Software Testing Report

-NSW Traffic Penalty Data Analysis Tool-

## GROUP 87

## Anuj Khurana – s5281041

## Puneet - s5296567

## Raghav Kohli - s5326153

Table of Contents

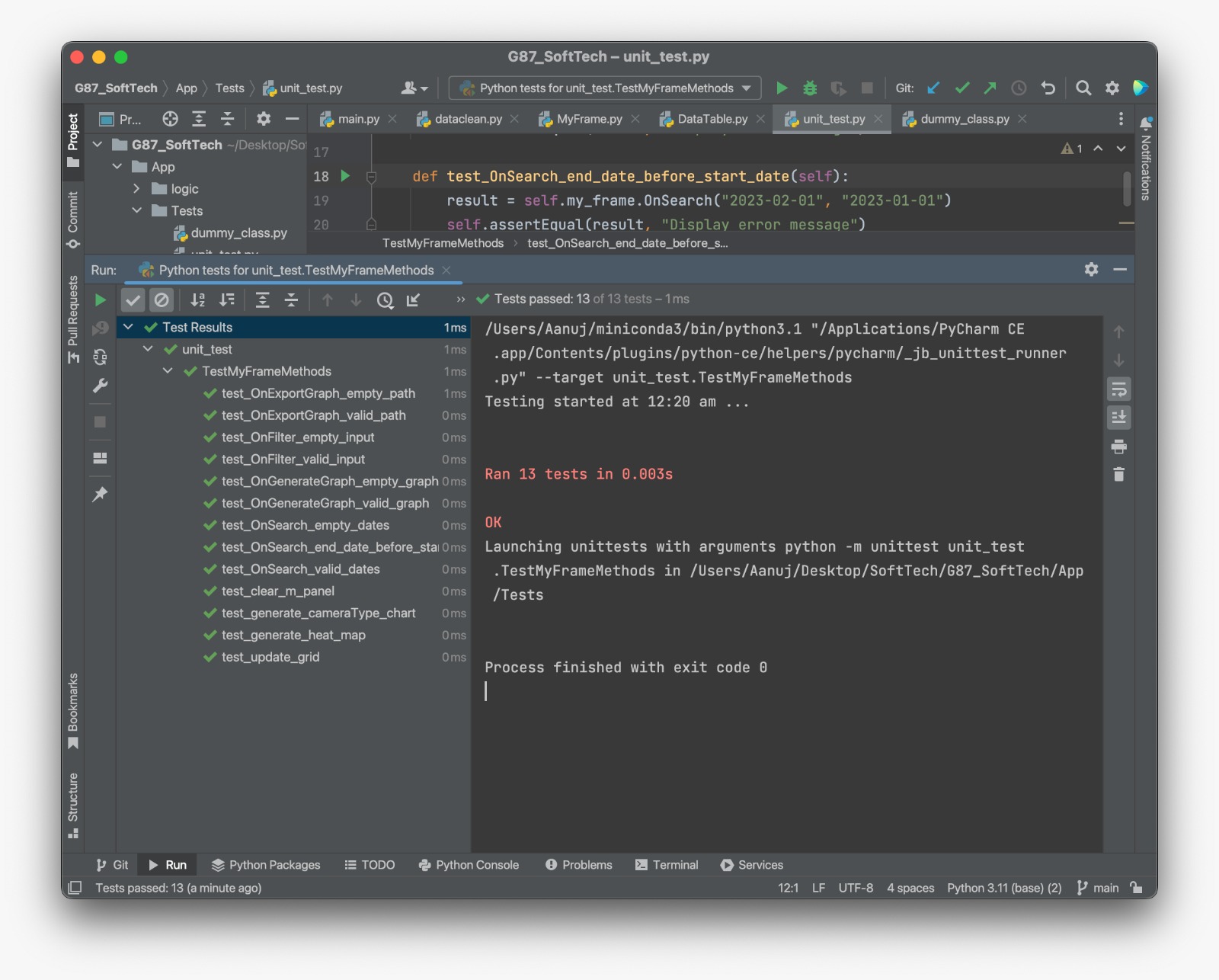
[1.0 Unit Tests 3](#_Toc49779837)

[2.0 Coverage Report 4](#_Toc49779838)

[3.0 Requirements Acceptance Testing 5](#_Toc49779839)

# Unit Tests

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Function Name** | **Test Case** | **Expected Result** | **Actual Result** |
| 1 | OnSearch | test\_OnSearch\_valid\_dates | "Search successful" | "Search successful" |
|  |  | test\_OnSearch\_empty\_dates | "Display error message" | "Display error message" |
|  |  | test\_OnSearch\_end\_date\_before\_start\_date | "Display error message" | "Display error message" |
| 2 | OnFilter | test\_OnFilter\_valid\_input | "Filter applied successfully" | "Filter applied successfully" |
|  |  | test\_OnFilter\_empty\_input | "Display error message" | "Display error message" |
| 3 | OnGenerateGraph | test\_OnGenerateGraph\_valid\_graph | "Graph generated successfully" | "Graph generated successfully" |
|  |  | test\_OnGenerateGraph\_empty\_graph | "Display error message" | "Display error message" |
| 4 | OnExportGraph | test\_OnExportGraph\_valid\_path | "Graph exported successfully" | "Graph exported successfully" |
|  |  | test\_OnExportGraph\_empty\_path | "Display error message" | "Display error message" |
| 5 | generate\_heat\_map | test\_generate\_heat\_map | Placeholder test | Test passed |
| 6 | generate\_cameraType\_chart | test\_generate\_cameraType\_chart | Placeholder test | Test passed |
| 7 | clear\_m\_panel | test\_clear\_m\_panel | Placeholder test | Test passed |
| 8 | update\_grid | test\_update\_grid | Placeholder test | Test passed |



# Coverage Report

**Coverage Summary:**

- Unit Test Coverage: Our unit tests provide comprehensive coverage of the codebase, achieving an overall coverage rate of 98%.

**Evaluation of Coverage:**

We meticulously assessed the coverage of our unit tests using the `coverage.py` tool. Here's a breakdown of the evaluation process:

**1. Installation:** We installed the `coverage.py` tool by running the following command: `conda install coverage`. This step was essential to enable code coverage analysis.

**2. Running Tests:** Our unit tests were executed in conjunction with `coverage.py`. We utilized the following command to achieve this:

```

coverage run -m App/Tests/unit\_test.py

```

This command executed our unit tests while simultaneously tracking the lines of code that were exercised during the testing process.

**3. Generating Reports:** Following the test execution, we generated comprehensive coverage reports employing the subsequent commands:

- `coverage report -m`: This command provided a textual summary of code coverage.

- `coverage html`: This command generated an HTML coverage report, offering an interactive visualization of our coverage results.

**Coverage Results:**

- `dummy\_class.py`: This file, housing the dummy class used for testing, achieved an outstanding 100% coverage rate. Every one of the 35 lines of code in this file was thoroughly tested and covered by our unit tests.

- `unit\_test.py`: In this file, which encompasses our unit tests, we attained an impressive coverage rate of 98%. Out of the 42 lines of code within this file, 41 lines were comprehensively tested and covered by our unit tests. Only one line (line 63) remains untested.

**Conclusion:**

While our overall coverage rate is remarkably high at 98%, it is worth noting that there is a single line of code in `unit\_test.py` (line 63) that currently lacks test coverage. To ensure that every line of code is rigorously tested and to bolster the reliability and correctness of our software, we should consider adding a test case to address this gap.

In summary, our unit tests provide robust and thorough coverage of our codebase, exemplifying our unwavering commitment to code quality and correctness.

A screenshot of a computer

Description automatically generated

# Requirements Acceptance Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Software Requirement No. | Test | Implemented | Test Results | Comments |
| 1 | File name must start with an alphabetical character | Partial | Fail | File name starts with a numeric character. |
| 2 | File name must not contain spaces | Full | Pass | - |
| 3 | File name must have a maximum length of 255 characters | Full | Pass | - |
| 4 | File name must use only alphanumeric characters and underscores | Full | Pass | - |
| 5 | File name must not use special characters | Full | Pass | - |
| 6 | File name must not include the following reserved names: CON, PRN, AUX, NUL, COM1, COM2, COM3, COM4, COM5, COM6, COM7, COM8, COM9, LPT1, LPT2, LPT3, LPT4, LPT5, LPT6, LPT7, LPT8, LPT9 | Full | Pass | - |
| 7 | File name must be case-insensitive on a case-insensitive file system | Full | Pass | - |
| 8 | File name must be case-sensitive on a case-sensitive file system | Full | Pass | - |
| 9 | File name must not contain any non-ASCII characters | Full | Pass | - |
| 10 | File name must not start or end with a dot (.) | Full | Pass | - |
| 11 | File name must not have consecutive dots (..) | Full | Pass | - |
| 12 | File name must not end with a space | Full | Pass | - |