Software Testing Report

Sydney’s Airbnb Data App

**Assignment Group 31**

Fizza Batool, s5160137

Uyen Nguyen s5306965

Table of Contents

[1.0 Unit Tests 3](#_Toc49779837)

[2.0 Coverage Report 7](#_Toc49779838)

[3.0 Requirements Acceptance Testing 11](#_Toc49779839)

# Unit Tests

| **No** | **Test Case** | **Expected Results** | **Actual Results** |
| --- | --- | --- | --- |
|  | **1.0 Report all Listings Function** | | |
| 1.1 | Invalid date format. | When invalid dates are provided, an error message  about the date format should be sent to output\_text. | When invalid dates are provided, an error message  about the date format is sent to output\_text. |
| 1.2 | Invalid date range (end date before start date). | If the end date is earlier than the start date, an error  message about date order should be sent to utput\_text. | When end date is earlier than the start date, an error  message about date order is sent to output\_text. |
| 1.3 | Suburb string is empty. | An error message should be sent to output\_text if the  suburb is empty or not provided. | An error message is sent to output\_text if the suburb is  empty or not provided. |
| 1.4 | Handling cases with no matching data. | If the suburb does not match any data and the date is  in the future or in the past, a message indicating the  search has been performed (like "Searching...")  should be sent to output\_text (since it seems like a  general message, not an error message). | When suburb does not match any data and the date is  in the future or in the past, a "Searching..." message is  performed. |
|  | **2.0 Display Prices Function** | | | |
| 2.1 | Test invalid date format. | The function should recognize an invalid date format and not proceed to plot; thus, mock\_plt.subplots should not be called. | The function recognizes an invalid date format and not proceed to plot; thus, mock\_plt.subplots is not be called. |
| 2.2 | Invalid date range (end date before start date). | If the start date is later than the end date, the function should prevent plotting; therefore, mock\_plt.subplots should not be called. | When start date is later than the end date, the function prevents plotting; therefore, mock\_plt.subplots is not be called. |
| 2.3 | Test valid dates and top N. | With valid dates and specified top\_n, the function should attempt to plot and thus mock\_plt.subplots should be called once. | With valid dates and specified top\_n, the function attempts to plot and thus mock\_plt.subplots is called once. |
| 2.4 | Test no data to plot. | If there's no data within the date range, the function should not proceed to plot; thus, mock\_plt.subplots should not be called. | When there's no data within the date range, the function does not proceed to plot; thus, mock\_plt.subplots is not be called. |
|  | **3.0 Search Keywords Function** | | |
| 3.1 | Test missing dates. | If any date is None, the message "Please enter valid start and end dates.\n" should be inserted into output\_text. | When any date is None, the message "Please enter valid start and end dates.\n" is inserted into output\_text. |
| 3.2 | Test start date greater than end date. | If the start date is greater than the end date, the message "Start date cannot be greater than end date.\n" should be inserted into output\_text. | When start date is greater than the end date, the message "Start date cannot be greater than end date.\n" is inserted into output\_text. |
| 3.3 | Test empty keyword. | If no keywords are provided, the message "Please enter at least one keyword.\n" should be inserted into output\_text. | When no keywords are provided, the message "Please enter at least one keyword.\n" is inserted into output\_text. |
| 3.4 | Test no matching listings. | If no listings match the search criteria, the message "No matching listings found.\n" should be inserted into output\_text. | When no listings match the search criteria, the message "No matching listings found.\n" is inserted into output\_text. |
| 3.5 | Test matching listings. | The expected\_text that the function demo3.search\_keyword expects to produce, which is:  ID: 1  Name: Test1  URL: url1  Last Scraped: 2019-01-01 | When listings match the search criteria, the message is inserted into output\_text as follow:  ID:  Name:  URL:  Last Scraped: |
|  | **4.0 Search Keywords Function** | | |
| 4.1 | Test retrieve count with empty data set. | The function retrieve\_count should check for an empty dataset.  messagebox.showerror should be called once with arguments ("Error”. “The dataset is empty!").  messagebox.showinfo should not be called since an error condition is met. | The function retrieve\_count should check for an empty dataset.  messagebox.showerror should be called once with arguments ("Error”. “The dataset is empty!").  messagebox.showinfo should not be called since an error condition is met. |
| 4.2 | Test retrieve count with data set. | The function retrieve\_count should check the dataset, identify the number of comments mentioning cleanliness, and act accordingly.  messagebox.showinfo should be called once with text=" Count of listings mentioning cleanliness: <a number>".  messagebox.showerror should not be called since we have valid data.  count\_label.config should be called once with text="Count of listings mentioning cleanliness: <a number>" to update the GUI label. | The function retrieve\_count checks the dataset, identify the number of comments mentioning cleanliness.  messagebox.showinfo is called once with text=" Count of listings mentioning cleanliness: 121093".  count\_label.config is called once with text="Count of listings mentioning cleanliness: 121093". |
| **5.0 Test Count Apartment Bedrooms Function** | | | |
| 5.1 | Test invalid input type. | If the count\_apartment\_bedrooms function is called, it should identify that the bedroom entry "Two" is invalid (because it's not a positive integer).  Consequently, it should show an error dialog with the following message: "Please enter a valid number for bedrooms." | When the count\_apartment\_bedrooms function is called, it identifies that the bedroom entry "Two" is invalid (because it's not a positive integer).  Consequently, it shows an error dialog with the following message: "Please enter a valid number for bedrooms." |
| 5.2 | Test non-existent suburb. | Expected: 0 because "NonExistentSuburb" is not in the dataset. | 0 because "NonExistentSuburb" is not in the dataset. |
| 5.3 | Test non-existent bedroom count. | Expected: 0 because there are no listings with 99 bedrooms in the dataset. | 0 because there are no listings with 99 bedrooms in the dataset. |
| 5.4 | Test zero-bedroom apartments. | Expected: 0 assuming there are no listings with 0 bedrooms in the dataset. | 0 assuming there are no listings with 0 bedrooms in the dataset. |
| 5.5 | Test case insensitive suburb matching. | Expected: 1. The function should handle case insensitively and find 1 apartment with 2 bedrooms in Sydney. | The result returned by count\_apartments(self.dataset, 2, 'sydney'). |

# Coverage Report

**1.0 Report all Listings Function**

A screen shot of a computer program

Description automatically generatedThere are 4 functions are executed during testing:

1. Invalid Date Format: Validates that the function handles incorrectly formatted dates appropriately.
2. Out-of-order Dates: Validates the scenario where the start\_date is later than the end\_date.
3. Empty Suburb Input: Checks for handling of situations where no suburb is provided.
4. No Matching Data: Checks the function's behavior when no relevant data is found.

**Statement Coverage:**

The number of executable statements that are executed: for demo1.py, it's at 83% and for test\_demo1.py, it's at 96%. 86% of total statements have been covered across both files.

try...except block regarding the date parsing is explicitly verified in demo1.py

if condition checking date and suburb is explicitly verified in demo1.py

**2.0 Display Prices Function**

A screenshot of a computer program

Description automatically generated

There are 4 functions are executed during testing:

1. Invalid Date Format: Validates that the function handles incorrectly formatted dates appropriately.
2. Out-of-order Dates: Validates the scenario where the start\_date is later than the end\_date.
3. Valid Dates and Top N: Checks for handling of situations where valid data is provided.
4. No Data to Plot: Checks the function's behavior when no relevant data is found.

**Statement Coverage:**

The number of executable statements that are executed: for demo2.py, it's at 89% and for test\_demo2.py, it's at 97%. 91% of total statements have been covered across both files.

try...except block regarding the date parsing is explicitly verified in demo2.py

if condition checking date and suburb is explicitly verified in demo2.py

**4.0 Search Keywords Function**

A screenshot of a computer program

Description automatically generated

There are 2 functions are executed during testing:

1. Test Retrieve Count with Data: Checks for handling of situations where valid data is provided.
2. Test Retrieve Count without Data: Checks the function's behavior when no relevant data is found.

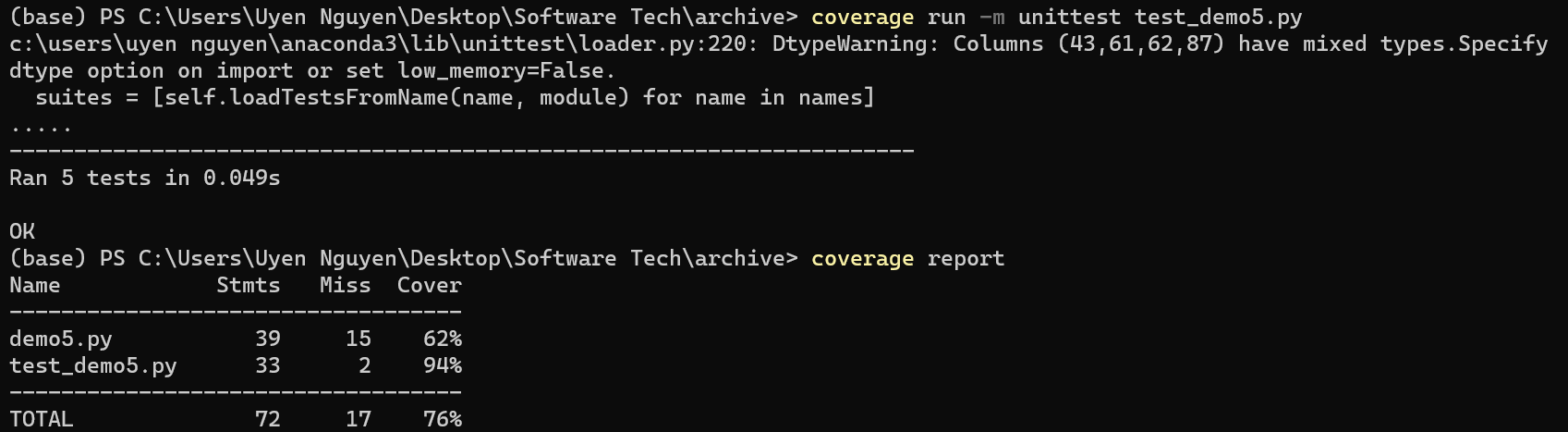
**Statement Coverage:**

The number of executable statements that are executed: for demo4.py, it's at 90% and for test\_demo4.py, it's at 96%. 92% of total statements have been covered across both files.

try...except block regarding the uploading file is explicitly verified in demo4.py

if condition checking search field cleanliness is explicitly verified in demo4.py

**5.0 Test Count Apartment Bedrooms Function**



There are 4 functions are executed during testing:

1. Invalid Input Type: Validates that the function handles incorrectly formatted input appropriately.
2. Non-existent Suburb: Validates the scenario where the suburb input incorrectly.
3. Non-existent Bedroom Count: Checks for handling of situations where invalid data is provided.
4. Zero Bedroom Apartments: Checks the function's behavior when data is empty.
5. Case Insensitive Suburb Matching: Validates that the function handles correctly matching input appropriately.

**Statement Coverage:**

The number of executable statements that are executed: for demo5.py, it's at 62% and for test\_demo2.py, it's at 94%. 76% of total statements have been covered across both files.

try...except block regarding the number of bedrooms and suburb is explicitly verified in demo5.py

if condition checking property type and bedroom is explicitly verified in demo5.py

# Requirements Acceptance Testing

| **Software  Requirement No** | **Test** | **Implemented (Full /Partial/ None)** | **Test Results (Pass/ Fail)** | **Comments (for partial implementation or failed test results)** |
| --- | --- | --- | --- | --- |
| 1 | The system must accurately report listing details between a specified date range for a given suburb. | Full |  |  |
| 2 | The system should provide informative error messages for invalid input. | Full |  |  |
| 3 | The system should handle various edge cases like empty fields and invalid date formats. | Full |  |  |
| 4 | The system should verify the availability of data for the specified period and notify the user if data is not available. | Full |  |  |
| 5 | The GUI should interactively display information or error messages. | Full |  |  |
| 6 | Given valid keywords and available data, the system should retrieve and display the count of listings that match the keywords. | Full |  |  |
| 7 | The system should accurately count the number of apartments in the given suburb with the specified number of bedrooms. | Full |  |  |
| 8 | The system should return a count of zero for non-existent suburbs or non-existent bedroom counts without errors. | Full |  |  |
| 9 | The suburb matching should be case-insensitive. | Full |  |  |