#### INTEGRATION TESTING

This code is designed to facilitate integration testing in a Node.js environment.

It consists of a module (retrieve.js) containing two main functions:

retrievePassword and reversePassword.

# **MongoDB Connection Setup:**

The code initializes a connection to a local MongoDB instance running on port **27017**. It utilizes the **MongoClient** class from the **mongodb** package to manage the connection.

#### retrievePassword Function:

- 1. This asynchronous function is responsible for retrieving a user's password from the MongoDB database based on the provided **username**.
- 2. Upon invocation, it establishes a connection to the MongoDB instance and selects the appropriate database and collection.
- 3. It constructs a query to find the user document based on the provided **username**.
- 4. If the user is found in the database, it logs the password to the console, reverses the password using the **reversePassword** function, logs the reversed password, and returns the reversed password.
- 5. If the user is not found, it logs a message indicating that the user was not found and returns **null**.
- 6. Finally, it ensures that the MongoDB connection is closed, regardless of the outcome.

#### reversePassword Function:

This simple function takes a string (**password**) as input and reverses it by splitting the characters, reversing the order, and joining them back together.

### **Exporting Functions:**

Both **retrievePassword** and **reversePassword** functions are exported to be used in other modules.

This module (**retrieve.js**) is utilized in integration testing scenarios, particularly with Mocha, to verify the functionality of retrieving passwords from a MongoDB database. By

calling **retrievePassword** with different usernames and asserting the returned passwords, one can ensure that the password retrieval mechanism behaves as expected.

# Test.js file:

#### **Mocha Test Suite:**

• The **describe** function is used to group tests together. In this case, it creates a test suite named "Integration Testing" to contain all integration tests.

### Stubbing the findOne Function:

Before each test, a stub function named findOneStub is defined. This function
mimics the behavior of MongoDB's findOne function. It returns a predefined user
object if the username matches 'username', and null otherwise. This stub function is
passed to retrievePassword during testing to simulate interactions with the
database without actually querying it.

#### **Test Cases:**

- The first test case, labeled "should retrieve password and reverse it", verifies that
  when retrievePassword is called with the username 'username', it correctly
  retrieves the password from the stubbed database function (findOneStub),
  reverses it using the reversePassword function, and returns the reversed
  password. The assert.strictEqual function ensures that the result matches the
  expected reversed password.
- The second test case, labeled "should handle user not found", checks the behavior of **retrievePassword** when the user is not found in the database. It ensures that the function returns **null** in this case.

### **Running the Tests:**

• These tests are designed to be executed using the Mocha testing framework. Run the following command to execute the mocha testcases: mocha test.js

# **Integration Testing Explanation:**

These tests are considered integration tests because they verify the interaction between multiple components of the system - in this case, the **retrievePassword** function and the **reversePassword** function, as well as the simulated database interaction through the stubbed **findOneStub** function. By testing these components together, we ensure that they work correctly when integrated into the larger system.