticates users against a MySQL database. It contains the following functionalities:

* GUI with fields for email and password input.
* Authentication logic that verifies user credentials from the database.
* Test suite to validate different authentication scenarios.

### **Test Suite Details**

The test cases are implemented using JUnit and focus on the authenticateUser method. The following scenarios are tested:

1. Valid email and password.
2. Invalid email.
3. Empty input fields.
4. SQL injection attempt.
5. Database connection failure.

### **Test Coverage Summary**

| **Metric** | **Coverage %** | **Covered Items** | **Total Items** |
| --- | --- | --- | --- |
| **Classes** | 66% | 2 | 3 |
| **Methods** | 61% | 8 | 13 |
| **Lines** | 46% | 28 | 60 |
| **Branches** | 25% | 3 | 12 |

#### 

#### **Interpretation**

* **Class Coverage**: Two out of three classes are covered. The LoginAppTest and main login logic were tested, while some auxiliary logic in the GUI may remain untested as there is no mention of **Mockito** so I couldn't mock the results.
* **Method Coverage**: The core methods like authenticateUser and its integration scenarios are covered.
* **Line Coverage**: Half of the code lines are executed, indicating room for improvement in test coverage by the use of Mockito.
* **Branch Coverage**: The decision paths (e.g., if-else conditions) are tested.

### **Code and Test Analysis**

#### **Strengths**

* The authenticateUser method handles core scenarios like valid/invalid inputs, SQL injection, and database failures.
* All tested cases pass successfully, verifying the functionality of authenticateUser.

#### **Weaknesses**

* **GUI Tests Missing**: The Swing GUI components (like button clicks and field validations) aren't covered. This reduces overall coverage.

### **Test Case Details**

#### **Test Case 1: Valid Email and Password**

* **Description**: Tests if a valid user is authenticated successfully.
* **Expected Outcome**: User authenticated, userName matches the expected value.
* **Result**: ✅ Pass.

#### **Test Case 2: Invalid Email**

* **Description**: Tests with an invalid email that doesn't exist in the database.
* **Expected Outcome**: Authentication fails, userName is null.
* **Result**: ✅ Pass.

#### **Test Case 3: Empty Fields**

* **Description**: Tests behavior when both email and password fields are empty.
* **Expected Outcome**: Authentication fails, userName is null.
* **Result**: ✅ Pass.

#### **Test Case 4: SQL Injection Attempt**

* **Description**: Simulates an SQL injection attempt by passing malicious input.
* **Expected Outcome**: Authentication fails, database integrity preserved.
* **Result**: ✅ Pass.

#### **Test Case 5: Database Connection Failure**

* **Description**: Tests behavior when the database connection cannot be established.
* **Expected Outcome**: Exception is thrown, appropriate message logged.
* **Result**: ✅ Pass.