**System Test Document**

**For**

**SWIP--SDTO-2025-13**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Editor(s) / Author(s)** | **Date** | **Change description** | **Revision** |
| Qian Peisheng | 28 July 2025 | Add test information | V0.1 |
|  |  |  |  |

Contents

[1. Introduction 3](#_Toc204598569)

[1.1 Acronyms 3](#_Toc204598570)

[1.2 References 3](#_Toc204598571)

[2. Test Cases 4](#_Toc204598572)

[2.1 Test Suite 1 : Flask HTTP Integration 4](#_Toc204598573)

[2.1.1 STC-1.1: Home page HTTP rendering 4](#_Toc204598574)

[2.1.2 STC-1.2: SiC data page rendering 5](#_Toc204598575)

[2.1.3 STC-1.3: Model page initialization 6](#_Toc204598576)

[2.2 Test Suite 2: Model API Test 6](#_Toc204598577)

[2.2.1 STC-2.1: Thickness models API 6](#_Toc204598578)

[2.2.2 STC-2.2: Doping models API 7](#_Toc204598579)

[2.3 Test Suite 3: Caches and Updates API 8](#_Toc204598580)

[2.3.1 STC-3.1: Thickness models API 8](#_Toc204598581)

[2.3.2 STC-3.2: Cell update functionality 8](#_Toc204598582)

# Introduction

## Acronyms

|  |  |
| --- | --- |
| Acronym | Definition |
| API | Application Programming Interface |
| HTTP | HyperText Transfer Protocol |
| JSON | JavaScript Object Notation |
| FR | Functional Requirement |
| STC | System Test Case |

## References

* Functional Requirements document
* SWIP--SDTO-2025-13
* See the code repository for more information.

# Test Cases

*[The following are the Test suites and Test cases. Test suite is a scenario as a collection of related test cases. A test suite may have one or more test cases. For example, test scenario could be “User Login”, and then various test cases under this test scenario could be “Enter valid credential”, “Enter Invalid credentials”, “Reset password”.*

*The Test cases are derived based on the functional requirements and should be given reference to the Requirement ID stated in the Functional requirements document. For each requirement, at least one test case is mandatory. It is recommended to create both positive and negative test cases.]*

***Test Case ID numbering scheme:***

*Purpose of System Test case Id is to uniquely identify a System test case to refer to a test and also to use in test log. System Test case ID numbering scheme is as following:*

*<STC-Sequence Number of Test Suite>.<Sequence Number of Test case under the test Suite>*

*e.g. For test case 1 and 2 under System Test suite 2, the numbering of test cases will be STC-2.1, STC-2.2 respectively.*

## Test Suite 1 : Flask HTTP Integration

|  |  |
| --- | --- |
| STC-1.1: Home page HTTP rendering | |
| Author: | <Name(s)> Qian Peisheng |
| **Pre-Condition:** | < state pre-condition e.g. which screen to start with>  Flask application running in test environment using Flask test client. |
| **Test Data** | N/A (uses demo data packaged with repository) |
| **Misc.** | **-** |
| **Reference** | **Requirement ID # (e.g. FR01)** |
| |  |  | | --- | --- | | **Steps:** | **Expected Results:** | | <Step 1>  Send HTTP GET request to `/`. | <Expected Result 1>  Expected: Server responds with status code 200 OK. | | <Step 2>  Verify HTML content contains expected page elements. | <Expected Result 2>  Expected: HTML body present and non‑empty. | |  |  | | |
| **Test Result (after execution)** | Pass |

|  |  |
| --- | --- |
| STC-1.2: SiC data page rendering | |
| Author: | <Name(s)> Qian Peisheng |
| **Pre-Condition:** | < state pre-condition e.g. which screen to start with>  Flask application running in test environment using Flask test client. |
| **Test Data** | N/A (uses demo data packaged with repository) |
| **Misc.** | **-** |
| Reference | FR01 |
| |  |  | | --- | --- | | **Steps:** | **Expected Results:** | | <Step 1>  Send HTTP GET request to `/sic\_data`. | <Expected Result 1>  Expected: Server responds with status code 200 OK. | | <Step 2>  Verify HTML content contains expected page elements. | <Expected Result 2>  Expected: HTML body present and non‑empty. | |  |  | | **Test Result (after execution)** | Pass | | |

**< Add more Test Cases if applicable for the Test Suite>**

|  |  |
| --- | --- |
| STC-1.3: Model page initialization | |
| Author: | <Name(s)> Qian Peisheng |
| **Pre-Condition:** | < state pre-condition e.g. which screen to start with>  Flask application running in test environment using Flask test client. |
| **Test Data** | N/A (uses demo data packaged with repository) |
| **Misc.** | **-** |
| Reference | FR01 |
| |  |  | | --- | --- | | **Steps:** | **Expected Results:** | | <Step 1>  Send HTTP GET request to `/sic\_model`. | <Expected Result 1>  Expected: Server responds with status code 200 OK. | | <Step 2>  Verify HTML content contains expected page elements. | <Expected Result 2>  Expected: HTML body present and non‑empty. | |  |  | | **Test Result (after execution)** | Pass | | |

## Test Suite 2: Model API Test

|  |  |
| --- | --- |
| STC-2.1: Thickness models API | |
| Author: | <Name(s)> Qian Peisheng |
| **Pre-Condition:** | < state pre-condition e.g. which screen to start with>  Flask application running in test environment using Flask test client. |
| **Test Data** | **N/A (uses demo data packaged with repository)** |
| **Misc.** | **-** |
| Reference | FR01 |
| |  |  | | --- | --- | | **Steps:** | **Expected Results:** | | <Step 1>  Send HTTP GET request to `/get\_thickness\_models`. | <Expected Result 1>  Expected: Status code 200 OK. | | <Step 2>  Verify response JSON list. | <Expected Result 2>  Expected: List length ≥ 1 and contains model names. | |  |  | | **Test Result (after execution)** | Pass | | |

|  |  |
| --- | --- |
| STC-2.2: Doping models API | |
| Author: | <Name(s)> Qian Peisheng |
| **Pre-Condition:** | < state pre-condition e.g. which screen to start with>  Flask application running in test environment using Flask test client. |
| **Test Data** | N/A (uses demo data packaged with repository) |
| **Misc.** | **-** |
| Reference | FR01 |
|  |  |
| |  |  | | --- | --- | | **Steps:** | **Expected Results:** | | <Step 1>  Send HTTP GET request to `/get\_doping\_models`. | <Expected Result 1>  Expected: Status code 200 OK. | | <Step 2>  Validate returned list. | <Expected Result 2>  Expected: List contains available doping prediction models. | |  |  | |  |  | | **Test Result (after execution)** | Pass | | |

## Test Suite 3: Caches and Updates API

|  |  |
| --- | --- |
| STC-3.1: Thickness models API | |
| Author: | <Name(s)> Qian Peisheng |
| **Pre-Condition:** | < state pre-condition e.g. which screen to start with>  Flask application running in test environment using Flask test client. |
| **Test Data** | **N/A (uses demo data packaged with repository)** |
| **Misc.** | **-** |
| Reference | FR01 |
| |  |  | | --- | --- | | **Steps:** | **Expected Results:** | | <Step 1>  Send sequence of two HTTP GET requests to `/get\_cached\_data` with identical parameters. | <Expected Result 1>  Expected: Second request served faster using cached data. | | <Step 2>  Compare responses. | <Expected Result 2>  Expected: Both responses identical, confirming cache validity. | |  |  | | **Test Result (after execution)** | Pass | | |

|  |  |
| --- | --- |
| STC-3.2: Cell update functionality | |
| Author: | <Name(s)> Qian Peisheng |
| **Pre-Condition:** | < state pre-condition e.g. which screen to start with>  Flask application running in test environment using Flask test client. |
| **Test Data** | N/A (uses demo data packaged with repository) |
| **Misc.** | **-** |
| Reference | FR01 |
|  |  |
| |  |  | | --- | --- | | **Steps:** | **Expected Results:** | | <Step 1>  Initial read‑back – GET /sic\_data (or /get\_cached\_data) and locate row cell\_id="C01"; note the current value of thickness. | <Expected Result 1>  Response 200 OK with JSON payload; row C01 present and thickness ≠ 42.5 (ensures the change is observable). | | <Step 2>  Update call – POST /update\_cell with the JSON body shown in Test Data (thickness = 42.5). | <Expected Result 2>  Response 200 OK with body:  {"status": "success", "updated": true} | | <Step 3>  Read‑back after update – repeat the GET /sic\_data call. | <Expected Result 3>  Row C01 now shows thickness = 42.5. | | <Step 4>  State‑persistence check – perform any other endpoint (e.g., GET /sic\_model) that relies on the same cached dataset. | <Expected Result 4>  All subsequent responses reflect the updated value, confirming cache/database consistency. | | **Test Result (after execution)** | Pass | | |

**< Add more Test Cases if applicable for the Test Suite>**

**<Add more Test Suites and Test Cases>**