Team Gold

Testing Document

The **`SelectTest`** class plays a vital role in ensuring the data retrieval functionality of the application is working correctly.

1. TESTFIXTURE: This test case checks if the `get_one_row_id_user` and `get_one_row_id` functions of the `Select` class are working correctly by inserting a user and a nap spot into their respective tables and then retrieving the data using the mentioned functions. The retrieved data is compared with the expected values to ensure the functions are returning the correct results. If these functions are not working correctly, it could lead to incorrect data being displayed to the users or other parts of the application relying on incorrect data.
2. get_row_count: This test verifies the correctness of the `get_row_count` function, which returns the number of rows in a specified table. It checks if the function returns the expected number of rows in the `user_information` table after inserting a user.
3. determine_if_user_exists: This test checks if the `determine_if_user_exists` function, which verifies if a user exists in the database based on their username and password, is working correctly. It does this by inserting a user and then calling the function with the correct username and password combination to ensure it returns the expected user ID. If this function does not work as expected, it could lead to security vulnerabilities or incorrect access controls.

The **`UpdateTest`** class is responsible for modifying existing data in the database tables.

1. UpdateReservationTest: This test case checks if the `update_reservation` function of the `Update` class is working correctly. It does this by inserting a user and a nap spot into their respective tables, and then calling the `update_reservation` function to update the reservation status of the nap spot. The test then retrieves the updated nap spot data and compares it with the expected values to ensure the reservation status was updated correctly. If this function is not working correctly, it could lead to incorrect reservation data, double bookings, or other issues related to nap spot reservations.
2. UpdateFavoriteTest: This test case checks if the `update_favorite` function, which is likely responsible for updating a user's favorite nap spot, is working correctly. It inserts a user and two nap spots into their respective tables and then calls the `update_favorite` function to set one of the nap spots as the user's favorite. After the update, the test retrieves the user data and compares it with the expected values to ensure the favorite nap spot was updated correctly.

The **`InsertTest`** class is designed to verify the correctness of the `Insert` class, which is responsible for inserting new data into the database tables.

1. InsertUserTest: This test case calls the `insert_user` function to insert a new user into the `user_information` table with specific user details. After the insertion, the test retrieves the newly inserted user data using the `get_one_row_id_user` function from the `Select` class and compares

it with the expected values to ensure the user was inserted correctly. This ensures that the application can correctly add new users to the system.

2. InsertNapSpotTest: This test case checks if the `insert_nap_spot` function, which is responsible for inserting a nap spot into the `nap_spots` table, is working correctly. It follows a similar process to the `InsertNewNapSpotTest`, where it calls the `insert_nap_spot` function with specific nap spot details, retrieves the newly inserted data using the `get_one_row_id` function from the `Select` class, and compares it with the expected values. This test ensures that the application can correctly add nap spots to the main `nap_spots` table.

The **`DeleteTest`** class is responsible for removing data from the database tables.

1. DeleteUserTest: This test case checks if the `delete_from_table` function of the `Delete` class is working correctly when deleting a user from the `user_information` table. It inserts a test user into the table, calls the `delete_from_table` function to delete the user, and then verifies that the number of rows in the `user_information` table has decreased by one. The test assumes that there are two other users in the database before running the test. This test ensures that the application can correctly remove user accounts from the system.
2. DeleteNewNapSpotTest: This test case checks if the `delete_from_table` function is working correctly when deleting a nap spot from the `new_nap_spots` table. It follows a similar process to the previous tests, where it inserts a test nap spot into the `new_nap_spots` table, calls the `delete_from_table` function to delete the nap spot, and then verifies that the number of rows in the `new_nap_spots` table is zero, assuming there were no other nap spots in the table before running the test. This test ensures that the application can correctly remove newly added nap spots from the system before they are migrated or validated.