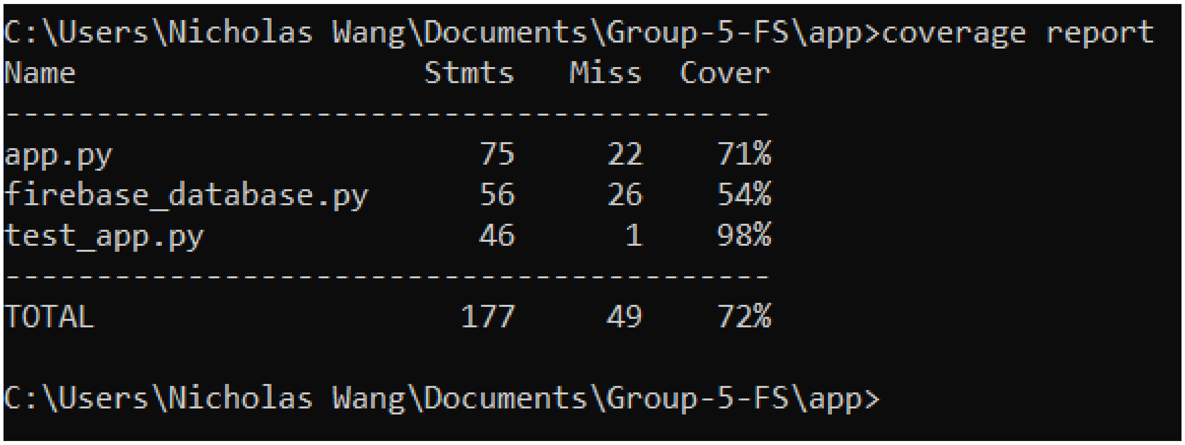


```
C:\Users\Nicholas Wang\Documents\Group-5-FS\app>coverage report
Name                     Stmts   Miss  Cover
--------------------------------------------
app.py                      75     22    71%
firebase_database.py        56     26    54%
test_app.py                 46      1    98%
--------------------------------------------
TOTAL                      177     49    72%

C:\Users\Nicholas Wang\Documents\Group-5-FS\app>
```

Fig1. initial coverage report

app.py

Duplicate Email Check in signup Route

- **Missed Statement**: The conditional check that returns an error if the email is already in use.
- **Why It Was Missed**: Initially, there was no test to register a user with an existing email, so this branch wasn't executed. The test_duplicate_email_registration test now covers this scenario, ensuring that the system prevents duplicate registrations.

Session Validation in Protected Routes (/my_bookings and /bookings)

- **Missed Statement**: Conditional checks that verify if a user session is active before allowing access to these routes.
- **Why It Was Missed**: Without tests accessing these routes without logging in, the session validation branches weren't executed. The test_access_protected_route_without_login and test_create_booking_without_login tests now cover these cases, ensuring the system restricts access to logged-in users only.

Logout Functionality and Session Clearance

- **Missed Statement**: The logout route that clears the user session and ensures restricted access to protected routes afterward.
- **Why It Was Missed**: Previously, no test checked that logging out would remove session data and prevent access to protected routes. The test_logout test now verifies this functionality, covering session clearance and restricted access post-logout.

Field Validation in create_booking_route

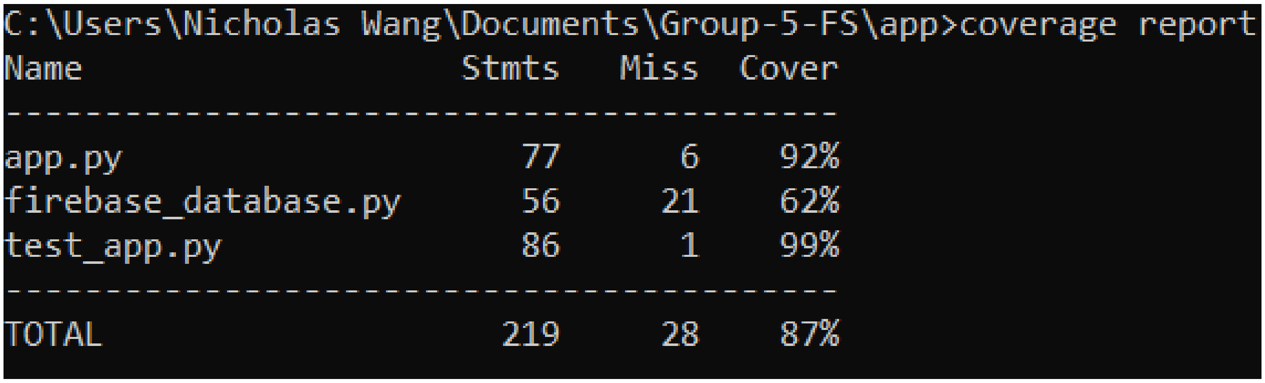
- **Missed Statement**: The branch that checks for the presence of all required fields (from, to, departure, etc.) when creating a booking.
- **Why It Was Missed**: Without a test submitting a booking request with missing fields, this validation logic wasn’t triggered. The test_create_booking_with_missing_fields test now covers this branch, ensuring the system properly validates input completeness.

Handling Empty Bookings for New Users in /my_bookings

- **Missed Statement**: The response logic in /my_bookings that returns an empty bookings list for new users.
- **Why It Was Missed**: Initially, there was no test for a new user with no bookings. The test_my_bookings_empty_for_new_user test now covers this scenario, confirming that the system responds correctly with an empty list for users with no bookings.

Invalid JSON Format Handling in signup

- **Missed Statement**: Error handling for invalid JSON format in the signup route.
- **Why It Was Missed**: Without a test for malformed JSON, this error handling logic wasn’t triggered. The test_invalid_json_format test now addresses this by sending incorrectly formatted JSON, ensuring the system returns an appropriate error for invalid JSON inputs.



```
C:\Users\Nicholas Wang\Documents\Group-5-FS\app>coverage report
Name                   Stmts   Miss  Cover
------------------------------------------
app.py                    77      6    92%
firebase_database.py      56     21    62%
test_app.py               86      1    99%
------------------------------------------
TOTAL                    219     28    87%
```

Fig2. coverage report with better app.py test script

firebase_database.py

Exception Handling in create_user

- **Missed Statement**: The exception handling block in create_user to catch any error in pushing new user data to Firebase.
- **Why It Was Missed**: The initial set of tests had no test case that mimicked an exception in user creation. This test covers test_create_user_exception by performing a mock on an exception in users_ref.push() that allows the system to catch an exception and not allow it while handling database errors smoothly.

No User Found in get_user_by_email

- **Missed Statement:** The condition wherein get_user_by_email does not find a matching user and returns None.
- **Why It Was Missed:** Initially, no tests existed for an e-mail not registered in the database. When test_get_user_by_email_not_found was added, it started checking that when there is no user with given mail, None should be returned by the system.

Exception Handling in create_booking

- **Missed Statement:** create_booking exception handling that makes an error to push new booking data to Firebase.
- **Why It Was Missed:** An exception case during the creation of a booking was not simulated by any test. Now, the test_create_booking_exception test covers this situation by mocking bookings_ref.push() with an exception to see if the system handles database errors while creating a booking.

No Bookings Found in get_bookings_by_user

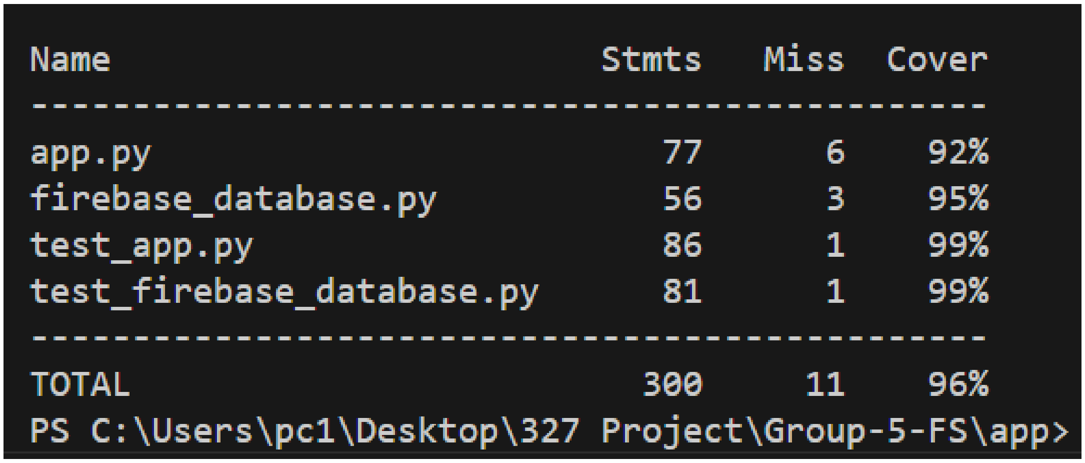
- **Missed Statement:** The condition where get_bookings_by_user does not find any bookings for a specified user and returns an empty dictionary.
- **Why It Was Missed:** Initially, there were no tests for a user with no bookings. The test_get_bookings_by_user_not_found test now covers this scenario, ensuring the system returns an empty dictionary when there are no bookings for the user.

Exception Handling in update_booking

- **Missed Statement:** The update_booking views exception handling block catches an exception, if it occurs while updating the bookings data on Firebase.
- **Why It Was Missed:** The test case was not there in order to simulate an exception during booking update. That is handled now with the test_update_booking_exception test through mocking an exception in bookings_ref.child().update() and confirming that a system handles this properly in the case of an error when a booking update occurs.

Exception Handling in delete_booking

- **Missed Statement:** This is the delete_booking exception handling block; catch all kinds of errors while deleting booking data in Firebase.
- **Why It Was Missed:** No test initially triggered an error during booking deletion. Now test_delete_booking_exception test covers such a scenario by mocking an exception in bookings_ref.child().delete() so as to see whether the system is graceful once there is an error deleting a booking.



```
Name                            Stmts   Miss  Cover
---------------------------------------------------
app.py                             77      6    92%
firebase_database.py               56      3    95%
test_app.py                        86      1    99%
test_firebase_database.py          81      1    99%
---------------------------------------------------
TOTAL                             300     11    96%
PS C:\Users\pc1\Desktop\327 Project\Group-5-FS\app>
```

Fig3. coverage report with better firebase_database.py test script