

# Verification and Validation Report: SynthEddy

Phil Du (Software)  
Nikita Holyev (Theory)

April 14, 2024

# 1 Revision History

| Date       | Version | Notes          |
|------------|---------|----------------|
| 2024-04-15 | 1.0     | Initial Commit |

## 2 Symbols, Abbreviations and Acronyms

| symbol | description |
|--------|-------------|
| T      | Test        |

[symbols, abbreviations or acronyms – you can reference the SRS tables if needed —SS]

# Contents

|           |  |           |
|-----------|--|-----------|
| <b>1</b>  | <b>Revision History</b>                      | <b>i</b>  |
| <b>2</b>  | <b>Symbols, Abbreviations and Acronyms</b>   | <b>ii</b> |
| <b>3</b>  | <b>Functional Requirements Evaluation</b>    | <b>1</b>  |
| <b>4</b>  | <b>Nonfunctional Requirements Evaluation</b> | <b>1</b>  |
| 4.1       | Usability . . . . .                          | 1         |
| 4.2       | Performance . . . . .                        | 1         |
| 4.3       | etc. . . . .                                 | 1         |
| <b>5</b>  | <b>Comparison to Existing Implementation</b> | <b>1</b>  |
| <b>6</b>  | <b>Unit Testing</b>                          | <b>1</b>  |
| <b>7</b>  | <b>Changes Due to Testing</b>                | <b>1</b>  |
| <b>8</b>  | <b>Automated Testing</b>                     | <b>1</b>  |
| <b>9</b>  | <b>Trace to Requirements</b>                 | <b>2</b>  |
| <b>10</b> | <b>Trace to Modules</b>                      | <b>2</b>  |
| <b>11</b> | <b>Code Coverage Metrics</b>                 | <b>2</b>  |

## List of Tables

## List of Figures

|   |                           |   |
|---|---------------------------|---|
| 1 | Coverage report . . . . . | 2 |
|---|---------------------------|---|

This document ...

### **3 Functional Requirements Evaluation**

### **4 Nonfunctional Requirements Evaluation**

#### **4.1 Usability**

#### **4.2 Performance**

#### **4.3 etc.**

### **5 Comparison to Existing Implementation**

This section will not be appropriate for every project.

### **6 Unit Testing**

### **7 Changes Due to Testing**

[This section should highlight how feedback from the users and from the supervisor (when one exists) shaped the final product. In particular the feedback from the Rev 0 demo to the supervisor (or to potential users) should be highlighted. —SS]

### **8 Automated Testing**

```
pytest --cov=src --cov-report html test -m "(unit and not slow) or  
system"
```

## 9 Trace to Requirements

## 10 Trace to Modules

## 11 Code Coverage Metrics

Coverage is tested with `pytest-cov`, using the command shown in Section 8 that executes unit and system tests. An overall coverage of 98% is achieved, with coverage report is shown in Figure 1.

A screenshot of a terminal window showing a coverage report. The report title is 'Coverage report: 98%' and it includes a timestamp 'coverage.py v7.4.4, created at 2024-04-14 16:14 -0400'. Below this is a table with 5 columns: 'Module', 'statements', 'missing', 'excluded', and 'coverage'. The table lists 10 modules and a total, with their respective statement counts, missing counts, excluded counts, and coverage percentages. The modules are: src/main.py (59 statements, 2 missing, 0 excluded, 97% coverage), src/modules/eddy.py (15 statements, 0 missing, 2 excluded, 100% coverage), src/modules/eddy\_profile.py (41 statements, 0 missing, 0 excluded, 100% coverage), src/modules/file\_io.py (49 statements, 0 missing, 0 excluded, 100% coverage), src/modules/flow\_field.py (186 statements, 0 missing, 4 excluded, 100% coverage), src/modules/query.py (76 statements, 6 missing, 0 excluded, 92% coverage), src/modules/shape\_function.py (26 statements, 0 missing, 0 excluded, 100% coverage), src/modules/utlis.py (17 statements, 0 missing, 0 excluded, 100% coverage), src/modules/visualize.py (27 statements, 0 missing, 0 excluded, 100% coverage), and a Total row (496 statements, 8 missing, 6 excluded, 98% coverage). The report footer is 'coverage.py v7.4.4, created at 2024-04-14 16:14 -0400'.

| Module                        | statements | missing | excluded | coverage |
|-------------------------------|------------|---------|----------|----------|
| src/main.py                   | 59         | 2       | 0        | 97%      |
| src/modules/eddy.py           | 15         | 0       | 2        | 100%     |
| src/modules/eddy_profile.py   | 41         | 0       | 0        | 100%     |
| src/modules/file_io.py        | 49         | 0       | 0        | 100%     |
| src/modules/flow_field.py     | 186        | 0       | 4        | 100%     |
| src/modules/query.py          | 76         | 6       | 0        | 92%      |
| src/modules/shape_function.py | 26         | 0       | 0        | 100%     |
| src/modules/utlis.py          | 17         | 0       | 0        | 100%     |
| src/modules/visualize.py      | 27         | 0       | 0        | 100%     |
| Total                         | 496        | 8       | 6        | 98%      |

Figure 1: Coverage report

The only two modules that are not fully covered are `main` and `query`. The not covered lines are either related to exception handling with those exceptions already tested in lower level modules, or impossible to reach in automated testing (e.g. plot pop-up).

## References

## Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Reflection. Please answer the following question:

1. In what ways was the Verification and Validation (VnV) Plan different from the activities that were actually conducted for VnV? If there were differences, what changes required the modification in the plan? Why did these changes occur? Would you be able to anticipate these changes in future projects? If there weren't any differences, how was your team able to clearly predict a feasible amount of effort and the right tasks needed to build the evidence that demonstrates the required quality? (It is expected that most teams will have had to deviate from their original VnV Plan.)