1. **Overview**
2. **Introduction**

This document contains detailed implementation and testing information for Team 3's SafeHome Security System project, fulfilling the requirements outlined in the Software Requirements Specification (SRS) and the Software Design Specification (SDS). We implemented the core logic and security domain using Python, and structured the user interface and system simulation environment as follows:

• Web Interface: Implemented using a Flask backend for logic and plain > HTML for the user interface presentation.

• Virtual Devices: Established simulated devices (Virtual Devices) > based on the documentation provided by the TA (Teaching Assistant) > to emulate sensor and control panel functionalities.

We performed comprehensive quality assurance by utilizing Pytest for all test levels, including Unit, Integration, and System-Level testing, supplemented by Coverage analysis to ensure structural test completeness. This report provides full traceability from the design models to the code implementation, detailing the test procedures, environments, and final results.

1. **Goal**

The primary goals of this implementation and testing phase were to:

• Software Engineering Principles: Exercise core software engineering > principles in the implementation phase, maintaining traceability > from the SRS (Requirements) and SDS (Design).

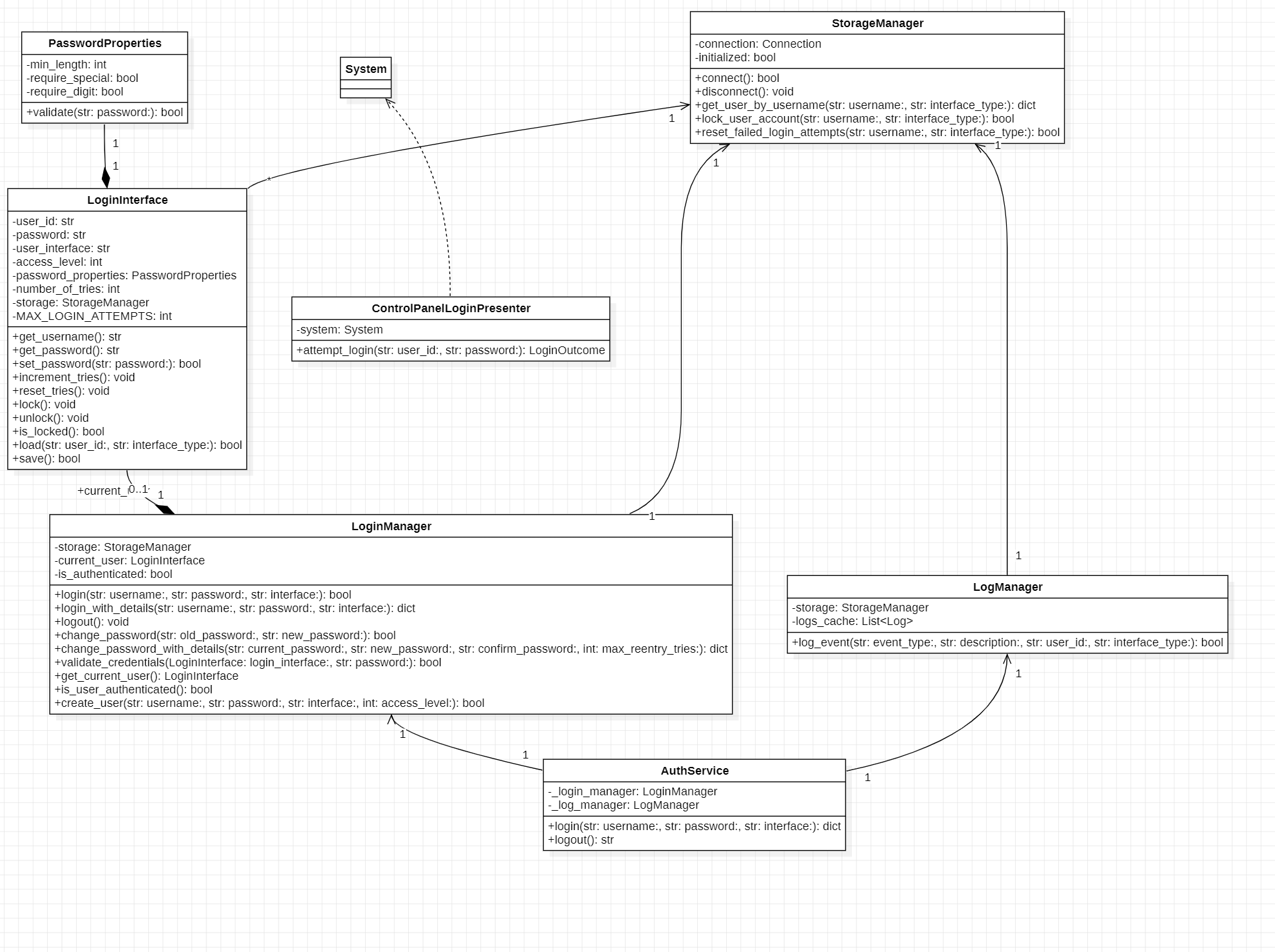
• Testing Proficiency: Demonstrate proficiency in performing software > testing at various levels: Unit Tests, Integration Tests, and > System-Level Tests.

• Program Documentation: Learn how to properly document the software > implementation, including the use of comments in code, generating > user manuals (if required), and producing this structured testing > report.

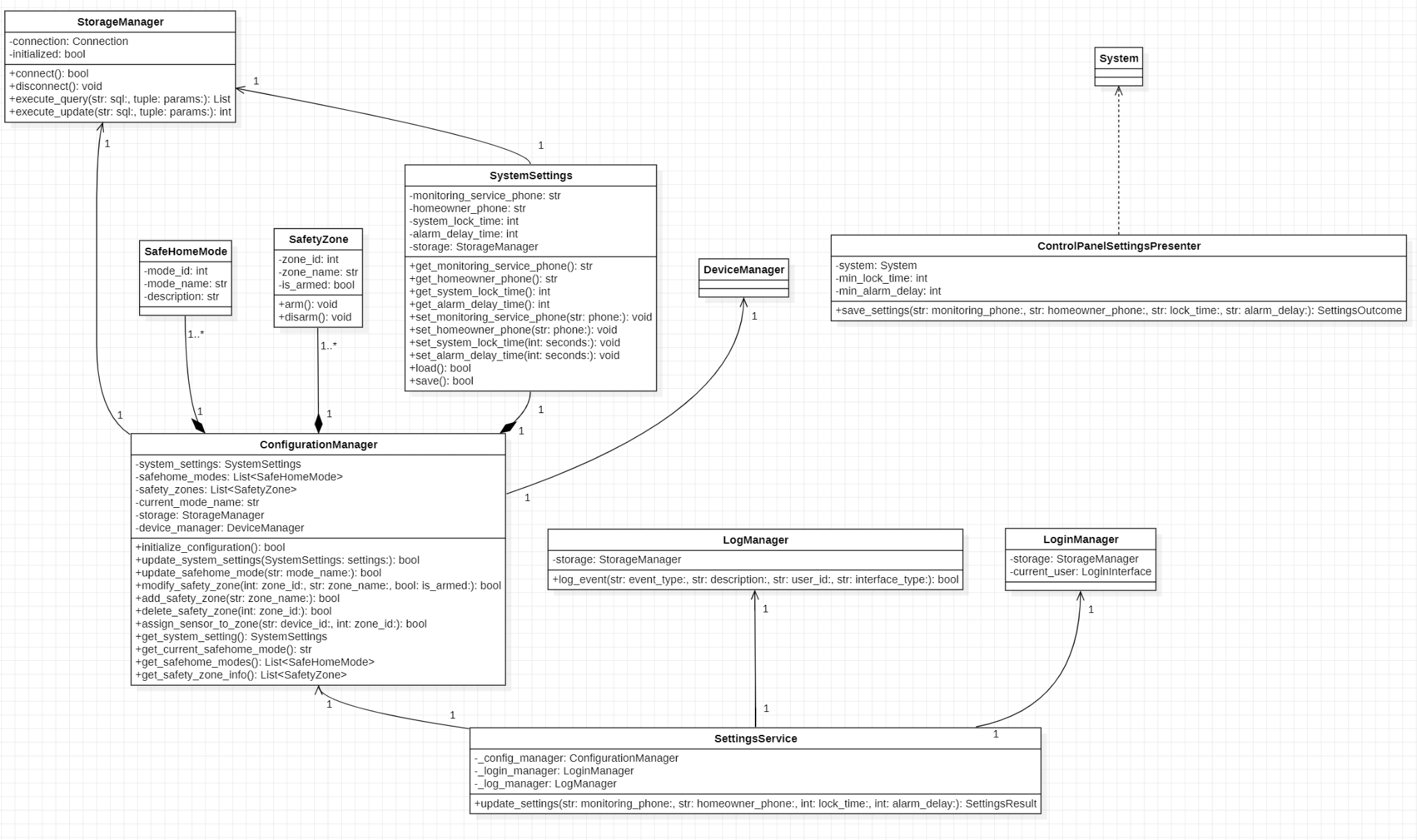
• Quality Assurance: Achieve a high level of code coverage, > specifically focusing on branch coverage, to certify the quality > and robustness of the implemented system logic.

# II. Class Diagram

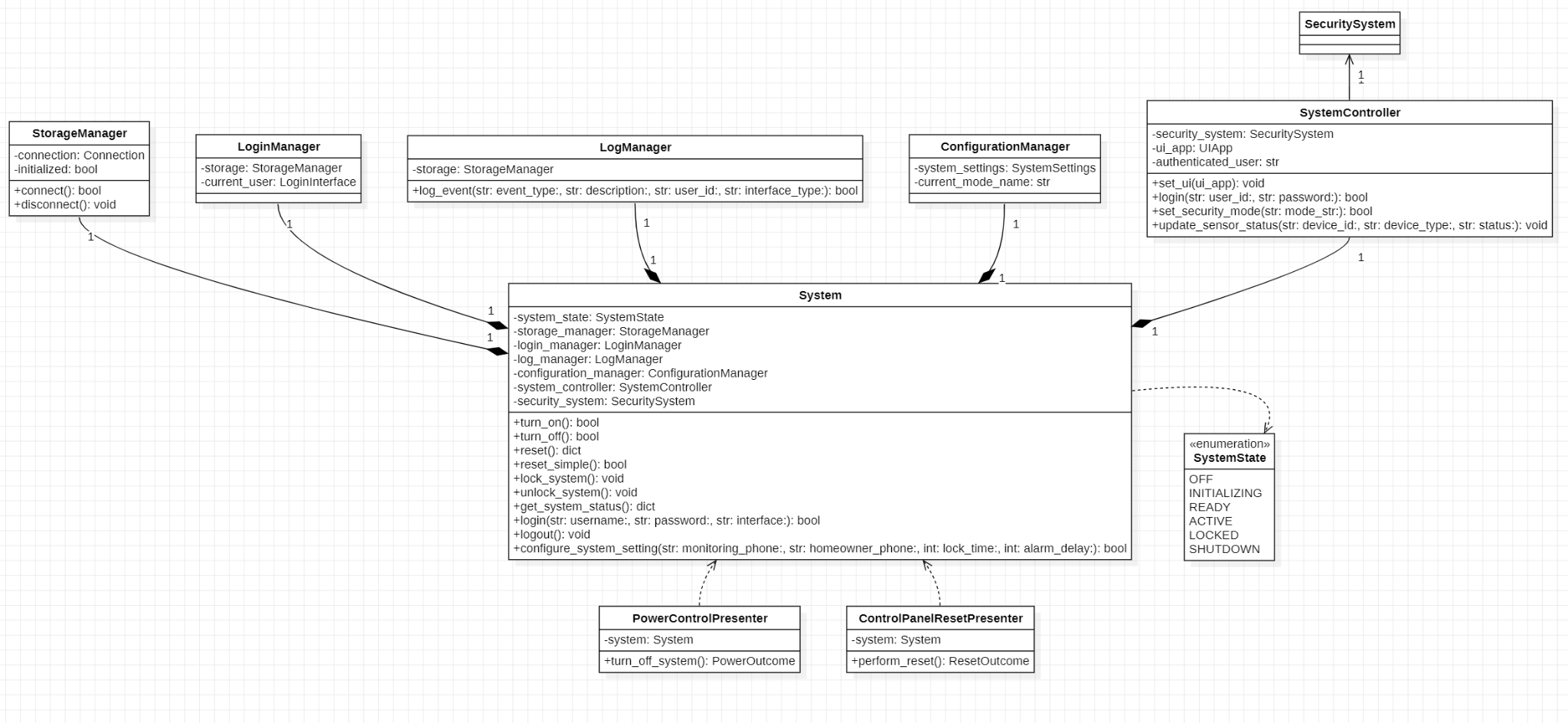
## Class Diagram of Login Module

****

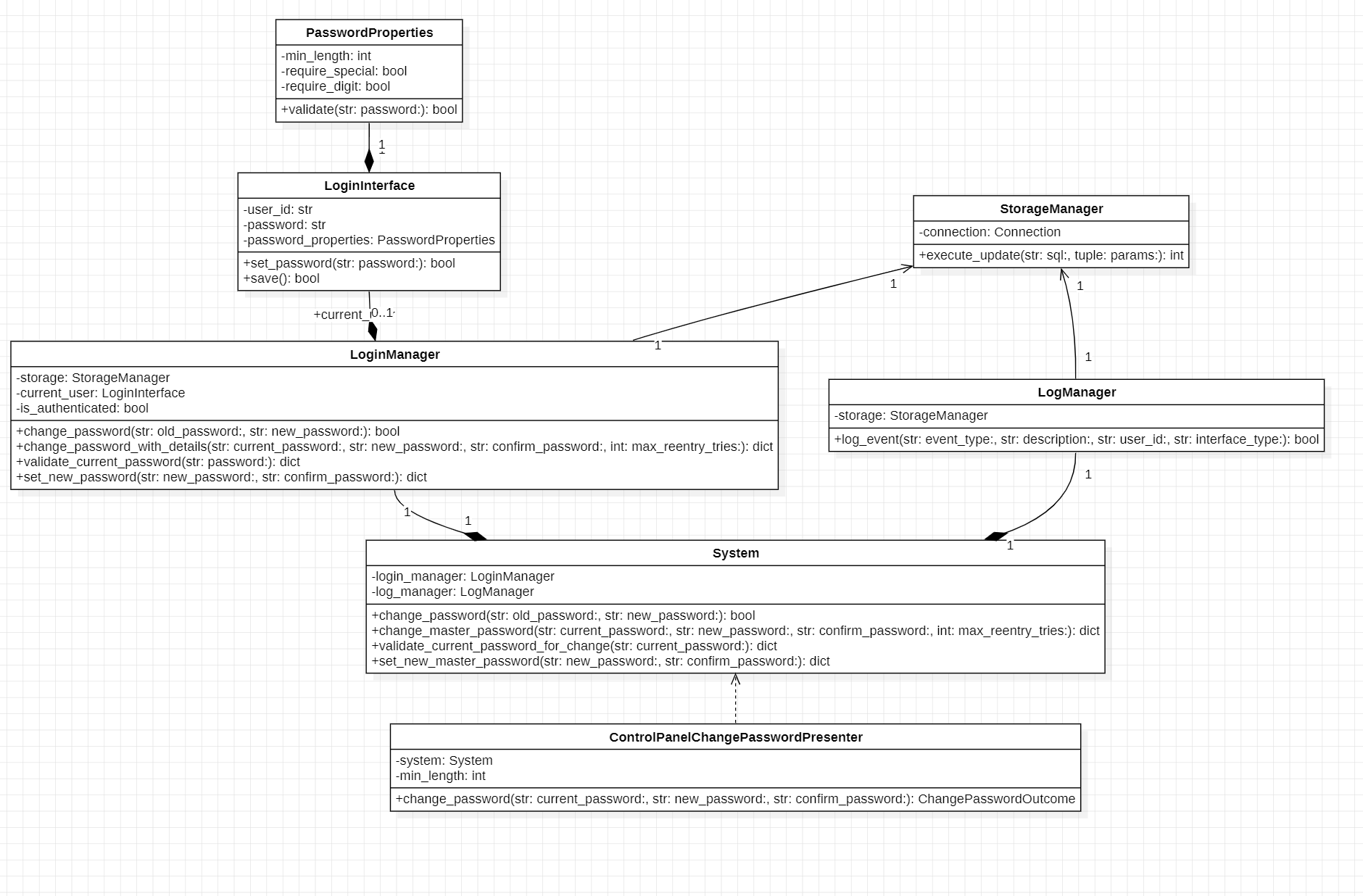
## 2. Configure Feature Class Diagram



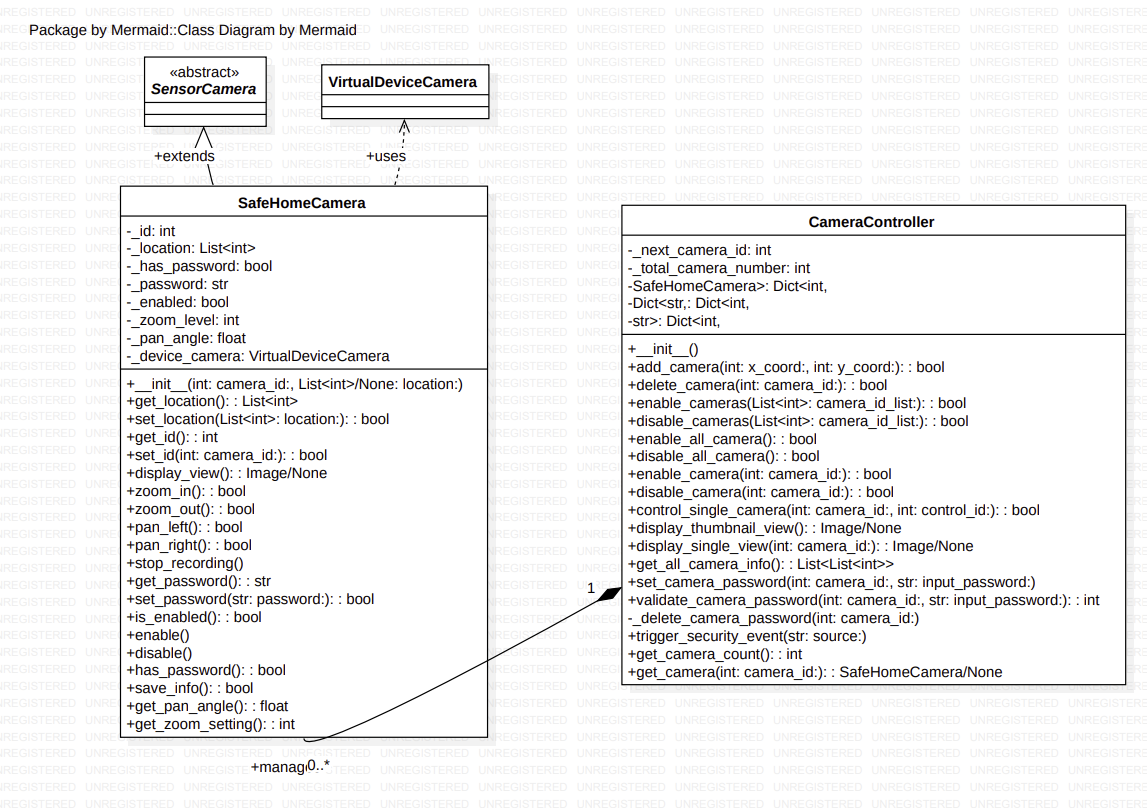
## 3. System On/Off/Reset Feature Class Diagram



## 4. Password Change Feature Class Diagram



5. Camera Management Class Diagram



# **III. Unit Tests**

## **1. External Communication Management**

### **A. Control Panel Management**

#### **1. ControlPanelLoginPresenter (Control Panel Login)**

##### **A. attempt\_login() (handling login from Control Panel)**

###### **a. test\_missing\_credentials\_triggers\_warning (UT-LP-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanelLoginPresenter |
| Method | attempt\_login() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that attempting login without providing both user ID and password triggers an appropriate warning message.

**Input Specifications**

• User ID: "" (empty string)

• Password: "pw" (any value)

**Expected Result**

LoginOutcome(  
 success=False,  
 alert\_level="warning",  
 alert\_title="Input Required",  
 alert\_message="Please enter both User ID and Password"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns failure outcome with warning level

• Pass: Alert message requests both credentials

• Pass: No system call is made

**Comment**

Input validation at the presenter layer prevents unnecessary system calls. Part of the Presenter pattern implementation introduced in Phase 2.

###### **b. test\_success\_navigates\_to\_main\_menu (UT-LP-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanelLoginPresenter |
| Method | attempt\_login() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that successful login results in navigation to the MainMenu screen.

**Input Specifications**

• User ID: "user"

• Password: "pw"

• System returns: `{"success": True}`

**Expected Result**

LoginOutcome(  
 success=True,  
 navigate\_to="MainMenu"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns success outcome

• Pass: navigate\_to field is set to "MainMenu"

**Comment**

Presenter encapsulates navigation logic, separating UI flow from business logic.

###### **c. test\_locked\_with\_remaining\_time\_sets\_status (UT-LP-003)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanelLoginPresenter |
| Method | attempt\_login() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that when an account is locked with remaining time, the presenter formats an appropriate user-friendly message with time remaining.

**Input Specifications**

• System returns: `{"locked": True, "remaining\_time": 90, "message": "Locked due to attempts"}`

**Expected Result**

LoginOutcome(  
 success=False,  
 status\_text="Account locked. Try again in 1 minutes 30 seconds",  
 status\_color="red",  
 alert\_level="error",  
 alert\_title="Account Locked"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Remaining time is formatted as minutes and seconds

• Pass: Status color is red

• Pass: Alert level is error

**Comment**

Presenter handles time formatting for user-friendly display. Implements time-based unlock feedback.

###### **d. test\_general\_failure\_with\_attempts\_shows\_message (UT-LP-004)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanelLoginPresenter |
| Method | attempt\_login() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that failed login attempts display remaining attempts to the user.

**Input Specifications**

• System returns: `{"success": False, "tries": 2, "remaining": 1}`

**Expected Result**

LoginOutcome(  
 success=False,  
 status\_text="Login failed. Remaining attempts: 1",  
 status\_color="red",  
 alert\_level="error",  
 alert\_title="Login Failed",  
 alert\_message="Incorrect password.\n\nFailed attempts: 2\nRemaining attempts: 1"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Status text shows remaining attempts

• Pass: Alert message includes both failed and remaining counts

**Comment**

User feedback for failed login attempts helps prevent account lockout through awareness.

#### **2. ControlPanelSettingsPresenter (Control Panel Settings Configuration)**

##### **A. save\_settings() (handling settings save from Control Panel)**

###### **a. test\_rejects\_non\_numeric\_values (UT-SP-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanelSettingsPresenter |
| Method | save\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that the presenter rejects non-numeric values for time fields.

**Input Specifications**

• monitoring\_phone: "911"

• homeowner\_phone: "010"

• lock\_time: "abc" (invalid)

• alarm\_delay: "5"

**Expected Result**

SettingsOutcome(  
 success=False,  
 alert\_level="error",  
 message="Invalid input. Please enter valid numbers for time fields."  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns error outcome

• Pass: Message indicates invalid input

**Comment**

Input validation at presenter layer prevents invalid data from reaching business logic.

###### **b. test\_rejects\_negative\_values (UT-SP-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanelSettingsPresenter |
| Method | save\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that the presenter rejects negative values for timing fields.

**Input Specifications**

• monitoring\_phone: "911"

• homeowner\_phone: "010"

• lock\_time: "-1" (negative)

• alarm\_delay: "0"

**Expected Result**

SettingsOutcome(  
 success=False,  
 alert\_level="error",  
 message="Timing values cannot be negative."  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns error outcome for negative values

• Pass: Appropriate validation message

**Comment**

Business rule: Timing values must be non-negative.

###### **c. test\_successful\_save\_invokes\_system (UT-SP-003)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanelSettingsPresenter |
| Method | save\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that valid settings are successfully passed to the system layer.

**Input Specifications**

• monitoring\_phone: "911"

• homeowner\_phone: "010"

• lock\_time: "30"

• alarm\_delay: "60"

**Expected Result**

SettingsOutcome(  
 success=True,  
 alert\_level="info",  
 message="System settings saved successfully!"  
 )

System receives:

{  
 'monitoring\_phone': '911',  
 'homeowner\_phone': '010',  
 'lock\_time': 30,  
 'alarm\_delay': 60  
 }

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns success outcome

• Pass: System method called with correct parameters

• Pass: Integer conversion performed correctly

**Comment**

Presenter handles type conversion from string to integer for time fields.

#### **3. ControlPanelChangePasswordPresenter (Control Panel Password Management)**

##### **A. change\_password() (handling password change from Control Panel)**

###### **a. test\_mismatched\_passwords\_return\_error (UT-CPP-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanelChangePasswordPresenter |
| Method | change\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that presenter-level validation catches mismatched passwords before calling system.

**Input Specifications**

• current\_password: "old"

• new\_password: "new-1"

• confirm\_password: "new-2" (mismatch)

**Expected Result**

ChangePasswordOutcome(  
 success=False,  
 message="New passwords do not match!",  
 alert\_level="error"  
 )

• System method is NOT called

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns error outcome

• Pass: No system call made

**Comment**

Early validation at presenter layer prevents unnecessary system calls. Test file: test\_change\_password\_presenter.py:21

###### **b. test\_short\_password\_rejected (UT-CPP-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanelChangePasswordPresenter |
| Method | change\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that presenter validates minimum password length.

**Input Specifications**

• min\_length: 4 (default)

• new\_password: "123" (only 3 characters)

• confirm\_password: "123"

**Expected Result**

ChangePasswordOutcome(  
 success=False,  
 message="Password must be at least 4 characters long.",  
 alert\_level="error"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns error for short password

• Pass: No system call made

**Comment**

Password policy enforcement at presenter layer. Test file: test\_change\_password\_presenter.py:28

###### **c. test\_successful\_change\_returns\_info (UT-CPP-003)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanelChangePasswordPresenter |
| Method | change\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that successful password change through presenter returns appropriate outcome.

**Input Specifications**

• current\_password: "old"

• new\_password: "1234"

• confirm\_password: "1234"

• System returns: True

**Expected Result**

ChangePasswordOutcome(  
 success=True,  
 message="Password changed successfully!",  
 alert\_level="info"  
 )

• System called with: ("old", "1234")

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns success outcome with info level

• Pass: System method called correctly

**Comment**

Presenter translates boolean system response into structured outcome. Test file: test\_change\_password\_presenter.py:35

#### **4. ControlPanelResetPresenter (Control Panel System Reset)**

##### **A. perform\_reset() (handling system reset from Control Panel)**

###### **a. test\_perform\_reset\_success (UT-RP-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanelResetPresenter |
| Method | perform\_reset() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that the presenter correctly handles successful system reset.

**Input Specifications**

• System reset returns: `{"success": True, "message": "Reset complete", "phase": 3}`

**Expected Result**

ResetOutcome(  
 success=True,  
 alert\_level="info",  
 alert\_title="Success",  
 alert\_message="Reset complete",  
 phase=3  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns success outcome

• Pass: Alert level is info

• Pass: Phase information preserved

**Comment**

Presenter translates system response into UI-appropriate format.

### **B. Web Interface Management**

#### **1. WebLoginPresenter (Web Browser Login - 2FA)**

##### **A. validate\_first\_password() (handling first password validation from Web)**

###### **a. test\_first\_password\_missing\_username (UT-WL-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter |
| Method | validate\_first\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that attempting to validate first password without username triggers validation error.

**Input Specifications**

• username: "" (empty string)

• password: "password123"

**Expected Result**

WebLoginOutcome(  
 success=False,  
 message="Username and password are required"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns failure outcome

• Pass: Appropriate error message

**Comment**

Input validation at the presenter layer prevents unnecessary system calls. Web interface uses 2-factor authentication (first password + second password/PIN).

###### **b. test\_first\_password\_missing\_password (UT-WL-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter |
| Method | validate\_first\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that attempting to validate first password without password triggers validation error.

**Input Specifications**

• username: "admin"

• password: "" (empty string)

**Expected Result**

WebLoginOutcome(  
 success=False,  
 message="Username and password are required"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns failure outcome

• Pass: Appropriate error message

**Comment**

Prevents submission of incomplete credentials.

###### **c. test\_first\_password\_success (UT-WL-003)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter |
| Method | validate\_first\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that successful first password validation returns success.

**Input Specifications**

• username: "admin"

• password: "password123"

• System returns: `{"success": True}`

**Expected Result**

WebLoginOutcome(  
 success=True,  
 message="First password correct"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns success outcome

• Pass: User proceeds to second password input

**Comment**

First step of 2-factor authentication for web interface.

###### **d. test\_first\_password\_locked\_with\_remaining\_time (UT-WL-004)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter |
| Method | validate\_first\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that locked account with remaining time returns appropriate information.

**Input Specifications**

• username: "admin"

• password: "wrong"

• Account locked with remaining\_time: 30 seconds

**Expected Result**

WebLoginOutcome(  
 success=False,  
 locked=True,  
 remaining\_time=30,  
 tries=5,  
 message="Account locked due to too many attempts"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns locked status

• Pass: Includes remaining time information

**Comment**

Time-based account locking for security.

###### **e. test\_first\_password\_system\_off (UT-WL-005)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter |
| Method | validate\_first\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that first password validation fails when system is OFF.

**Input Specifications**

• System state: OFF

• Valid credentials provided

**Expected Result**

WebLoginOutcome(  
 success=False,  
 system\_off=True,  
 message="System is currently off"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns failure with system\_off flag

• Pass: Appropriate error message

**Comment**

Precondition: System must be ON for login.

###### **f. test\_first\_password\_system\_locked (UT-WL-006)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter |
| Method | validate\_first\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that first password validation fails when system is LOCKED.

**Input Specifications**

• System state: LOCKED

• Valid credentials provided

**Expected Result**

WebLoginOutcome(  
 success=False,  
 system\_locked=True,  
 message="System is locked by administrator"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns failure with system\_locked flag

**Comment**

System-level lock prevents all logins.

###### **g. test\_first\_password\_failed\_with\_attempts (UT-WL-007)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter |
| Method | validate\_first\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that failed first password validation includes attempt information.

**Input Specifications**

• username: "admin"

• password: "wrong"

• tries: 2, remaining: 3

**Expected Result**

WebLoginOutcome(  
 success=False,  
 message="Incorrect first password",  
 tries=2,  
 remaining=3  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns failure with attempt counts

**Comment**

User feedback helps prevent account lockout.

###### **h. test\_first\_password\_system\_not\_available (UT-WL-008)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter |
| Method | validate\_first\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies proper error handling when system is not available.

**Input Specifications**

• system: None

**Expected Result**

WebLoginOutcome(  
 success=False,  
 message="System not available"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns appropriate error

**Comment**

Graceful degradation when system unavailable.

##### **B. validate\_second\_password() (handling second password/PIN validation from Web)**

###### **a. test\_second\_password\_missing (UT-WL2-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter |
| Method | validate\_second\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that empty second password triggers validation error.

**Input Specifications**

• username: "admin"

• second\_password: "" (empty string)

**Expected Result**

WebLoginOutcome(  
 success=False,  
 message="Second password is required"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns failure outcome

**Comment**

Second factor of 2FA must be provided.

###### **b. test\_second\_password\_success (UT-WL2-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter |
| Method | validate\_second\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that successful second password validation completes login.

**Input Specifications**

• username: "admin"

• second\_password: "pin1234"

• System returns: `{"success": True}`

**Expected Result**

WebLoginOutcome(  
 success=True,  
 message="Login successful",  
 redirect\_url="/dashboard"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns success with redirect URL

**Comment**

Completes 2-factor authentication and redirects to dashboard.

###### **c. test\_second\_password\_locked\_with\_remaining\_time (UT-WL2-003)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter |
| Method | validate\_second\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies handling of locked account during second password validation.

**Input Specifications**

• Account locked with remaining\_time: 45 seconds

**Expected Result**

WebLoginOutcome(  
 success=False,  
 locked=True,  
 remaining\_time=45,  
 tries=5  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns locked status with remaining time

**Comment**

Account can be locked at either authentication stage.

###### **d. test\_second\_password\_system\_off (UT-WL2-004)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter |
| Method | validate\_second\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that second password validation fails when system is OFF.

**Input Specifications**

• System state: OFF

**Expected Result**

WebLoginOutcome(  
 success=False,  
 system\_off=True  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns failure with system\_off flag

**Comment**

System state checked at each authentication stage.

###### **e. test\_second\_password\_system\_locked (UT-WL2-005)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter |
| Method | validate\_second\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that second password validation fails when system is LOCKED.

**Input Specifications**

• System state: LOCKED

**Expected Result**

WebLoginOutcome(  
 success=False,  
 system\_locked=True  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns failure with system\_locked flag

**Comment**

System-level lock enforced at all authentication stages.

###### **f. test\_second\_password\_failed\_with\_attempts (UT-WL2-006)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter |
| Method | validate\_second\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies failed second password validation includes attempt information.

**Input Specifications**

• second\_password: "wrong"

• tries: 1, remaining: 4

**Expected Result**

WebLoginOutcome(  
 success=False,  
 message="Incorrect second password",  
 tries=1,  
 remaining=4  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns failure with attempt counts

**Comment**

Separate attempt tracking for each authentication factor.

###### **g. test\_second\_password\_system\_not\_available (UT-WL2-007)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter |
| Method | validate\_second\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies error handling when system unavailable during second password validation.

**Input Specifications**

• system: None

**Expected Result**

WebLoginOutcome(  
 success=False,  
 message="System not available"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns appropriate error

**Comment**

Consistent error handling across both authentication stages.

###### **h. test\_calls\_login\_manager\_with\_correct\_interface (UT-WL2-008)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter |
| Method | validate\_first\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that presenter calls LoginManager with 'web\_browser' interface.

**Input Specifications**

• username: "admin"

• password: "password123"

**Expected Result**

• LoginManager.validate\_first\_password called with interface='web\_browser'

**Actual Result (Pass/Fail/Exception)**

• Pass: Correct interface parameter passed

**Comment**

Ensures proper logging and tracking of web-based logins. Test file: test\_web\_login\_presenter.py:162

#### **2. WebSettingsPresenter (Web Browser Settings Configuration)**

##### **A. update\_settings() (handling settings update from Web)**

###### **a. test\_invalid\_monitoring\_phone\_format (UT-WS-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that invalid monitoring phone format is rejected.

**Input Specifications**

• monitoring\_service\_phone: "invalid"

• homeowner\_phone: "123-456-7890"

**Expected Result**

WebSettingsOutcome(  
 success=False,  
 message="Validation failed",  
 errors=["Invalid monitoring service phone format (recommended: XXX-XXX-XXXX)"]  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns validation error

**Comment**

Phone number format validation at presenter layer.

###### **b. test\_invalid\_homeowner\_phone\_format (UT-WS-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that invalid homeowner phone format is rejected.

**Input Specifications**

• monitoring\_service\_phone: "123-456-7890"

• homeowner\_phone: "abc"

**Expected Result**

WebSettingsOutcome(  
 success=False,  
 message="Validation failed",  
 errors=["Invalid homeowner phone format (recommended: XXX-XXX-XXXX)"]  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns validation error

**Comment**

Consistent phone validation for both fields.

###### **c. test\_valid\_phone\_formats\_accepted (UT-WS-003)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that valid phone formats are accepted (standard and flexible).

**Input Specifications**

• Standard format: "111-222-3333"

• Flexible format: "(123) 456-7890", "+1-234-567-8900"

**Expected Result**

• Both formats accepted successfully

**Actual Result (Pass/Fail/Exception)**

• Pass: Standard XXX-XXX-XXXX format accepted

• Pass: Flexible format with at least 3 chars accepted

**Comment**

Flexible validation supports international formats while maintaining basic validation.

###### **d. test\_zero\_lock\_time\_rejected (UT-WS-004)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that zero lock time is rejected.

**Input Specifications**

• system\_lock\_time: 0

**Expected Result**

WebSettingsOutcome(  
 success=False,  
 errors=["System lock time must be greater than 0"]  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns validation error

**Comment**

Business rule: Lock time must be positive.

###### **e. test\_negative\_lock\_time\_rejected (UT-WS-005)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that negative lock time is rejected.

**Input Specifications**

• system\_lock\_time: -5

**Expected Result**

WebSettingsOutcome(  
 success=False,  
 errors=["System lock time must be greater than 0"]  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns validation error

**Comment**

Prevents invalid negative values.

###### **f. test\_negative\_alarm\_delay\_rejected (UT-WS-006)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that negative alarm delay is rejected.

**Input Specifications**

• alarm\_delay\_time: -10

**Expected Result**

WebSettingsOutcome(  
 success=False,  
 errors=["Alarm delay time must be non-negative"]  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns validation error

**Comment**

Alarm delay can be 0 (immediate) but not negative.

###### **g. test\_invalid\_lock\_time\_type (UT-WS-007)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that invalid lock time type is rejected.

**Input Specifications**

• system\_lock\_time: "abc"

**Expected Result**

WebSettingsOutcome(  
 success=False,  
 errors=["System lock time must be a valid number"]  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns type validation error

**Comment**

Type checking prevents non-numeric values.

###### **h. test\_invalid\_alarm\_delay\_type (UT-WS-008)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that invalid alarm delay type is rejected.

**Input Specifications**

• alarm\_delay\_time: "xyz"

**Expected Result**

WebSettingsOutcome(  
 success=False,  
 errors=["Alarm delay time must be a valid number"]  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns type validation error

**Comment**

Consistent type validation for time fields.

###### **i. test\_multiple\_validation\_errors (UT-WS-009)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that multiple validation errors are collected.

**Input Specifications**

• monitoring\_service\_phone: "invalid"

• homeowner\_phone: "bad"

• system\_lock\_time: 0

• alarm\_delay\_time: -5

**Expected Result**

• At least 4 validation errors returned

**Actual Result (Pass/Fail/Exception)**

• Pass: All validation errors collected

**Comment**

Comprehensive validation feedback improves user experience.

###### **j. test\_successful\_settings\_update (UT-WS-010)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies successful settings update with valid inputs.

**Input Specifications**

• monitoring\_service\_phone: "111-222-3333"

• homeowner\_phone: "444-555-6666"

• system\_lock\_time: 30

• alarm\_delay\_time: 60

**Expected Result**

WebSettingsOutcome(  
 success=True,  
 message="Settings updated successfully"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns success outcome

• Pass: Settings persisted to database

**Comment**

Complete update flow with all parameters.

###### **k. test\_partial\_update\_only\_phones (UT-WS-011)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies partial update of only phone numbers.

**Input Specifications**

• monitoring\_service\_phone: "123-456-7890"

• homeowner\_phone: "098-765-4321"

• (time values not provided)

**Expected Result**

• Update succeeds with only phones modified

**Actual Result (Pass/Fail/Exception)**

• Pass: Partial update succeeds

• Pass: Only phone fields updated

**Comment**

Supports selective field updates.

###### **l. test\_partial\_update\_only\_times (UT-WS-012)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies partial update of only time values.

**Input Specifications**

• system\_lock\_time: 45

• alarm\_delay\_time: 90

• (phone numbers not provided)

**Expected Result**

• Update succeeds with only times modified

**Actual Result (Pass/Fail/Exception)**

• Pass: Partial update succeeds

• Pass: Only time fields updated

**Comment**

Flexible update mechanism for different field combinations.

###### **m. test\_update\_converts\_string\_times\_to\_int (UT-WS-013)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that string time values are converted to integers.

**Input Specifications**

• system\_lock\_time: "30" (string)

• alarm\_delay\_time: "60" (string)

**Expected Result**

• Values converted to integers: 30, 60

**Actual Result (Pass/Fail/Exception)**

• Pass: String to int conversion successful

• Pass: Values stored as integers

**Comment**

Type coercion handles web form data (strings).

###### **n. test\_save\_failure\_returns\_error (UT-WS-014)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies handling of database save failure.

**Input Specifications**

• Valid inputs provided

• Database save fails

**Expected Result**

WebSettingsOutcome(  
 success=False,  
 message="Failed to save settings"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns failure outcome

**Comment**

Graceful error handling for persistence failures.

###### **o. test\_system\_not\_available (UT-WS-015)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies error handling when system is not available.

**Input Specifications**

• system: None

**Expected Result**

WebSettingsOutcome(  
 success=False,  
 message="System not available"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns appropriate error

**Comment**

System availability check prevents null reference errors.

###### **p. test\_get\_settings\_system\_not\_available (UT-WS-016)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | get\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies get\_settings error handling when system unavailable.

**Input Specifications**

• system: None

**Expected Result**

WebSettingsOutcome(  
 success=False,  
 message="System not available"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns appropriate error

**Comment**

Consistent error handling across all operations.

###### **q. test\_get\_settings\_success (UT-WS-017)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | get\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies successful retrieval of current settings.

**Input Specifications**

• System and ConfigurationManager available

**Expected Result**

WebSettingsOutcome(  
 success=True,  
 message="Settings retrieved successfully"  
 )

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns success outcome

**Comment**

Retrieval operation for GET requests.

###### **r. test\_empty\_update\_succeeds (UT-WS-018)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that update with no parameters succeeds.

**Input Specifications**

• No parameters provided

**Expected Result**

• Operation succeeds (no-op)

**Actual Result (Pass/Fail/Exception)**

• Pass: Empty update succeeds

**Comment**

Handles edge case of empty update request.

###### **s. test\_zero\_alarm\_delay\_accepted (UT-WS-019)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter |
| Method | update\_settings() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that zero alarm delay is accepted (immediate alarm).

**Input Specifications**

• alarm\_delay\_time: 0

**Expected Result**

• Update succeeds

**Actual Result (Pass/Fail/Exception)**

• Pass: Zero alarm delay accepted

**Comment**

Zero is valid for alarm delay (immediate activation), unlike lock time which must be positive.

## **2. Surveillance**

### A. Camera Management

#### 1. SafeHomeCamera

##### A. get\_location()

###### a. test\_get\_location (UT-CAM-LOC-CPY)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | get\_location() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling get\_location() returns a defensive copy of the location list, ensuring the camera's internal state is protected from external modifications."

Input Specifications

"Initial location: [100, 200]. Action: Modify the returned list value."

**Detailed Step**

1. Call get\_location() and store the list.
2. Modify the list (e.g., change index 0 to 999).
3. Call get\_location() again to assert the internal value remains [100, 200].

Expected Result

"The internal location must remain [100, 200], verifying that a copy was returned."

Actual Result (Pass/Fail/Exception)

Pass: Internal state was preserved.

Comment (including references)

Verification of defensive programming based on the Object State Protection Principle.

###### b. test\_get\_location\_returns\_copy (UT-CAM-LOC-CPY-ENF)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | get\_location() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that appending elements to the returned list does not affect the original list's length, reinforcing the defensive copy check."

Input Specifications

"Initial location: [100, 200]. Action: Append 999 to the returned list."

**Detailed Step**

1. Call get\_location() and append a new element to the returned list.
2. Call get\_location() again and assert the returned list's length is still 2.

Expected Result

The original list length must remain 2.

Actual Result (Pass/Fail/Exception)

Pass: Defensive copy prevented unauthorized length modification.

Comment (including references)

Coverage reinforcement for the defensive copy check.

###### c. test\_initialization\_with\_location (UT-CAM-INIT-LOC)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | \_\_init\_\_(), get\_location() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that when the object is initialized with a specific location, get\_location() correctly retrieves the set coordinates."

Input Specifications

"Instantiate SafeHomeCamera with location=[100, 200]."

**Detailed Step**

1. Instantiate the object with the specified location.
2. Call get\_location() and assert the returned value.

Expected Result

"get\_location() must return exactly [100, 200]."

Actual Result (Pass/Fail/Exception)

Pass: Initial location was correctly set and retrieved.

Comment (including references)

Validation of initial attribute setting.

###### d. test\_initialization\_without\_location (UT-CAM-INIT-DEF)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | \_\_init\_\_(), get\_location() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that when the object is initialized without specifying a location, get\_location() correctly returns the default coordinates [0, 0]."

Input Specifications

Instantiate SafeHomeCamera omitting the location parameter.

**Detailed Step**

1. Instantiate the object without the location.
2. Call get\_location() and assert the returned value.

Expected Result

"get\_location() must return the default location [0, 0]."

Actual Result (Pass/Fail/Exception)

"Pass: Default location [0, 0] was correctly assigned."

Comment (including references)

Verification of default value handling.

##### B. set\_location()

###### a. test\_set\_location (UT-CAM-LOC-SET)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | set\_location(), get\_location() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling set\_location() with valid coordinates ([x, y]) successfully updates the location, returns True, and the new value is retrieved by get\_location()."

Input Specifications

"Initial location: [100, 200]. Action: Call set\_location([300, 400])."

**Detailed Step**

1. Call set\_location([300, 400]).
2. Assert the return value is True.
3. Call get\_location() and assert the new value [300, 400].

Expected Result

"set\_location() must return True, and get\_location() must return [300, 400]."

Actual Result (Pass/Fail/Exception)

Pass: Location was successfully updated.

Comment (including references)

Validation of the camera coordinate setting functionality.

###### b. test\_set\_invalid\_location (UT-CAM-LOC-INV-BAS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | set\_location(), get\_location() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that an invalid location format (specifically, missing the Y-coordinate) is rejected, returns False, and preserves the existing location."

Input Specifications

"Initial location: [100, 200]. Action: Call set\_location([100]) (invalid format)."

**Detailed Step**

1. Call set\_location([100]).
2. Assert the return value is False.
3. Call get\_location() and assert the location remains [100, 200].

Expected Result

"set\_location() must return False, and get\_location() must remain [100, 200]."

Actual Result (Pass/Fail/Exception)

"Pass: Invalid input was correctly rejected, and state was preserved."

Comment (including references)

Validation of input validation and error handling logic.

###### c. test\_set\_location\_invalid\_lengths (UT-CAM-LOC-INV-COV)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | set\_location(), get\_location() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that the input validation handles various incorrect list lengths (e.g., 0, 1, or 3 elements), rejecting them and ensuring the current location is unchanged."

Input Specifications

"Initial location: [100, 200]. Action: Call set\_location with [], [100], and [100, 200, 300] sequentially."

**Detailed Step**

1. Iterate through invalid inputs.
2. For each, assert set\_location returns False.
3. Assert get\_location() remains [100, 200] after each attempt.

Expected Result

"set\_location() must return False for all invalid inputs, and the location must remain [100, 200]."

Actual Result (Pass/Fail/Exception)

Pass: All invalid list lengths were correctly handled and rejected.

Comment (including references)

Coverage test ensuring validation handles various incorrect list lengths.

##### C. get\_id()

###### a. test\_get\_id (UT-CAM-ID-GET)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | get\_id() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that the camera's unique ID, assigned during initialization, is correctly retrieved."

Input Specifications

Initial state: Camera ID is 1 (set by fixture/initialization).

**Detailed Step**

1. Instantiate the camera object.
2. Call get\_id().
3. Assert the returned ID matches the initialized value.

Expected Result

get\_id() must return 1.

Actual Result (Pass/Fail/Exception)

Pass: ID was correctly retrieved.

Comment (including references)

Basic getter functionality check for the camera's static identifier.

###### b. test\_initialization\_with\_location (UT-CAM-INIT-ID)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | \_\_init\_\_(), get\_id() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Verifies that the camera ID (1) is correctly set and immediately retrieved after object initialization.

Input Specifications

Instantiate SafeHomeCamera with camera\_id=1.

**Detailed Step**

1. Instantiate the object with the specified ID.
2. Call get\_id() and assert the returned value.

Expected Result

get\_id() must return 1.

Actual Result (Pass/Fail/Exception)

Pass: ID was correctly set during initialization.

Comment (including references)

Fundamental initial state check for ID consistency.

##### D. set\_id()

###### a. test\_set\_id (UT-CAM-ID-SET-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | set\_id(), get\_id() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Verifies that calling set\_id() with a valid positive integer successfully updates the camera's ID and is correctly retrieved by get\_id().

Input Specifications

Initial ID: 1. Action: Call set\_id(5).

**Detailed Step**

1. Call set\_id(5).
2. Assert the return value is True.
3. Call get\_id() and assert the new ID is 5.

Expected Result

"set\_id() must return True, and get\_id() must return 5."

Actual Result (Pass/Fail/Exception)

Pass: ID was successfully changed and retrieved.

Comment (including references)

Verification of successful ID mutation and retrieval.

###### b. test\_set\_invalid\_id (UT-CAM-ID-SET-FAIL)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | set\_id(), get\_id() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling set\_id() with non-positive integers (0 or negative) is rejected, returns False, and preserves the original ID."

Input Specifications

Initial ID: 1. Action: Call set\_id(0) and then set\_id(-1).

**Detailed Step**

1. Call set\_id(0).
2. Assert the return value is False.
3. Call get\_id() and assert the ID remains the initial value (1).

Expected Result

"set\_id() must return False for non-positive inputs, and get\_id() must return 1."

Actual Result (Pass/Fail/Exception)

"Pass: Invalid ID attempts were rejected, and state was preserved."

Comment (including references)

Validation of ID input validation (ID must be a positive integer) and state preservation.

###### c. test\_set\_id\_boundary\_values (UT-CAM-ID-BOUND)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | set\_id(), get\_id() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the ID setter correctly handles valid boundary values, specifically the minimum positive integer (1) and a large valid ID (99999)."

Input Specifications

Initial state ID: 1. Action: Call set\_id(1) and then set\_id(99999).

**Detailed Step**

1. Call set\_id(1) and assert result is True.
2. Call set\_id(99999) and assert result is True.
3. Verify get\_id() reflects the final value 99999.

Expected Result

"set\_id() must return True for both valid boundary attempts, and the final get\_id() must return 99999."

Actual Result (Pass/Fail/Exception)

Pass: Valid ID boundaries were correctly accepted.

Comment (including references)

Comprehensive check of valid ID boundaries for the set\_id method.

##### E. display\_view()

###### a. test\_display\_view (UT-CAM-VIEW-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | display\_view() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that when the camera is enabled, display\_view() successfully returns a non-null object, confirming the feed is accessible."

Input Specifications

Initial state: Camera is enabled (is\_enabled() is True).

**Detailed Step**

1. Ensure the camera is enabled.
2. Call display\_view().
3. Assert the result is not None and is an instance of PIL.Image.Image.

Expected Result

Returns a valid PIL.Image.Image object.

Actual Result (Pass/Fail/Exception)

Pass: View object was successfully retrieved.

Comment (including references)

Core function test for accessing the camera feed.

###### b. test\_display\_view\_disabled\_camera (UT-CAM-VIEW-FAIL)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | display\_view(), disable() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that when the camera is disabled, display\_view() enforces the security policy by returning None, blocking access to the feed."

Input Specifications

Action: camera.disable() is called to set the state to disabled.

**Detailed Step**

1. Call camera.disable().
2. Call display\_view().
3. Assert the result is None.

Expected Result

Returns None.

Actual Result (Pass/Fail/Exception)

Pass: Access to the view was correctly denied when disabled.

Comment (including references)

Security policy validation for access control in a disabled state.

###### c. test\_display\_view\_exception\_handling (UT-CAM-VIEW-EXC)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | display\_view() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the method's robustness by ensuring that if the underlying device camera throws an exception during view retrieval, the method handles it gracefully and returns None."

Input Specifications

"Mocking: The underlying device camera's get\_view() method is patched to raise an Exception(""Camera error"")."

**Detailed Step**

1. Patch get\_view() to raise an exception.
2. Call display\_view() within the patched context.
3. Assert the result is None.

Expected Result

"Returns None, and the system does not crash or raise the exception."

Actual Result (Pass/Fail/Exception)

Pass: External component failure was handled gracefully.

Comment (including references)

Robustness check for failures in the virtual device layer.

##### F. zoom\_in()

###### a. test\_zoom\_in (UT-CAM-ZOOM-IN-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | zoom\_in(), get\_zoom\_setting() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that when the current zoom level is below the maximum, calling zoom\_in() returns True and increases the zoom level by exactly 1."

Input Specifications

Initial state: Zoom level is 1. Action: Call zoom\_in() once.

**Detailed Step**

1. Call zoom\_in().
2. Assert the return value is True.
3. Call get\_zoom\_setting() and assert the new level is 2.

Expected Result

"zoom\_in() must return True, and the new zoom\_setting must be 2."

Actual Result (Pass/Fail/Exception)

Pass: Zoom level was successfully incremented.

Comment (including references)

Core function test for camera zoom operation.

###### b. test\_zoom\_in\_max (UT-CAM-ZOOM-IN-MAX)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | zoom\_in(), get\_zoom\_setting() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the boundary condition: when the maximum zoom level (10) is reached, subsequent calls to zoom\_in() fail, return False, and the level remains 10."

Input Specifications

"Action: Repeatedly call zoom\_in() until the level reaches 10, then call it one more time."

**Detailed Step**

1. Set zoom level to 10.
2. Call zoom\_in().
3. Assert the return value is False.
4. Assert get\_zoom\_setting() is still 10.

Expected Result

"zoom\_in() must return False , and the zoom level must remain 10."

Actual Result (Pass/Fail/Exception)

Pass: Zoom operation was correctly blocked at the maximum limit.

Comment (including references)

Boundary condition check for maximum zoom level.

###### c. test\_zoom\_in\_disabled\_camera (UT-CAM-ZOOM-IN-FAIL)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | zoom\_in(), disable(), get\_zoom\_setting() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling zoom\_in() on a disabled camera is rejected, returns False, and preserves the original zoom level."

Input Specifications

Initial state: Zoom level 1. Action: camera.disable() is called.

**Detailed Step**

1. Disable the camera.
2. Record the initial zoom level.
3. Call zoom\_in() and assert the return value is False.
4. Assert the zoom level has not changed.

Expected Result

"zoom\_in() must return False, and the zoom level must remain the initial value."

Actual Result (Pass/Fail/Exception)

Pass: Control command was correctly blocked in the disabled state.

Comment (including references)

Security and state awareness check for control operations.

###### d. test\_zoom\_in\_device\_camera\_integration (UT-CAM-ZOOM-IN-INT)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | zoom\_in() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling zoom\_in() successfully executes the underlying virtual device interaction logic and returns a boolean result, confirming proper integration."

Input Specifications

Initial state: Camera is enabled and below max zoom.

**Detailed Step**

1. Call zoom\_in().
2. Assert the return value is a boolean (True) and the internal logic path for device interaction is executed.

Expected Result

The method must execute the device interaction path and return True.

Actual Result (Pass/Fail/Exception)

Pass: Integration with the virtual device layer for zoom control was confirmed.

Comment (including references)

Verifies the linkage between the domain model and the virtual device simulator.

##### G. zoom\_out()

###### a. test\_zoom\_out (UT-CAM-ZOOM-OUT-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | zoom\_out(), get\_zoom\_setting() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that when the current zoom level is above the minimum, calling zoom\_out() returns True and decreases the zoom level by exactly 1."

Input Specifications

Initial state: Zoom level is set to 2 (after one zoom\_in call). Action: Call zoom\_out() once.

**Detailed Step**

1. Ensure zoom level is > 1.
2. Call zoom\_out().
3. Assert the return value is True.
4. Call get\_zoom\_setting() and assert the new level is 1.

Expected Result

"zoom\_out() must return True, and the new zoom\_setting must be 1."

Actual Result (Pass/Fail/Exception)

Pass: Zoom level was successfully decremented.

Comment (including references)

Core function test for camera zoom operation.

###### b. test\_zoom\_out\_min (UT-CAM-ZOOM-OUT-MIN)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | zoom\_out(), get\_zoom\_setting() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the boundary condition: when the minimum zoom level (1) is reached, subsequent calls to zoom\_out() fail, return False, and the level remains unchanged."

Input Specifications

Initial state: Zoom level is 1. Action: Call zoom\_out() once.

**Detailed Step**

1. Ensure zoom level is 1.
2. Call zoom\_out().
3. Assert the return value is False.
4. Assert get\_zoom\_setting() is still 1.

Expected Result

"zoom\_out() must return False , and the zoom level must remain 1."

Actual Result (Pass/Fail/Exception)

Pass: Zoom operation was correctly blocked at the minimum limit.

Comment (including references)

Boundary condition check for minimum zoom level.

###### c. test\_zoom\_out\_disabled\_camera (UT-CAM-ZOOM-OUT-FAIL)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | zoom\_out(), disable(), get\_zoom\_setting() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling zoom\_out() on a disabled camera is rejected, returns False, and preserves the original zoom level."

Input Specifications

Initial state: Zoom level 2. Action: camera.disable() is called.

**Detailed Step**

1. Disable the camera.
2. Record the initial zoom level.
3. Call zoom\_out() and assert the return value is False.
4. Assert the zoom level has not changed.

Expected Result

"zoom\_out() must return False, and the zoom level must remain the initial value."

Actual Result (Pass/Fail/Exception)

Pass: Control command was correctly blocked in the disabled state.

Comment (including references)

Security and state awareness check for control operations.

###### d. test\_zoom\_out\_device\_camera\_integration (UT-CAM-ZOOM-OUT-INT)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | zoom\_out() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling zoom\_out() correctly executes the underlying virtual device interaction logic and returns a boolean result, confirming proper integration."

Input Specifications

"Initial state: Camera is enabled and zoomed in (e.g., level 2)."

**Detailed Step**

1. Ensure zoom level > 1.
2. Call zoom\_out().
3. Assert the return value is a boolean (True) and the internal logic path for device interaction is executed.

Expected Result

The method must execute the device interaction path and return True.

Actual Result (Pass/Fail/Exception)

Pass: Integration with the virtual device layer for zoom control was confirmed.

Comment (including references)

Verifies the linkage between the domain model and the virtual device simulator.

##### H. pan\_left()

###### a. test\_pan\_left (UT-CAM-PAN-LEFT-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | pan\_left(), get\_pan\_angle() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that when the pan angle is above the minimum limit, calling pan\_left() returns True and decreases the angle by exactly 5.0 degrees."

Input Specifications

Initial state: Pan angle is 0.0 degrees. Action: Call pan\_left() once.

**Detailed Step**

1. Call pan\_left().
2. Assert the return value is True.
3. Call get\_pan\_angle() and assert the new angle is -5.0 degrees.

Expected Result

"pan\_left() must return True, and the new pan\_angle must be -5.0."

Actual Result (Pass/Fail/Exception)

Pass: Pan angle was successfully decremented.

Comment (including references)

Core function test for camera pan operation.

###### b. test\_pan\_left\_limit (UT-CAM-PAN-LEFT-MIN)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | pan\_left(), get\_pan\_angle() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the boundary condition: once the minimum pan angle (-180.0 degrees) is reached, subsequent calls to pan\_left() preserve the limit and the angle does not decrease further."

Input Specifications

"Action: Repeatedly call pan\_left() until the angle reaches -180.0 degrees, then call it one more time."

**Detailed Step**

1. Set pan angle to -180.0 degrees.
2. Call pan\_left().
3. Assert the return value is True.
4. Assert get\_pan\_angle() is still -180.0.

Expected Result

"pan\_left() must return True, and the angle must remain -180.0 degrees."

Actual Result (Pass/Fail/Exception)

Pass: Pan operation was correctly blocked at the minimum limit.

Comment (including references)

Boundary condition check for minimum pan angle.

###### c. test\_pan\_left\_disabled\_camera (UT-CAM-PAN-LEFT-FAIL)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | pan\_left(), disable(), get\_pan\_angle() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling pan\_left() on a disabled camera is rejected, returns False, and preserves the original angle."

Input Specifications

Initial state: Angle 0.0. Action: camera.disable() is called.

**Detailed Step**

1. Disable the camera.
2. Record the initial angle.
3. Call pan\_left() and assert the return value is False.
4. Assert the angle has not changed.

Expected Result

"pan\_left() must return False, and the angle must remain the initial value."

Actual Result (Pass/Fail/Exception)

Pass: Control command was correctly blocked in the disabled state.

Comment (including references)

Security and state awareness check for control operations.

###### d. test\_pan\_left\_device\_camera\_integration (UT-CAM-PAN-LEFT-INT)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | pan\_left() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling pan\_left() successfully executes the underlying virtual device interaction logic and returns a boolean result, confirming proper integration."

Input Specifications

Initial state: Camera is enabled and at 0.0 degrees.

**Detailed Step**

1. Call pan\_left().
2. Assert the return value is a boolean (True) and the internal logic path for device interaction is executed.

Expected Result

The method must execute the device interaction path and return True.

Actual Result (Pass/Fail/Exception)

Pass: Integration with the virtual device layer for pan control was confirmed.

Comment (including references)

Verifies the linkage between the domain model and the virtual device simulator.

##### I. pan\_right()

###### a. test\_pan\_right (UT-CAM-PAN-RIGHT-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | pan\_right(), get\_pan\_angle() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that when the pan angle is below the maximum limit, calling pan\_right() returns True and increases the angle by exactly 5.0 degrees."

Input Specifications

Initial state: Pan angle is 0.0 degrees. Action: Call pan\_right() once.

**Detailed Step**

1. Call pan\_right().
2. Assert the return value is True.
3. Call get\_pan\_angle() and assert the new angle is 5.0 degrees.

Expected Result

"pan\_right() must return True, and the new pan\_angle must be 5.0."

Actual Result (Pass/Fail/Exception)

Pass: Pan angle was successfully incremented.

Comment (including references)

Core function test for camera pan operation.

###### b. test\_pan\_right\_limit (UT-CAM-PAN-RIGHT-MAX)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | pan\_right(), get\_pan\_angle() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the boundary condition: once the maximum pan angle (+180.0 degrees) is reached, subsequent calls to pan\_right() preserve the limit and the angle does not increase further."

Input Specifications

"Action: Repeatedly call pan\_right() until the angle reaches 180.0 degrees, then call it one more time."

**Detailed Step**

1. Set pan angle to 180.0 degrees.
2. Call pan\_right().
3. Assert the return value is True.
4. Assert get\_pan\_angle() is still 180.0.

Expected Result

"pan\_right() must return True, and the angle must remain 180.0 degrees."

Actual Result (Pass/Fail/Exception)

Pass: Pan operation was correctly blocked at the maximum limit.

Comment (including references)

Boundary condition check for maximum pan angle.

###### c. test\_pan\_right\_disabled\_camera (UT-CAM-PAN-RIGHT-FAIL)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | pan\_right(), disable(), get\_pan\_angle() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling pan\_right() on a disabled camera is rejected, returns False, and preserves the original angle."

Input Specifications

Initial state: Angle 0.0. Action: camera.disable() is called.

**Detailed Step**

1. Disable the camera.
2. Record the initial angle.
3. Call pan\_right() and assert the return value is False.
4. Assert the angle has not changed.

Expected Result

"pan\_right() must return False, and the angle must remain the initial value."

Actual Result (Pass/Fail/Exception)

Pass: Control command was correctly blocked in the disabled state.

Comment (including references)

Security and state awareness check for control operations.

###### d. test\_pan\_right\_device\_camera\_integration (UT-CAM-PAN-RIGHT-INT)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | pan\_right() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling pan\_right() successfully executes the underlying virtual device interaction logic and returns a boolean result, confirming proper integration."

Input Specifications

Initial state: Camera is enabled and at 0.0 degrees.

**Detailed Step**

1. Call pan\_right().
2. Assert the return value is a boolean (True) and the internal logic path for device interaction is executed.

Expected Result

The method must execute the device interaction path and return True.

Actual Result (Pass/Fail/Exception)

Pass: Integration with the virtual device layer for pan control was confirmed.

Comment (including references)

Verifies the linkage between the domain model and the virtual device simulator.

##### J. get\_password()

###### a. test\_get\_password\_not\_set (UT-CAM-PW-GET-DEF)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | get\_password() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that when the camera password has not been initialized or set, get\_password() correctly returns an empty string ("""")."

Input Specifications

Initial state: No password set.

**Detailed Step**

1. Instantiate the camera object (default state).
2. Call get\_password().
3. Assert the returned value is "".

Expected Result

"get\_password() must return ""."

Actual Result (Pass/Fail/Exception)

Pass: Empty string was correctly retrieved in the default state.

Comment (including references)

Basic getter functionality check for the default password state.

###### b. test\_set\_password (UT-CAM-PW-SET-GET)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | set\_password(), get\_password() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that after a successful password setting (set\_password), get\_password() correctly returns the newly stored password."

Input Specifications

"Action: Call set\_password(""mypassword123"")."

**Detailed Step**

1. Call set\_password("mypassword123").
2. Assert the return value is True.
3. Call get\_password() and assert the value is "mypassword123".

Expected Result

"get\_password() must return ""mypassword123""."

Actual Result (Pass/Fail/Exception)

Pass: Password was successfully stored and retrieved.

Comment (including references)

Verification of password storage and retrieval functionality.

##### K. set\_password()

###### a. test\_set\_password (UT-CAM-PW-SET-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | set\_password(), has\_password(), get\_password() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling set\_password() with a valid password string returns True, updates the password internally, and sets the has\_password() flag to True."

Input Specifications

"Action: Call set\_password(""mypassword123"")."

**Detailed Step**

1. Call set\_password("mypassword123").
2. Assert the return value is True.
3. Assert has\_password() is True and get\_password() returns the set value.

Expected Result

"set\_password() must return True, and the new password must be successfully stored."

Actual Result (Pass/Fail/Exception)

Pass: Password was successfully set and validated.

Comment (including references)

Core function test for password storage and retrieval.

###### b. test\_set\_empty\_password (UT-CAM-PW-SET-EMPTY)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | set\_password(), has\_password() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that setting an empty string ("""") is rejected by set\_password(), which returns False and ensures the has\_password() state remains False."

Input Specifications

"Action: Call set\_password("""")."

**Detailed Step**

1. Call set\_password("").
2. Assert the return value is False.
3. Assert has\_password() is False and get\_password() is "".

Expected Result

"set\_password() must return False , and no password should be set."

Actual Result (Pass/Fail/Exception)

Pass: Empty password setting was correctly rejected.

Comment (including references)

Input validation check for minimum password length/content.

###### c. test\_set\_password\_special\_characters (UT-CAM-PW-SET-SPEC)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | set\_password() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that the system can successfully accept and store passwords containing various special characters and non-standard characters, confirming character set compatibility."

Input Specifications

"Action: Call set\_password with strings like ""!@#$%"", ""암호123"" ."

**Detailed Step**

1. Call set\_password with a special character string.
2. Assert the return value is True.
3. Assert get\_password() returns the exact string.

Expected Result

"set\_password() must return True, and the special character password must be saved correctly."

Actual Result (Pass/Fail/Exception)

Pass: Special character passwords were successfully accepted.

Comment (including references)

Coverage test for character set compatibility in password fields.

###### d. test\_set\_password\_long\_password (UT-CAM-PW-SET-LONG)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | set\_password() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the system's robustness by ensuring it can successfully accept and store a very long password (e.g., 1000 characters) without truncation or failure."

Input Specifications

Action: Call set\_password with a 1000-character long string.

**Detailed Step**

1. Create a long string.
2. Call set\_password() with the long string.
3. Assert the return value is True and get\_password() returns the full, intact long string.

Expected Result

"set\_password() must return True, and the long password must be saved correctly."

Actual Result (Pass/Fail/Exception)

Pass: Long password was successfully stored and retrieved.

Comment (including references)

Stress and robustness test for maximum password length constraints.

###### e. test\_set\_password\_whitespace\_only (UT-CAM-PW-SET-SPACE)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | set\_password(), get\_password() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies how the system handles passwords consisting only of whitespace characters, confirming the outcome (in this implementation, it is accepted as a valid, non-empty string)."

Input Specifications

"Action: Call set\_password("" "") (three spaces)."

**Detailed Step**

1. Call set\_password(" ").
2. Assert the return value is True.
3. Assert get\_password() returns the exact whitespace string.

Expected Result

"set\_password() must return True, and the password must be saved as "" ""."

Actual Result (Pass/Fail/Exception)

Pass: Whitespace-only string was accepted and stored as a valid password.

Comment (including references)

Detailed input validation check for whitespace content.

##### L. is\_enabled()

###### a. test\_is\_enabled (UT-CAM-STAT-FULL)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | is\_enabled(), disable(), enable() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that is\_enabled() accurately reflects the camera's status through a full operational cycle: checking the initial state, switching to disabled, and switching back to enabled."

Input Specifications

"Initial state: Enabled. Actions: Call disable(), then call enable()."

**Detailed Step**

1. Assert initial state is True.
2. Call disable() and assert state is False.
3. Call enable() and assert state is True.

Expected Result

"The sequence of is\_enabled() returns must be True, then False, then True."

Actual Result (Pass/Fail/Exception)

Pass: The camera state was correctly tracked throughout the cycle.

Comment (including references)

Core functionality test for state retrieval and status consistency.

###### b. test\_enable (UT-CAM-STAT-EN)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | enable(), is\_enabled() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling enable() successfully transitions the camera to the active state, as confirmed by is\_enabled()."

Input Specifications

Initial state: Camera is disabled (via disable() call). Action: Call enable().

**Detailed Step**

1. Set the initial state to disabled.
2. Call enable().
3. Assert is\_enabled() returns True.

Expected Result

is\_enabled() must return True after enable() is called.

Actual Result (Pass/Fail/Exception)

Pass: The camera successfully transitioned to the enabled state.

Comment (including references)

Verification of the enable setter's result.

###### c. test\_disable (UT-CAM-STAT-DIS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | disable(), is\_enabled() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling disable() successfully transitions the camera to the inactive state, as confirmed by is\_enabled()."

Input Specifications

Initial state: Camera is enabled. Action: Call disable().

**Detailed Step**

1. Assert initial state is True.
2. Call disable().
3. Assert is\_enabled() returns False.

Expected Result

is\_enabled() must return False after disable() is called.

Actual Result (Pass/Fail/Exception)

Pass: The camera successfully transitioned to the disabled state.

Comment (including references)

Verification of the disable setter's result.

###### d. test\_enable\_disable\_toggle (UT-CAM-STAT-TOG)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | is\_enabled(), enable(), disable() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the stability of the camera's enabled state by checking its status after multiple consecutive, redundant calls to enable() and disable()."

Input Specifications

"Actions: Multiple calls to disable() (3x), followed by multiple calls to enable() (3x)."

**Detailed Step**

1. Call disable() multiple times. Assert state is False.
2. Call enable() multiple times. Assert final state is True.

Expected Result

"The state should correctly switch and persist based on the last command, regardless of redundant calls."

Actual Result (Pass/Fail/Exception)

Pass: State toggling was stable across redundant commands.

Comment (including references)

Robustness check for state persistence across repeated setter calls.

##### M. enable()

###### a. test\_enable (UT-CAM-EN-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | enable(), is\_enabled() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling enable() successfully transitions the camera from the disabled state to the active state, as confirmed by is\_enabled()."

Input Specifications

Initial state: Camera is first disabled. Action: Call enable().

**Detailed Step**

1. Set the initial state to disabled.
2. Call enable().
3. Assert is\_enabled() returns True.

Expected Result

is\_enabled() must return True after enable() is called.

Actual Result (Pass/Fail/Exception)

Pass: The camera successfully transitioned to the enabled state.

Comment (including references)

Core function test for setting the camera to an active state.

###### b. test\_enable\_disable\_toggle (UT-CAM-EN-TOG)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | enable(), is\_enabled() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the stability of the enabled state by checking the status after multiple consecutive, redundant calls to enable()."

Input Specifications

Initial state: Camera is disabled. Action: Multiple (3x) consecutive calls to enable().

**Detailed Step**

1. Set the initial state to disabled.
2. Call enable() multiple times.
3. Assert is\_enabled() returns True.

Expected Result

"The state should consistently switch and persist as True, regardless of redundant calls."

Actual Result (Pass/Fail/Exception)

"Pass: State toggling was stable, and the enabled state persisted."

Comment (including references)

Robustness check for state persistence across repeated enable() calls.

##### N. disable()

###### a. test\_disable (UT-CAM-DIS-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | disable(), is\_enabled() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling disable() successfully transitions the camera from the active state to the inactive state, as confirmed by is\_enabled()."

Input Specifications

Initial state: Camera is enabled. Action: Call disable().

**Detailed Step**

1. Assert initial state is True.
2. Call disable().
3. Assert is\_enabled() returns False.

Expected Result

is\_enabled() must return False after disable() is called.

Actual Result (Pass/Fail/Exception)

Pass: The camera successfully transitioned to the disabled state.

Comment (including references)

Core function test for setting the camera to an inactive state.

###### b. test\_display\_view\_disabled\_camera (UT-CAM-DIS-CONSEQ)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | disable(), display\_view() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the direct consequence of the disabled state: after calling disable(), attempting to access the camera view (display\_view()) is correctly denied, returning None."

Input Specifications

Action: Call disable()., Consequence Check: Call display\_view().

**Detailed Step**

1. Call disable() to change the state.
2. Call display\_view().
3. Assert the result is None.

Expected Result

"display\_view() must return None , confirming view access is blocked in the disabled state."

Actual Result (Pass/Fail/Exception)

Pass: View access was correctly denied after disabling the camera.

Comment (including references)

Verification of the security consequence of the disabled state.

###### c. test\_enable\_disable\_toggle (UT-CAM-DIS-TOG)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | disable(), is\_enabled() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the stability of the disabled state by checking the status after multiple consecutive, redundant calls to disable()."

Input Specifications

Initial state: Camera is enabled. Action: Multiple (3x) consecutive calls to disable().

**Detailed Step**

1. Call disable() multiple times.
2. Assert is\_enabled() returns False.

Expected Result

"The state should consistently switch and persist as False, regardless of redundant calls."

Actual Result (Pass/Fail/Exception)

"Pass: State toggling was stable, and the disabled state persisted."

Comment (including references)

Robustness check for state persistence across repeated disable() calls.

##### O. has\_password()

###### a. test\_has\_password (UT-CAM-PW-STAT-FULL)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | has\_password(), set\_password() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that has\_password() accurately reflects the camera's password status: initially False, and switching to True after a valid password is set."

Input Specifications

"Initial state: No password set. Action: Call set\_password(""password123"")."

**Detailed Step**

1. Assert initial state is False.
2. Call set\_password("password123").
3. Assert has\_password() returns True.

Expected Result

"The sequence of has\_password() returns must be False, then True."

Actual Result (Pass/Fail/Exception)

Pass: Password status was correctly tracked through the lifecycle.

Comment (including references)

Core functionality test for password status retrieval.

###### b. test\_set\_empty\_password (UT-CAM-PW-STAT-INV)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | has\_password(), set\_password() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that when an attempt to set an invalid password (empty string) is rejected, has\_password() correctly remains False, ensuring the status is not corrupted by failed operations."

Input Specifications

"Initial state: No password set. Action: Call set\_password("""")."

**Detailed Step**

1. Assert initial state is False.
2. Call set\_password("").
3. Assert the return value is False.
4. Assert has\_password() remains False.

Expected Result

"set\_password() must return False, and has\_password() must remain False."

Actual Result (Pass/Fail/Exception)

Pass: Status was correctly preserved after the failed operation.

Comment (including references)

Validation check ensuring failed password setting attempts do not corrupt the has\_password flag.

##### P. save\_info()

###### a. test\_save\_info (UT-CAM-SAVE)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera | save\_info() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that the method is callable and returns True, confirming that the logic for triggering the camera's information persistence (saving its state) was successfully executed."

Input Specifications

None.

**Detailed Step**

1. Call camera.save\_info().
2. Assert the return value is True.

Expected Result

save\_info() must return True.

Actual Result (Pass/Fail/Exception)

Pass: The data persistence interface call succeeded.

Comment (including references)

Verification of the interface method responsible for triggering data persistence (state saving) in the system.

#### 2. CameraController

##### A. add\_camera()

###### a. test\_add\_camera (UT-CC-ADD-SINGLE)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | add\_camera(x, y) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the successful addition of a single camera, ensuring the correct ID (1) is assigned, the total count increments, and all internal metadata (\_cameras, \_camera\_info) is updated with the location and enabled status."

Input Specifications

"Initial state: 0 cameras. Action: Call add\_camera(100, 200)."

**Detailed Step**

1. Call add\_camera(100, 200).
2. Assert the return value is True.
3. Assert total count is 1 and the camera information is correctly registered with ID 1.

Expected Result

"Returns True. \_total\_camera\_number is 1, and \_next\_camera\_id is 2. Camera info (100, 200, True) is stored."

Actual Result (Pass/Fail/Exception)

Pass: Camera was successfully added and tracked.

Comment (including references)

Core function test for registering a new surveillance device.

###### b. test\_add\_multiple\_cameras (UT-CC-ADD-MULTI)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | add\_camera(x, y) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Verifies that multiple sequential calls to add\_camera() correctly handle the ID sequencing and consistently update the total camera count and internal tracking.

Input Specifications

Action: Call add\_camera() three times with varying coordinates.

**Detailed Step**

1. Call add\_camera() three times.
2. Assert that \_total\_camera\_number is 3.
3. Assert that \_next\_camera\_id is 4.
4. Verify coordinates for cameras 1, 2, and 3.

Expected Result

"\_total\_camera\_number must be 3, and the next available ID must be 4. All three cameras should be registered."

Actual Result (Pass/Fail/Exception)

"Pass: Sequential camera additions were successful, and ID management was correct."

Comment (including references)

Verification of internal ID management and state tracking robustness under sequential operation.

##### B. delete\_camera()

###### a. test\_delete\_camera (UT-CC-DEL-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | delete\_camera(id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the successful deletion of an existing camera, ensuring the total count decreases, and the camera is removed from all internal collections (\_cameras, \_camera\_info)."

Input Specifications

"Initial state: Two cameras exist (IDs 1, 2). Action: Call delete\_camera(1)."

**Detailed Step**

1. Add two cameras.
2. Call delete\_camera(1).
3. Assert the return value is True.
4. Assert total count is 1 and ID 1 is no longer in the tracking lists.

Expected Result

"Returns True. \_total\_camera\_number decreases by 1, and the camera ID is removed from all internal dictionaries."

Actual Result (Pass/Fail/Exception)

Pass: Camera was successfully de-registered.

Comment (including references)

Core function test for camera de-registration and state update.

###### b. test\_delete\_nonexistent\_camera (UT-CC-DEL-FAIL)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | delete\_camera(id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling delete\_camera() with a non-existent ID fails, returns False, and ensures the controller's state remains unaltered."

Input Specifications

Initial state: 0 cameras. Action: Call delete\_camera(999).

**Detailed Step**

1. Ensure no cameras exist.
2. Call delete\_camera(999).
3. Assert the return value is False.
4. Assert total count remains 0.

Expected Result

"Returns False. The internal state (e.g., \_total\_camera\_number) must remain unchanged."

Actual Result (Pass/Fail/Exception)

Pass: Deletion of a non-existent camera was correctly rejected.

Comment (including references)

Input validation and state preservation check for deletion logic.

###### c. test\_delete\_camera\_with\_password (UT-CC-DEL-PW)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | delete\_camera(id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies data consistency: when a camera with a stored password is deleted, its entry is removed from the \_camera\_passwords collection alongside the camera object."

Input Specifications

Initial state: Camera ID 1 has a password set. Action: Call delete\_camera(1).

**Detailed Step**

1. Add camera and set password.
2. Call delete\_camera(1).
3. Assert \_camera\_passwords no longer contains ID 1.

Expected Result

Returns True. ID 1 must be removed from the \_camera\_passwords dictionary.

Actual Result (Pass/Fail/Exception)

Pass: Associated password data was correctly cleaned up upon camera deletion.

Comment (including references)

Data consistency and cleanup validation for linked attributes.

###### d. test\_delete\_camera\_with\_device\_camera\_stop (UT-CC-DEL-INT)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | delete\_camera(id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies integration: ensures that when a camera is deleted, the underlying virtual device's stop() method is called once to correctly shut down the device simulator."

Input Specifications

Initial state: Camera 1 exists with a mockable device. Action: Call delete\_camera(1).

**Detailed Step**

1. Add camera. Patch the underlying device's stop method.
2. Call delete\_camera(1).
3. Assert the mock stop method was called exactly once.

Expected Result

The underlying \_device\_camera.stop() method must be called once. Deletion returns True.

Actual Result (Pass/Fail/Exception)

Pass: Device cleanup protocol was correctly initiated.

Comment (including references)

Verification of the internal device cleanup protocol upon de-registration.

###### e. test\_delete\_camera\_handles\_stop\_exception (UT-CC-DEL-ROB)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | delete\_camera(id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies robustness: ensures that the deletion process completes successfully (returns True) and the controller's state is updated, even if the underlying device's stop() method throws an exception."

Input Specifications

Initial state: Camera 1 exists. Action: Mock the device's stop() method to raise an Exception. Call delete\_camera(1).

**Detailed Step**

1. Add camera. Patch device stop() to raise an Exception.
2. Call delete\_camera(1).
3. Assert the return value is True.
4. Assert the camera is actually deleted (count decreased).

Expected Result

Returns True. The deletion must succeed despite the internal device error.

Actual Result (Pass/Fail/Exception)

Pass: The system recovered from the device exception and completed the deletion.

Comment (including references)

Robustness check for external component failure during cleanup.

##### C. enable\_cameras()

###### a. test\_enable\_cameras\_list (UT-CC-EN-LIST-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | enable\_cameras(camera\_id\_list) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Verifies that calling enable\_cameras() with a list of valid camera IDs successfully switches all specified cameras to the enabled state.

Input Specifications

"Initial state: Cameras 1 and 2 are disabled. Camera 3 is enabled. Action: Call enable\_cameras([1, 2])."

**Detailed Step**

1. Add 3 cameras; disable IDs 1 and 2.
2. Call enable\_cameras([1, 2]).
3. Assert the return value is True.
4. Assert cameras 1, 2, and 3 are all enabled.

Expected Result

"Returns True. Cameras 1 and 2 are successfully enabled, and the state of Camera 3 is preserved as enabled."

Actual Result (Pass/Fail/Exception)

Pass: Bulk activation of cameras succeeded.

Comment (including references)

Core function test for bulk activation operation.

###### b. test\_enable\_cameras\_partial\_failure (UT-CC-EN-LIST-ROB)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | enable\_cameras(camera\_id\_list) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the method's robustness by ensuring that it correctly processes and enables valid cameras even if the input list contains non-existent camera IDs, without crashing or failing the entire operation."

Input Specifications

"Initial state: Cameras 1 and 2 exist. Action: Call enable\_cameras([1, 2, 999]) (ID 999 does not exist)."

**Detailed Step**

1. Add 2 cameras.
2. Call enable\_cameras([1, 2, 999]).
3. Assert that cameras 1 and 2 are successfully enabled.

Expected Result

"Cameras 1 and 2 must be successfully enabled, ignoring the non-existent ID 999."

Actual Result (Pass/Fail/Exception)

Pass: Robustness verified; valid operations completed despite invalid input in the list.

Comment (including references)

Robustness and error tolerance check for invalid inputs in bulk operations.

##### D. disable\_cameras()

###### a. test\_disable\_cameras\_list (UT-CC-DIS-LIST-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | disable\_cameras(camera\_id\_list) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Verifies that calling disable\_cameras() with a list of valid camera IDs successfully switches all specified cameras to the disabled state.

Input Specifications

"Initial state: Cameras 1 and 2 are enabled. Action: Call disable\_cameras([1, 2])."

**Detailed Step**

1. Add 2 cameras; ensure both are enabled.
2. Call disable\_cameras([1, 2]).
3. Assert the return value is True.
4. Assert cameras 1 and 2 are disabled.

Expected Result

Returns True. Cameras 1 and 2 are successfully disabled.

Actual Result (Pass/Fail/Exception)

Pass: Bulk deactivation of cameras succeeded.

Comment (including references)

Core function test for bulk deactivation operation.

###### b. test\_disable\_cameras\_partial\_failure (UT-CC-DIS-LIST-ROB)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | disable\_cameras(camera\_id\_list) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the method's robustness by ensuring that it correctly processes and disables valid cameras even if the input list contains non-existent camera IDs, without failing the entire operation."

Input Specifications

"Initial state: Cameras 1 and 2 exist and are enabled. Action: Call disable\_cameras([1, 2, 999]) (ID 999 does not exist)."

**Detailed Step**

1. Add 2 cameras; ensure both are enabled.
2. Call disable\_cameras([1, 2, 999]).
3. Assert that cameras 1 and 2 are successfully disabled.

Expected Result

"Cameras 1 and 2 must be successfully disabled, ignoring the non-existent ID 999."

Actual Result (Pass/Fail/Exception)

Pass: Robustness verified; valid operations completed despite invalid input in the list.

Comment (including references)

Robustness and error tolerance check for invalid inputs in bulk operations.

##### E. enable\_all\_camera()

###### a. test\_enable\_all\_cameras (UT-CC-EN-ALL-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | enable\_all\_camera() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling enable\_all\_camera() successfully switches all registered cameras to the enabled state, regardless of their previous disabled state."

Input Specifications

"Initial state: Two cameras exist, both are disabled. Action: Call enable\_all\_camera()."

**Detailed Step**

1. Add 2 cameras; disable both.
2. Call enable\_all\_camera().
3. Assert the return value is True.
4. Assert both cameras are enabled.

Expected Result

Returns True. All registered cameras must be successfully enabled.

Actual Result (Pass/Fail/Exception)

Pass: Global activation succeeded.

Comment (including references)

Core function test for bulk activation of all devices.

###### b. test\_camera\_enable\_disable\_workflow (UT-CC-EN-ALL-WORK)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | enable\_all\_camera() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the operation of enable\_all\_camera() at the end of a complex state change workflow (including individual and bulk disabling), ensuring it correctly resets all cameras to the enabled state."

Input Specifications

"Initial state: Three cameras exist. Action: Sequentially disable various cameras, followed by a final call to enable\_all\_camera()."

**Detailed Step**

1. Add 3 cameras.
2. Perform mixed disable/enable operations.
3. Call disable\_all\_camera().
4. Call enable\_all\_camera().
5. Assert all 3 cameras are enabled.

Expected Result

All cameras must be confirmed to be in the enabled state (is\_enabled() is True).

Actual Result (Pass/Fail/Exception)

Pass: The final global enable operation correctly restored all camera states.

Comment (including references)

Integration and consistency check across multiple bulk state changes.

##### F. disable\_all\_camera()

###### a. test\_disable\_all\_cameras (UT-CC-DIS-ALL-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | disable\_all\_camera() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling disable\_all\_camera() successfully switches all registered cameras to the disabled state, regardless of their previous enabled state."

Input Specifications

"Initial state: Two cameras exist, both are enabled. Action: Call disable\_all\_camera()."

**Detailed Step**

1. Add 2 cameras; ensure both are enabled.
2. Call disable\_all\_camera().
3. Assert the return value is True.
4. Assert both cameras are disabled.

Expected Result

Returns True. All registered cameras must be successfully disabled.

Actual Result (Pass/Fail/Exception)

Pass: Global deactivation succeeded.

Comment (including references)

Core function test for bulk deactivation of all devices.

###### b. test\_camera\_enable\_disable\_workflow (UT-CC-DIS-ALL-WORK)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | disable\_all\_camera() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the operation of disable\_all\_camera() within a complex state change workflow, ensuring it correctly switches all cameras to the disabled state after individual and bulk enabling/disabling operations."

Input Specifications

"Initial state: Three cameras exist. Action: Sequentially disable/enable cameras, followed by a final call to disable\_all\_camera()."

**Detailed Step**

1. Add 3 cameras.
2. Perform mixed enable/disable operations.
3. Call disable\_all\_camera().
4. Assert all 3 cameras are disabled.

Expected Result

All cameras must be confirmed to be in the disabled state (is\_enabled() is False).

Actual Result (Pass/Fail/Exception)

Pass: The final global disable operation correctly set all camera states to disabled.

Comment (including references)

Integration and consistency check across multiple bulk state changes.

##### G. enable\_camera()

###### a. test\_enable\_camera (UT-CC-EN-SINGLE-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | enable\_camera(id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Verifies that calling enable\_camera() with a valid ID successfully transitions the specified camera from the disabled state to the active state.

Input Specifications

Initial state: Camera ID 1 exists and is disabled. Action: Call enable\_camera(1).

**Detailed Step**

1. Add camera and disable it.
2. Call enable\_camera(1).
3. Assert the return value is True.
4. Assert the camera's enabled state is True.

Expected Result

Returns True. The camera's enabled state must be True.

Actual Result (Pass/Fail/Exception)

Pass: The camera successfully transitioned to the enabled state.

Comment (including references)

Core function test for setting a single camera to an active state.

###### b. test\_enable\_nonexistent\_camera (UT-CC-EN-SINGLE-FAIL)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | enable\_camera(id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that attempting to call enable\_camera() using a non-existent ID fails, returns False, and does not alter the controller's state."

Input Specifications

Initial state: No camera with ID 999 exists. Action: Call enable\_camera(999).

**Detailed Step**

1. Call enable\_camera(999).
2. Assert the return value is False.

Expected Result

Returns False.

Actual Result (Pass/Fail/Exception)

Pass: Request for a nonexistent camera was correctly rejected.

Comment (including references)

Input validation check for non-existent camera IDs.

##### H. disable\_camera()

###### a. test\_disable\_camera (UT-CC-DIS-SINGLE-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | disable\_camera(id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Verifies that calling disable\_camera() with a valid ID successfully transitions the specified camera from the active state to the inactive state.

Input Specifications

Initial state: Camera ID 1 exists and is enabled. Action: Call disable\_camera(1).

**Detailed Step**

1. Add camera and ensure it is enabled.
2. Call disable\_camera(1).
3. Assert the return value is True.
4. Assert the camera's enabled state is False.

Expected Result

Returns True. The camera's enabled state must be False.

Actual Result (Pass/Fail/Exception)

Pass: The camera successfully transitioned to the disabled state.

Comment (including references)

Core function test for setting a single camera to an inactive state.

###### b. test\_disable\_nonexistent\_camera (UT-CC-DIS-SINGLE-FAIL)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | disable\_camera(id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that attempting to call disable\_camera() using a non-existent ID fails, returns False, and does not alter the controller's state."

Input Specifications

Initial state: No camera with ID 999 exists. Action: Call disable\_camera(999).

**Detailed Step**

1. Call disable\_camera(999).
2. Assert the return value is False.

Expected Result

Returns False.

Actual Result (Pass/Fail/Exception)

Pass: Request for a nonexistent camera was correctly rejected.

Comment (including references)

Input validation check for non-existent camera IDs.

##### I. control\_single\_camera()

###### a. test\_control\_single\_camera\_take\_picture (UT-CC-CTRL-SNAP)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | control\_single\_camera(id, control\_id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Verifies that using control ID 0 successfully triggers the camera's snapshot (take picture) functionality.

Input Specifications

"Initial state: Camera 1 is enabled. Action: Call control\_single\_camera(1, 0)."

**Detailed Step**

1. Add camera 1.
2. Call control\_single\_camera(1, 0).
3. Assert the return value is True.

Expected Result

Returns True. The snapshot command is correctly dispatched.

Actual Result (Pass/Fail/Exception)

Pass: Snapshot command execution succeeded.

Comment (including references)

Dispatcher test for core image capture action.

###### b. test\_control\_single\_camera\_start\_recording (UT-CC-CTRL-REC-START)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | control\_single\_camera(id, control\_id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that using control ID 1 successfully triggers the recording start functionality, and the camera's status transitions to ""Recording""."

Input Specifications

"Initial state: Camera 1 is idle. Action: Call control\_single\_camera(1, 1)."

**Detailed Step**

1. Call control\_single\_camera(1, 1).
2. Assert the return value is True.
3. Assert camera status is "Recording".

Expected Result

"Returns True. Camera status is updated to ""Recording""."

Actual Result (Pass/Fail/Exception)

"Pass: Recording start command succeeded, and status updated."

Comment (including references)

Dispatcher test for core recording start action.

###### c. test\_control\_single\_camera\_stop\_recording (UT-CC-CTRL-REC-STOP)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | control\_single\_camera(id, control\_id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that using control ID 2 successfully triggers the recording stop functionality, and the camera's status returns to its default state (""idle"")."

Input Specifications

"Initial state: Camera 1 is recording. Action: Call control\_single\_camera(1, 2)."

**Detailed Step**

1. Start recording (ID 1).
2. Call control\_single\_camera(1, 2).
3. Assert the return value is True.
4. Assert camera status is "idle".

Expected Result

"Returns True. Camera status transitions away from ""Recording""."

Actual Result (Pass/Fail/Exception)

Pass: Recording stop command succeeded.

Comment (including references)

Dispatcher test for core recording stop action.

###### d. test\_control\_single\_camera\_pan\_left (UT-CC-CTRL-PAN)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | control\_single\_camera(id, control\_id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the dispatcher correctly executes a movement command (e.g., pan left, using ID 0 in the coverage test's dispatcher context) by checking the return value."

Input Specifications

"Initial state: Camera 1 enabled. Action: Call control\_single\_camera(1, 0)."

**Detailed Step**

1. Add camera 1.
2. Call control\_single\_camera(1, 0) (Pan Left).
3. Assert the return value is True.

Expected Result

Returns True. The Pan Left command is successfully dispatched.

Actual Result (Pass/Fail/Exception)

Pass: Pan left command successfully dispatched.

Comment (including references)

Dispatcher test for movement control functionality.

###### e. test\_control\_disabled\_camera (UT-CC-CTRL-FAIL-DIS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | control\_single\_camera(id, control\_id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that attempting to control a disabled camera (e.g., snapshot, ID 0) is rejected, returns False, and prevents the execution of the command."

Input Specifications

"Initial state: Camera 1 is disabled. Action: Call control\_single\_camera(1, 0)."

**Detailed Step**

1. Disable camera 1.
2. Call control\_single\_camera(1, 0).
3. Assert the return value is False.

Expected Result

Returns False.

Actual Result (Pass/Fail/Exception)

Pass: Control command was correctly rejected in the disabled state.

Comment (including references)

Security validation check for controlling inactive devices.

###### f. test\_control\_nonexistent\_camera (UT-CC-CTRL-FAIL-NON)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | control\_single\_camera(id, control\_id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that attempting to control a non-existent camera ID is rejected, returns False, and prevents the dispatcher from executing."

Input Specifications

"Action: Call control\_single\_camera(999, 0) (ID 999 does not exist)."

**Detailed Step**

1. Call control\_single\_camera(999, 0).
2. Assert the return value is False.

Expected Result

Returns False.

Actual Result (Pass/Fail/Exception)

Pass: Command was correctly rejected for a non-existent ID.

Comment (including references)

Input validation check for camera existence.

###### g. test\_control\_invalid\_control\_id (UT-CC-CTRL-FAIL-INV)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | control\_single\_camera(id, control\_id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that providing an undefined or invalid control\_id (e.g., 99) is rejected by the dispatcher, returning False."

Input Specifications

"Initial state: Camera 1 exists. Action: Call control\_single\_camera(1, 99)."

**Detailed Step**

1. Add camera 1.
2. Call control\_single\_camera(1, 99).
3. Assert the return value is False.

Expected Result

Returns False.

Actual Result (Pass/Fail/Exception)

Pass: Command was rejected due to an invalid control ID.

Comment (including references)

Dispatcher validation check for command ID range and definition.

##### J. display\_thumbnail\_view()

###### a. test\_display\_thumbnail\_view\_no\_cameras (UT-CC-THUMB-EMPTY)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | display\_thumbnail\_view() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Verifies that calling display\_thumbnail\_view() when no cameras are registered gracefully handles the empty list edge case by returning None.

Input Specifications

Initial state: 0 cameras registered. Action: Call display\_thumbnail\_view().

**Detailed Step**

1. Ensure the controller is empty.
2. Call display\_thumbnail\_view().
3. Assert the return value is None.

Expected Result

Returns None.

Actual Result (Pass/Fail/Exception)

Pass: Empty state was handled correctly.

Comment (including references)

Input validation check for the empty list edge case.

###### b. test\_display\_thumbnail\_view\_with\_multiple\_cameras (UT-CC-THUMB-MULTI)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | display\_thumbnail\_view() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling display\_thumbnail\_view() when multiple cameras exist successfully initiates the composite view generation, returning a valid image object or None if image files are unavailable."

Input Specifications

Initial state: 3 enabled cameras exist. Action: Call display\_thumbnail\_view().

**Detailed Step**

1. Add 3 cameras.
2. Call display\_thumbnail\_view().
3. Assert the result is None or an instance of PIL.Image.Image.

Expected Result

"Returns a non-null object or None, confirming the operation succeeds without crashing."

Actual Result (Pass/Fail/Exception)

Pass: Composite view generation succeeded.

Comment (including references)

Core functionality test for generating the composite thumbnail view.

###### c. test\_display\_thumbnail\_view\_with\_mixed\_cameras (UT-CC-THUMB-MIXED)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | display\_thumbnail\_view(), disable\_camera() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that when a mix of enabled and disabled cameras exists, the thumbnail view correctly processes only the active cameras for composition."

Input Specifications

Initial state: 3 cameras exist; Camera 1 is disabled. Action: Call display\_thumbnail\_view().

**Detailed Step**

1. Add 3 cameras.
2. Disable Camera 1.
3. Call display\_thumbnail\_view().
4. Assert the operation succeeds (returns image or None), implying only enabled cameras were used.

Expected Result

"Returns a non-null object or None, confirming the operation succeeds and correctly handles disabled camera filtering."

Actual Result (Pass/Fail/Exception)

Pass: State awareness verified; view composition succeeded with mixed camera states.

Comment (including references)

State awareness check for view composition (only enabled cameras should contribute).

###### d. test\_display\_thumbnail\_view\_grid\_calculation (UT-CC-THUMB-GRID)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | display\_thumbnail\_view() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that the internal logic correctly calculates the grid dimensions and processes the view generation for varying numbers of cameras (e.g., 1, 3, 4), ensuring the layout is handled correctly."

Input Specifications

"Sequential actions adding 1, 3, and 4 cameras and calling display\_thumbnail\_view() after each addition."

**Detailed Step**

1. Add 1 camera, call view.
2. Add 2 more (total 3), call view.
3. Add 1 more (total 4), call view.
4. Assert all calls succeed.

Expected Result

"The function must execute successfully for varying camera counts without error, validating internal grid dimension calculations."

Actual Result (Pass/Fail/Exception)

Pass: Grid calculation logic passed for varying camera counts.

Comment (including references)

Coverage test for internal thumbnail layout/grid dimension calculations.

##### K. display\_single\_view()

###### a. test\_display\_single\_view (UT-CC-VIEW-SINGLE-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | display\_single\_view(id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling display\_single\_view() with a valid ID successfully retrieves the view from the specified, enabled camera."

Input Specifications

Initial state: Camera 1 exists and is enabled. Action: Call display\_single\_view(1).

**Detailed Step**

1. Add camera 1.
2. Call display\_single\_view(1).
3. Assert the result is a non-null image object or None if image files are unavailable.

Expected Result

"Returns a valid PIL.Image.Image object or None, confirming the operation succeeds."

Actual Result (Pass/Fail/Exception)

Pass: Single view retrieval succeeded.

Comment (including references)

Core functionality test for retrieving a single camera feed.

###### b. test\_display\_single\_view\_disabled\_camera (UT-CC-VIEW-SINGLE-FAIL-DIS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | display\_single\_view(id), disable\_camera() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that attempting to display the view of a disabled camera is rejected, returning None, thus enforcing the security policy."

Input Specifications

Initial state: Camera 1 is disabled. Action: Call display\_single\_view(1).

**Detailed Step**

1. Disable camera 1.
2. Call display\_single\_view(1).
3. Assert the result is None.

Expected Result

Returns None.

Actual Result (Pass/Fail/Exception)

Pass: View access was correctly denied due to the disabled state.

Comment (including references)

Security validation check for view access control in a disabled state.

###### c. test\_display\_single\_view\_nonexistent\_camera (UT-CC-VIEW-SINGLE-FAIL-NON)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | display\_single\_view(id) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that attempting to display the view of a non-existent camera ID is correctly rejected, returning None."

Input Specifications

Action: Call display\_single\_view(999) (ID 999 does not exist).

**Detailed Step**

1. Ensure no camera with ID 999 exists.
2. Call display\_single\_view(999).
3. Assert the result is None.

Expected Result

Returns None.

Actual Result (Pass/Fail/Exception)

Pass: Request for a nonexistent camera was correctly rejected.

Comment (including references)

Input validation check for non-existent camera IDs.

##### L. get\_all\_camera\_info()

###### a. test\_get\_all\_camera\_info\_empty (UT-CC-INFO-EMPTY)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | get\_all\_camera\_info() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that calling get\_all\_camera\_info() when no cameras are registered gracefully handles the empty controller edge case by returning an empty list ([])."

Input Specifications

Initial state: 0 cameras registered. Action: Call get\_all\_camera\_info().

**Detailed Step**

1. Ensure the controller is empty.
2. Call get\_all\_camera\_info().
3. Assert the returned value is [].

Expected Result

Returns [].

Actual Result (Pass/Fail/Exception)

Pass: Empty state was handled correctly.

Comment (including references)

Input validation check for the empty controller edge case.

###### b. test\_get\_all\_camera\_info (UT-CC-INFO-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | get\_all\_camera\_info() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that the method correctly retrieves the info list for all registered cameras, including their ID, location, and enabled status."

Input Specifications

"Initial state: 2 cameras exist (IDs 1, 2). Action: Call get\_all\_camera\_info()."

**Detailed Step**

1. Add 2 cameras.
2. Call get\_all\_camera\_info().
3. Assert the list length is 2 and check the coordinates and enabled status (1) of the returned info.

Expected Result

"Returns a list of length 2, where each element contains the correct [ID, X, Y, Enabled Status]."

Actual Result (Pass/Fail/Exception)

Pass: Global status information was correctly retrieved.

Comment (including references)

Core functionality test for retrieving global camera status.

###### c. test\_get\_all\_camera\_info\_sorted\_order (UT-CC-INFO-ORDER)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | get\_all\_camera\_info() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that the camera information list is returned in a predictable, sorted order based on the Camera ID, ensuring reliable data display sequence."

Input Specifications

"Initial state: 3 cameras added, possibly in non-sequential order if IDs were deleted/reused (though IDs are sequential here). Action: Call get\_all\_camera\_info()."

**Detailed Step**

1. Add 3 cameras.
2. Call get\_all\_camera\_info().
3. Extract the IDs from the returned list and assert they are in numerical ascending order.

Expected Result

"The list of camera IDs in the output must be sorted numerically (e.g., [1, 2, 3])."

Actual Result (Pass/Fail/Exception)

Pass: Data ordering was confirmed to be reliable.

Comment (including references)

Integrity check for reliable data display ordering (sorting by ID).

##### M. set\_camera\_password()

###### a. test\_set\_camera\_password (UT-CC-PW-SET-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | set\_camera\_password(id, input\_password) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that a valid password is successfully set for an existing camera, correctly updating the internal \_camera\_passwords collection."

Input Specifications

"Initial state: Camera 1 exists. Action: Call set\_camera\_password(1, ""password123"")."

**Detailed Step**

1. Add camera 1.
2. Call set\_camera\_password(1, "password123").
3. Assert that ID 1 is present in \_camera\_passwords with the correct value.

Expected Result

"ID 1 must be present in \_camera\_passwords with the value ""password123""."

Actual Result (Pass/Fail/Exception)

Pass: Password was successfully assigned to the camera.

Comment (including references)

Core function test for password assignment and internal data consistency.

###### b. test\_set\_password\_nonexistent\_camera (UT-CC-PW-SET-FAIL-NON)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | set\_camera\_password(id, input\_password) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that attempting to set a password for a non-existent camera ID is rejected, ensuring no unintended entries are created in the internal collection."

Input Specifications

"Initial state: 0 cameras exist. Action: Call set\_camera\_password(999, ""password123"")."

**Detailed Step**

1. Call set\_camera\_password(999, "password123").
2. Assert that ID 999 is not present in \_camera\_passwords.

Expected Result

ID 999 must not be found in the \_camera\_passwords dictionary.

Actual Result (Pass/Fail/Exception)

Pass: Password setting for a non-existent camera was correctly ignored.

Comment (including references)

Input validation check for camera existence prior to password assignment.

###### c. test\_set\_camera\_password\_nonexistent\_camera\_handling (UT-CC-PW-SET-ROB)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | set\_camera\_password(id, input\_password) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the system's robustness by ensuring that attempting to set a password for a non-existent camera ID does not raise an unhandled exception, confirming graceful failure."

Input Specifications

"Action: Call set\_camera\_password(999, ""password123"") and monitor for exceptions."

**Detailed Step**

1. Call set\_camera\_password(999, "password123") within a try/except block (or implicitly check for raised exceptions).
2. Assert that the process completes without raising an Exception.

Expected Result

The method must complete successfully (without crashing).

Actual Result (Pass/Fail/Exception)

Pass: Robustness verified; the process handled the invalid ID gracefully.

Comment (including references)

Coverage check for robust handling of invalid IDs during password assignment.

##### N. validate\_camera\_password()

###### a. test\_validate\_camera\_password\_success (UT-CC-PW-VAL-PASS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | validate\_camera\_password(id, input\_password) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that when the correct password is provided for a set camera, the function returns the success code (0)."

Input Specifications

"Initial state: Camera 1 password set to ""password123"". Action: Call validate\_camera\_password(1, ""password123"")."

**Detailed Step**

1. Set password for camera 1.
2. Call validate\_camera\_password(1, "password123").
3. Assert the returned code is 0.

Expected Result

Returns 0 (Success).

Actual Result (Pass/Fail/Exception)

Pass: Correct password validation succeeded.

Comment (including references)

Core function test for password validation success path.

###### b. test\_validate\_camera\_password\_failure (UT-CC-PW-VAL-FAIL)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | validate\_camera\_password(id, input\_password) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that when an incorrect password is provided for a set camera, the function returns the password mismatch code (1)."

Input Specifications

"Initial state: Camera 1 password set to ""password123"". Action: Call validate\_camera\_password(1, ""wrongpassword"")."

**Detailed Step**

1. Set password for camera 1.
2. Call validate\_camera\_password(1, "wrongpassword").
3. Assert the returned code is 1.

Expected Result

Returns 1 (Mismatch/Failure).

Actual Result (Pass/Fail/Exception)

Pass: Incorrect password validation failed as expected.

Comment (including references)

Validation test for the password mismatch path.

###### c. test\_validate\_camera\_password\_nonexistent\_camera (UT-CC-PW-VAL-NON)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | validate\_camera\_password(id, input\_password) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Verifies that attempting to validate a password for a non-existent camera ID returns the camera not found code (-1).

Input Specifications

"Initial state: Camera 999 does not exist. Action: Call validate\_camera\_password(999, ""any"")."

**Detailed Step**

1. Call validate\_camera\_password(999, "password123").
2. Assert the returned code is -1.

Expected Result

Returns -1 (Camera Not Found).

Actual Result (Pass/Fail/Exception)

Pass: Correct error code was returned for the nonexistent ID.

Comment (including references)

Security check for handling validation attempts on nonexistent resources.

###### d. test\_validate\_camera\_password\_not\_set (UT-CC-PW-VAL-UNSET)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | validate\_camera\_password(id, input\_password) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Verifies that validating a camera that exists but has no password set returns the password not set code (-2).

Input Specifications

"Initial state: Camera 1 exists but has no password set. Action: Call validate\_camera\_password(1, ""any"")."

**Detailed Step**

1. Add camera 1.
2. Call validate\_camera\_password(1, "password123").
3. Assert the returned code is -2.

Expected Result

Returns -2 (Password Not Set).

Actual Result (Pass/Fail/Exception)

Pass: Correct error code was returned for the unset password state.

Comment (including references)

Validation check for the initial state of the password attribute.

###### e. test\_validate\_camera\_password\_edge\_cases (UT-CC-PW-VAL-EDGE)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | validate\_camera\_password(id, input\_password) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the security logic when an empty string ("""") is provided as the input password for a camera that has a password set (should fail with mismatch code 1)."

Input Specifications

"Initial state: Camera 1 password set to ""test"". Action: Call validate\_camera\_password(1, """")."

**Detailed Step**

1. Set password for camera 1 to "test".
2. Call validate\_camera\_password(1, "").
3. Assert the returned code is 1.

Expected Result

Returns 1 (Mismatch/Failure).

Actual Result (Pass/Fail/Exception)

Pass: Empty input was correctly handled as a failed validation attempt.

Comment (including references)

Coverage test for security logic on empty string input against a set password.

##### O. delete\_camera\_password()

###### a. test\_delete\_camera\_with\_password (UT-CC-PW-DEL-IMPL)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | delete\_camera\_password() (Implicitly verified by delete\_camera()) | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies data consistency: when a camera with a stored password is deleted, the underlying password removal logic is executed, ensuring the entry is removed from the \_camera\_passwords collection."

Input Specifications

Initial state: Camera ID 1 has a password set. Action: Call delete\_camera(1).

**Detailed Step**

1. Add camera and set password.
2. Call delete\_camera(1).
3. Assert the password entry for ID 1 is no longer present in \_camera\_passwords.

Expected Result

"Camera deletion must return True, and the associated password data must be successfully removed."

Actual Result (Pass/Fail/Exception)

Pass: Associated password data was correctly cleaned up upon camera deletion.

Comment (including references)

"Verification of internal data cleanup and consistency, confirming the password removal logic is executed as part of the de-registration process."

## **3. Security**

### A. Alarm Core (SecuritySystem)

| ID | File:Line | Test | Scenario & Inputs | Expected Result | Traceability |
| --- | --- | --- | --- | --- | --- |
| UT-SEC-001 | security/test\_security\_system.py:70 | test\_arm\_disarm\_basic | Validates ‘Arm disarm basic’ behavior | Pass: observed state/response matches SDS/SRS design. | SRS UC2 / SDS VI.2d |
| UT-SEC-002 | security/test\_security\_system.py:83 | test\_entry\_delay\_disarm\_prevents\_alarm | Validates ‘Entry delay disarm prevents alarm’ behavior | Pass: observed state/response matches SDS/SRS design. | SRS UC2 / SDS VI.2d |
| UT-SEC-003 | security/test\_security\_system.py:107 | test\_entry\_delay\_expires\_triggers\_alarm | Validates ‘Entry delay expires triggers alarm’ behavior | Pass: observed state/response matches SDS/SRS design. | SRS UC2 / SDS VI.2d |
| UT-SEC-004 | security/test\_security\_system.py:127 | test\_home\_mode\_door\_triggers\_delay\_but\_motion\_ignored | Validates ‘Home mode door triggers delay but motion ignored’ behavior | Pass: observed state/response matches SDS/SRS design. | SRS UC2 / SDS VI.2d |
| UT-SEC-005 | security/test\_security\_system.py:159 | test\_home\_mode\_motion\_ignored | Validates ‘Home mode motion ignored’ behavior | Pass: observed state/response matches SDS/SRS design. | SRS UC2 / SDS VI.2d |
| UT-SEC-006 | security/test\_security\_system.py:179 | test\_away\_mode\_motion\_triggers\_entry\_delay\_and\_alarm | Validates ‘Away mode motion triggers entry delay and alarm’ behavior | Pass: observed state/response matches SDS/SRS design. | SRS UC2 / SDS VI.2d |
| UT-SEC-007 | security/test\_security\_system.py:202 | test\_zone\_specific\_arming\_blocks\_other\_zones | Validates ‘Zone specific arming blocks other zones’ behavior | Pass: observed state/response matches SDS/SRS design. | SRS UC2 / SDS VI.2d |
| UT-SEC-008 | security/test\_security\_system.py:236 | test\_readiness\_check\_blocks\_arming\_when\_door\_open | Validates ‘Readiness check blocks arming when door open’ behavior | Pass: observed state/response matches SDS/SRS design. | SRS UC2 / SDS VI.2d |
| UT-SEC-009 | security/test\_security\_system.py:254 | test\_trigger\_panic\_from\_disarmed | Validates ‘Trigger panic from disarmed’ behavior | Pass: observed state/response matches SDS/SRS design. | SRS UC2 / SDS VI.2d |

For every alarm branch in security/security\_system.py, there is a corresponding test with explicit inputs: security/test\_security\_system.py:107 opens a DOOR sensor and advances the fake clock past the entry delay to prove SecuritySystem.tick raises the siren. security/test\_security\_system.py:159 switches to STAY mode, emits a motion event, and asserts the motion-only sub-test leaves the system idle. security/test\_security\_system.py:202 introduces two zones to show that arming a subset keeps the inactive zone silent. Finally, the readiness guard at security/test\_security\_system.py:236 prevents arming when a door reports OPEN, matching SDS readiness diagrams.

**Per-test detail (UT-SEC)**

#### 1. test\_arm\_disarm\_basic

| Field | Value |
| --- | --- |
| Class | SecuritySystem |
| Method | arm / disarm |
| Author | Jonghwa An |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**  
Validates the happy path of arming then disarming with no sensor activity.

**Input Specifications**  
- Call arm(SecurityMode.AWAY) followed by disarm().

**Expected Result**  
- Mode transitions DISARMED ??AWAY ??DISARMED.  
- Siren remains off.

**Actual Result (Pass/Fail/Exception)**  
- Pass: FSM transitions recorded; no alarms fired.

**Comment (including references)**  
SRS UC2a; SDS VI.2d base flow.

#### 2. test\_entry\_delay\_disarm\_prevents\_alarm

| Field | Value |
| --- | --- |
| Class | SecuritySystem |
| Method | handle\_sensor\_event / disarm |
| Author | Jonghwa An |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**  
Ensures disarming during ENTRY\_DELAY suppresses alarm activation.

**Input Specifications**  
- Arm AWAY, trigger door sensor, advance clock but call disarm() before delay expires.

**Expected Result**  
- Countdown cancelled, state returns to DISARMED, no monitoring call.

**Actual Result (Pass/Fail/Exception)**  
- Pass: timer cleared and alarm\_state never reaches ALARM\_ACTIVE.

**Comment (including references)**  
SRS UC2c alternate; SDS VI.2d entry delay.

#### 3. test\_entry\_delay\_expires\_triggers\_alarm

| Field | Value |
| --- | --- |
| Class | SecuritySystem |
| Method | tick |
| Author | Jonghwa An |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**  
Confirms that failing to disarm within the delay causes an alarm.

**Input Specifications**  
- Arm AWAY, trigger sensor, advance fake clock beyond configured delay.

**Expected Result**  
- alarm\_state becomes ALARM\_ACTIVE, siren + monitoring triggered.

**Actual Result (Pass/Fail/Exception)**  
- Pass: siren mock activated; monitoring stub records INTRUSION.

**Comment (including references)**  
SRS UC2c main flow; SDS VI.2d timing.

#### 4. test\_home\_mode\_door\_triggers\_delay\_but\_motion\_ignored

| Field | Value |
| --- | --- |
| Class | SecuritySystem |
| Method | handle\_sensor\_event |
| Author | Jonghwa An |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**  
Home/STAY should watch perimeter sensors but ignore interior motion.

**Input Specifications**  
- Arm STAY, trigger door sensor then motion sensor.

**Expected Result**  
- Door event enters ENTRY\_DELAY; motion event ignored.

**Actual Result (Pass/Fail/Exception)**  
- Pass: logs show door countdown, motion produces no change.

**Comment (including references)**  
SRS UC2b selective arming; SDS VI.2f zone handling.

#### 5. test\_home\_mode\_motion\_ignored

| Field | Value |
| --- | --- |
| Class | SecuritySystem |
| Method | handle\_sensor\_event |
| Author | Jonghwa An |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**  
Reinforces that interior-only events never change state under STAY.

**Input Specifications**  
- With STAY armed, emit motion event only.

**Expected Result**  
- State stays STAY; no timers started.

**Actual Result (Pass/Fail/Exception)**  
- Pass: \_pending\_entry\_delay remains None.

**Comment (including references)**  
SRS UC2b alt; SDS VI.2f.

#### 6. test\_away\_mode\_motion\_triggers\_entry\_delay\_and\_alarm

| Field | Value |
| --- | --- |
| Class | SecuritySystem |
| Method | handle\_sensor\_event / tick |
| Author | Jonghwa An |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**  
Ensures motion sensors participate when the system is AWAY.

**Input Specifications**  
- Arm AWAY, emit motion event, advance time beyond delay.

**Expected Result**  
- Entry delay engaged then escalates to ALARM\_ACTIVE.

**Actual Result (Pass/Fail/Exception)**  
- Pass: siren triggered and monitoring notified.

**Comment (including references)**  
SRS UC2d; SDS VI.2d AWAY branch.

#### 7. test\_zone\_specific\_arming\_blocks\_other\_zones

| Field | Value |
| --- | --- |
| Class | SecuritySystem |
| Method | arm\_selected\_zones |
| Author | Jonghwa An |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**  
Verifies that when arming subset of zones, other zones remain idle.

**Input Specifications**  
- Arm ["Front Door"], emit event from unarmed zone.

**Expected Result**  
- Unarmed zone events ignored; armed zone event triggers delay.

**Actual Result (Pass/Fail/Exception)**  
- Pass: \_armed\_zones filter enforces behavior.

**Comment (including references)**  
SRS UC2b; SDS VI.2f zone assignments.

#### 8. test\_readiness\_check\_blocks\_arming\_when\_door\_open

| Field | Value |
| --- | --- |
| Class | SecuritySystem |
| Method | arm |
| Author | Jonghwa An |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**  
Readiness logic should refuse to arm when sensors report open/active.

**Input Specifications**  
- Mark door sensor OPEN, attempt arm(SecurityMode.AWAY).

**Expected Result**  
- Returns False, readiness warning issued, state remains DISARMED.

**Actual Result (Pass/Fail/Exception)**  
- Pass: arm rejects request and records readiness issue.

**Comment (including references)**  
SRS UC2f readiness; SDS VI.2d guard.

#### 9. test\_trigger\_panic\_from\_disarmed

| Field | Value |
| --- | --- |
| Class | SecuritySystem |
| Method | trigger\_panic |
| Author | Jonghwa An |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**  
Confirms panic operates even when system is DISARMED.

**Input Specifications**  
- With DISARMED state, call trigger\_panic().

**Expected Result**  
- Immediate ALARM\_ACTIVE, monitoring reason PANIC, siren activated.

**Actual Result (Pass/Fail/Exception)**  
- Pass: logs show panic escalation without entry delay.

**Comment (including references)**  
SRS UC2g; SDS VI.2h panic wiring.

### B. Lifecycle

#### 1. System

##### A. turn\_on()

###### a. test\_turn\_on\_from\_off\_state (LT-001)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_on() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Verifies that calling turn\_on() when system is OFF results in successful initialization and system state becomes READY.

**Input Specifications**

* System is in OFF state (initial state after constructor)

**Detailed Step**

1. Instantiate System (initially OFF).
2. Call turn\_on().
3. Verify return value and system state.

**Expected Result**

* turn\_on() returns True
* system\_state becomes SystemState.READY
* All core components are initialized (not None)

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn on from off state' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset); Sequence Diagram on Turn the System On, SDS

###### b. test\_turn\_on\_when\_already\_running (LT-002)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_on() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Verifies that calling turn\_on() when system is already running returns False and does not reinitialize components.

**Input Specifications**

* System is already in READY state (turn\_on() called previously)

**Detailed Step**

1. Ensure system is READY.
2. Call turn\_on() again.

**Expected Result**

* turn\_on() returns False
* system\_state remains READY (or current state)
* No exception is raised

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn on when already running' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset); Error handling in Turn the System On flow

###### c. test\_turn\_on\_initializes\_storage\_manager (LT-003)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_on() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Verifies that StorageManager is properly initialized and connected to the database during system turn\_on.

**Input Specifications**

* System is in OFF state

**Detailed Step**

1. Call turn\_on().
2. Inspect system.storage\_manager.

**Expected Result**

* StorageManager is created
* Database connection is established

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn on initializes storage manager' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset); Phase 5 (StorageManager) in Turn the System On sequence

###### d. test\_turn\_on\_initializes\_configuration\_manager (LT-004)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_on() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Verifies that ConfigurationManager is properly initialized with SafeHomeModes, SafetyZones, and SystemSettings.

**Input Specifications**

* System is in OFF state

**Detailed Step**

1. Call turn\_on().
2. Inspect system.configuration\_manager.

**Expected Result**

* ConfigurationManager is created
* SystemSettings are accessible

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn on initializes configuration manager' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset); Phase 4-7 in Turn the System On sequence

###### e. test\_turn\_on\_initializes\_login\_manager (LT-005)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_on() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Verifies that LoginManager is properly initialized during turn\_on.

**Input Specifications**

* System is in OFF state

**Detailed Step**

1. Call turn\_on().
2. Inspect system.login\_manager and default accounts.

**Expected Result**

* LoginManager is created
* Default admin account exists

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn on initializes login manager' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset); Phase 9 (LoginManager) in Turn the System On sequence

###### f. test\_turn\_on\_initializes\_log\_manager (LT-006)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_on() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Verifies that LogManager is properly initialized and logs the SYSTEM\_START event.

**Input Specifications**

* System is in OFF state

**Detailed Step**

1. Call turn\_on().
2. Check logs for SYSTEM\_START event.

**Expected Result**

* LogManager is created
* SYSTEM\_START event is logged

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn on initializes log manager' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset); Phase 8 (LogManager) in Turn the System On sequence

###### g. test\_turn\_on\_initializes\_security\_system (LT-007)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_on() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Verifies that SecuritySystem is properly initialized with alarm and siren components.

**Input Specifications**

* System is in OFF state

**Detailed Step**

1. Call turn\_on().
2. Inspect system.security\_system and its siren.

**Expected Result**

* SecuritySystem is created
* Siren component exists

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn on initializes security system' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset); Phase 3 (Alarm) in Turn the System On sequence

###### h. test\_turn\_on\_initializes\_camera\_controller (LT-008)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_on() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Verifies that CameraController is properly initialized.

**Input Specifications**

* System is in OFF state

**Detailed Step**

1. Call turn\_on().
2. Inspect system.camera\_controller.

**Expected Result**

* CameraController is created

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn on initializes camera controller' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset); Phase 2 (CameraController) in Turn the System On sequence

###### i. test\_turn\_on\_state\_transition (LT-009)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_on() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-ON-009: State transitions during turn\_on

**Input Specifications**

* System is in OFF state

**Detailed Step**

1. Assert state is OFF.
2. Call turn\_on().
3. Assert state is READY.

**Expected Result**

* Initial state is OFF
* Final state is READY after successful turn\_on

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn on state transition' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset); System lifecycle states in SDS

###### j. test\_turn\_on\_callback\_execution (LT-010)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_on() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Verifies that the on\_turn\_on\_complete callback is called after successful initialization.

**Input Specifications**

* System has on\_turn\_on\_complete callback set

**Detailed Step**

1. Register callback.
2. Call turn\_on().
3. Verify callback execution.

**Expected Result**

* Callback is executed after turn\_on completes

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn on callback execution' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset); Device initialization after turn\_on in main.py

###### k. test\_turn\_on\_with\_existing\_database (LT-015)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_on() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Validate that system correctly loads existing data when turning on with a pre-existing database.

**Input Specifications**

* Turn on an initial System instance, modify persisted settings, and shut it down.
* Instantiate a second System pointing at the same storage and call turn\_on().

**Detailed Step**

1. Setup DB with specific data.
2. Start new System instance.
3. Verify loaded configuration matches DB.

**Expected Result**

* Second turn\_on reads the stored configuration and keeps existing credentials intact.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn on with existing database' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset); Database persistence in system lifecycle

###### l. test\_turn\_on\_turn\_off\_cycle (LT-016)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_on() / turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Validate that system can be turned on and off multiple times without issues.

**Input Specifications**

* Reuse one System instance and repeatedly invoke turn\_on()/turn\_off() across multiple cycles.

**Detailed Step**

1. Call turn\_on(), then turn\_off().
2. Repeat 3 times.
3. Verify state after each transition.

**Expected Result**

* Every iteration succeeds without leaking resources, leaving state READY after turn\_on and OFF after turn\_off.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn on turn off cycle' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset); System lifecycle management

##### B. turn\_off()

###### a. test\_turn\_off\_from\_ready\_state (LT-017)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-OFF-001: Shuts the system down from READY.

**Input Specifications**

* Keep the fixture-prepared System in READY and call turn\_off() once.

**Detailed Step**

1. Ensure state is READY.
2. Call turn\_off().

**Expected Result**

* turn\_off() returns True.
* System transitions to OFF with no errors.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn off from ready state' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### b. test\_turn\_off\_when\_already\_off (LT-018)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-OFF-002: Guards double shutdown attempts.

**Input Specifications**

* Call turn\_off() to reach OFF, then immediately call turn\_off() again.

**Detailed Step**

1. Call turn\_off() (Returns True).
2. Call turn\_off() again.

**Expected Result**

* The second invocation returns False and leaves SystemState at OFF.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn off when already off' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### c. test\_turn\_off\_saves\_configuration (LT-019)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-OFF-003: Verifies that configuration is saved before shutdown.

**Input Specifications**

* Modify configuration values via ConfigurationManager before invoking turn\_off().

**Detailed Step**

1. Modify a setting.
2. Call turn\_off().
3. Verify storage persistence.

**Expected Result**

* Shutdown persists settings to storage and the system enters OFF cleanly.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn off saves configuration' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### d. test\_turn\_off\_deactivates\_sensors (LT-020)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-OFF-004: Verifies that all sensors are deactivated during shutdown.

**Input Specifications**

* Attach several MockSensor instances to system.sensors and call turn\_off().

**Detailed Step**

1. Add mock sensors.
2. Call turn\_off().
3. Verify deactivate() was called on sensors.

**Expected Result**

* Each sensor receives deactivate() and monitoring stops.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn off deactivates sensors' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### e. test\_turn\_off\_disables\_cameras (LT-021)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-OFF-005: Verifies that all cameras are disabled during shutdown.

**Input Specifications**

* Register cameras through CameraController and power down the system.

**Detailed Step**

1. Add cameras.
2. Call turn\_off().
3. Verify cameras are disabled.

**Expected Result**

* Camera controller disables every camera feed while transitioning to OFF.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn off disables cameras' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### f. test\_turn\_off\_deactivates\_siren (LT-022)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-OFF-006: Verifies that siren is deactivated during shutdown.

**Input Specifications**

* Force the siren into an active state prior to calling turn\_off().

**Detailed Step**

1. Activate siren.
2. Call turn\_off().
3. Verify siren is inactive.

**Expected Result**

* Siren deactivates so no alarm remains latched after shutdown.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn off deactivates siren' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### g. test\_turn\_off\_logs\_out\_user (LT-023)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-OFF-007: Verifies that logged-in user is logged out during shutdown.

**Input Specifications**

* Authenticate the default admin user through login\_manager and then call turn\_off().

**Detailed Step**

1. Log in as admin.
2. Call turn\_off().
3. Verify current user is None.

**Expected Result**

* Shutdown logs the user out and moves SystemState to OFF.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn off logs out user' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### h. test\_turn\_off\_logs\_shutdown\_event (LT-024)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-OFF-008: Verifies that SYSTEM\_SHUTDOWN event is logged.

**Input Specifications**

* Ensure LogManager is bound and call turn\_off().

**Detailed Step**

1. Call turn\_off().
2. Check log for SYSTEM\_SHUTDOWN.

**Expected Result**

* A SYSTEM\_SHUTDOWN event is appended to the audit log.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn off logs shutdown event' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### i. test\_turn\_off\_disconnects\_database (LT-025)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-OFF-009: Verifies that database connection is closed during shutdown.

**Input Specifications**

* Keep StorageManager connected to the DB before invoking turn\_off().

**Detailed Step**

1. Ensure DB connected.
2. Call turn\_off().
3. Verify DB connection is closed.

**Expected Result**

* StorageManager.disconnect() executes so no connections remain open.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn off disconnects database' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### j. test\_turn\_off\_state\_transition (LT-026)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-OFF-010: Verifies the state transitions: READY -> SHUTDOWN -> OFF

**Input Specifications**

* Operate from READY state and call turn\_off().

**Detailed Step**

1. Monitor state changes during turn\_off() execution.

**Expected Result**

* Lifecycle transitions progress from READY to SHUTDOWN to OFF without stalls.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn off state transition' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### k. test\_turn\_off\_executes\_shutdown\_sequence (LT-047)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Validates 'Turn off executes shutdown sequence' behavior

**Input Specifications**

* Build a System wired with stubbed managers, sensors, siren, and cameras, then call turn\_off().

**Detailed Step**

1. Configure stubs with hooks.
2. Call turn\_off().
3. Verify all hooks called.

**Expected Result**

* Each stub records its shutdown hook (config saved, sensors/cameras/siren deactivated, auth logged out, logs written, storage disconnected).

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn off executes shutdown sequence' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### l. test\_turn\_off\_returns\_false\_if\_already\_off (LT-048)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Validates 'Turn off returns false if already off' behavior

**Input Specifications**

* Instantiate a fresh System that is still OFF and call turn\_off() immediately.

**Detailed Step**

1. Create System (OFF).
2. Call turn\_off().

**Expected Result**

* Operation returns False because lifecycle preconditions are not met.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn off returns false if already off' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

##### C. reset()

###### a. test\_reset\_from\_ready\_state (LT-033)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | reset() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-RST-001: Verifies that reset() from READY state succeeds and system returns to READY state.

**Input Specifications**

* Ensure SystemState is READY and call reset().

**Detailed Step**

1. Call reset().
2. Verify return value and final state.

**Expected Result**

* reset() reports success, completes all phases, and leaves state READY.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Reset from ready state' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### b. test\_reset\_when\_system\_off (LT-034)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | reset() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-RST-002: Verifies that reset() when system is OFF returns error and does not proceed.

**Input Specifications**

* Transition the system to OFF first, then call reset().

**Detailed Step**

1. Turn off system.
2. Call reset().

**Expected Result**

* reset() immediately fails with a phase 0 error because lifecycle preconditions are unmet.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Reset when system off' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### c. test\_reset\_preserves\_settings (LT-035)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | reset() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-RST-003: Verifies that settings are preserved after reset.

**Input Specifications**

* Adjust lock time and alarm delay values before invoking reset().

**Detailed Step**

1. Change settings.
2. Call reset().
3. Verify settings match initial changes.

**Expected Result**

* Post-reset settings remain at the modified values (no persistence loss).

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Reset preserves settings' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### d. test\_reset\_recreates\_components (LT-036)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | reset() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-RST-004: Verifies that all components are recreated after reset.

**Input Specifications**

* Capture references to current managers, call reset(), and inspect the refreshed component graph.

**Detailed Step**

1. Store refs to old components.
2. Call reset().
3. Compare new component refs.

**Expected Result**

* All core managers and controllers are reinitialized and non-null.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Reset recreates components' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### e. test\_reset\_deactivates\_sensors (LT-037)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | reset() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-RST-005: Verifies that sensors are deactivated during Phase 1.

**Input Specifications**

* Populate system.sensors with mock devices and call reset().

**Detailed Step**

1. Add mock sensors.
2. Call reset().
3. Verify deactivate() called.

**Expected Result**

* Each sensor is deactivated during Phase 1 before components restart.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Reset deactivates sensors' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### f. test\_reset\_logs\_out\_user (LT-038)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | reset() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-RST-006: Verifies that logged-in user is logged out during reset.

**Input Specifications**

* Authenticate admin to push the system into ACTIVE, then invoke reset().

**Detailed Step**

1. Login.
2. Call reset().
3. Verify user is logged out.

**Expected Result**

* reset() logs the user out and returns the lifecycle to READY.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Reset logs out user' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### g. test\_reset\_logs\_events (LT-039)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | reset() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-RST-007: Verifies that reset events are logged.

**Input Specifications**

* With LogManager ready, trigger reset().

**Detailed Step**

1. Call reset().
2. Check logs.

**Expected Result**

* Reset flow emits start/complete events for the audit trail.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Reset logs events' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

###### h. test\_reset\_state\_transitions (LT-040)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | reset() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

UT-SYS-RST-008: Verifies state transitions: READY -> OFF -> READY

**Input Specifications**

* Operate from READY and issue reset().

**Detailed Step**

1. Monitor state during reset().

**Expected Result**

* Lifecycle transitions READY -> OFF -> READY are observed without error.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Reset state transitions' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

## **4. Configuration and Data Management**

### **A. Log Management**

#### **1. X**

##### **A. X**

###### **a. X**

### **B. LoginManager (Backend Authentication Logic)**

#### **1. LoginManager class**

##### **A. login\_with\_details() (detailed login response)**

###### **a. test\_login\_with\_details\_success (UT-LM-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | LoginManager |
| Method | login\_with\_details() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that successful login returns detailed information including username and access level.

**Input Specifications**

• Username: "admin"

• Password: "1234"

• Interface: "control\_panel"

• Account is not locked

• Credentials are correct

**Expected Result**

{  
 'success': True,  
 'message': 'Login successful',  
 'username': 'admin',  
 'access\_level': 1  
 }

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns success with detailed information

• Pass: Username and access level included in response

**Comment**

Enhanced login method provides additional information for UI display and logging. Replaces boolean-only return value.

###### **b. test\_locked\_account\_with\_auto\_unlock (UT-LM-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | LoginManager |
| Method | login\_with\_details() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that the system automatically unlocks an account when the configured lock duration has elapsed.

**Input Specifications**

• User account locked at timestamp T

• system\_lock\_time = 30 seconds (default)

• Current time = T + 31 seconds

• User attempts login

**Expected Result**

• Account is automatically unlocked

• Login proceeds with credential validation

• If credentials correct, login succeeds

**Actual Result (Pass/Fail/Exception)**

• Pass: Account automatically unlocked after duration

• Pass: Console message: "Account 'admin' automatically unlocked after 30 seconds"

• Pass: Successful login after auto-unlock

**Comment**

Implements time-based automatic unlock feature. Uses \_check\_and\_unlock\_if\_time\_passed() internal method.

###### **c. test\_locked\_account\_returns\_remaining\_time (UT-LM-003)**

|  |  |
| --- | --- |
| Field | Value |
| Class | LoginManager |
| Method | login\_with\_details() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that attempting to login with a locked account returns the remaining lock time.

**Input Specifications**

• User account locked at timestamp T

• system\_lock\_time = 30 seconds

• Current time = T + 10 seconds

• Remaining time = 20 seconds

**Expected Result**

{  
 'success': False,  
 'message': 'Account is locked. Please try again in 20 seconds.',  
 'locked': True,  
 'tries': 5,  
 'remaining\_time': 20  
 }

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns locked status with remaining time

• Pass: Remaining time calculated correctly (30 - 10 = 20)

**Comment**

Provides user feedback on when they can attempt login again.

### **C. System Integration (Login & Configuration)**

#### **1. System class - Login Integration**

##### **A. login\_with\_details() (system-level login)**

###### **a. test\_login\_system\_off (UT-SYS-Login-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | login\_with\_details() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that login attempt fails when system is OFF.

**Input Specifications**

• System state: SystemState.OFF

• Valid credentials provided

**Expected Result**

{  
 'success': False,  
 'message': 'System is off. Please turn on the system first.',  
 'system\_off': True  
 }

**Actual Result (Pass/Fail/Exception)**

• Pass: Login fails with system\_off flag

• Pass: Appropriate error message returned

**Comment**

Precondition check: System must be ON for login.

###### **b. test\_login\_system\_locked (UT-SYS-Login-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | login\_with\_details() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that login attempt fails when system is LOCKED.

**Input Specifications**

• System state: SystemState.LOCKED

• Valid credentials provided

**Expected Result**

{  
 'success': False,  
 'message': 'System is locked. Please contact administrator.',  
 'system\_locked': True  
 }

**Actual Result (Pass/Fail/Exception)**

• Pass: Login fails with system\_locked flag

• Pass: Appropriate error message returned

**Comment**

System-level lock prevents all logins regardless of user credentials.

#### **2. System class - Configuration Integration**

##### **A. configure\_system\_setting() (system-level configuration)**

###### **a. test\_configure\_system\_setting (UT-SYS-Config-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | configure\_system\_setting() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that system settings can be configured through the System class.

**Input Specifications**

• System is ON

• monitoring\_phone: "111-222-3333"

• homeowner\_phone: "444-555-6666"

• lock\_time: 45

• alarm\_delay: 90

**Expected Result**

• Method returns True

• Settings are persisted to database

• SettingsService is invoked correctly

**Actual Result (Pass/Fail/Exception)**

• Pass: Returns True

• Pass: Settings saved successfully

• Pass: Database contains updated values

**Comment**

Uses SettingsService internally for separation of concerns.

### **D. Password Management**

#### **1. System class - Password Change**

##### **A. change\_master\_password() (system-level password change)**

###### **a. test\_change\_password\_success (UT-PWD-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | change\_master\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that password change succeeds with correct inputs through all 3 phases.

**Input Specifications**

• User is logged in (admin/1234)

• current\_password: '1234'

• new\_password: 'newpass5678'

• confirm\_password: 'newpass5678'

**Expected Result**

{  
 'success': True,  
 'message': 'Password changed successfully',  
 'phase': 3  
 }

• Password is updated in database

• User remains logged in

**Actual Result (Pass/Fail/Exception)**

• Pass: Password change succeeds with phase 3

• Pass: New password works for subsequent login

**Comment**

Complete 3-phase password change flow. Test file: test\_change\_master\_password.py:68

###### **b. test\_change\_password\_wrong\_current (UT-PWD-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | change\_master\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that password change fails with wrong current password at Phase 1.

**Input Specifications**

• User is logged in

• current\_password: 'wrongpass' (incorrect)

• new\_password: 'newpass5678'

• confirm\_password: 'newpass5678'

**Expected Result**

{  
 'success': False,  
 'phase': 1,  
 'error\_type': 'INCORRECT\_PASSWORD',  
 'message': 'Current password is incorrect'  
 }

**Actual Result (Pass/Fail/Exception)**

• Pass: Fails at Phase 1 with INCORRECT\_PASSWORD error

• Pass: Password remains unchanged

**Comment**

Phase 1 validation prevents unauthorized password changes. Test file: test\_change\_master\_password.py:91

###### **c. test\_change\_password\_mismatch (UT-PWD-003)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | change\_master\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that password change fails when new passwords don't match at Phase 3.

**Input Specifications**

• User is logged in

• current\_password: '1234'

• new\_password: 'newpass1111'

• confirm\_password: 'newpass2222' (mismatch)

**Expected Result**

{  
 'success': False,  
 'phase': 3,  
 'error\_type': 'PASSWORD\_MISMATCH',  
 'message': 'New passwords do not match'  
 }

**Actual Result (Pass/Fail/Exception)**

• Pass: Fails at Phase 3 with PASSWORD\_MISMATCH error

• Pass: Password remains unchanged

**Comment**

Phase 3 confirmation prevents accidental password changes. Test file: test\_change\_master\_password.py:108

###### **d. test\_change\_password\_same\_as\_current (UT-PWD-004)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | change\_master\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that password change fails when new password is same as current.

**Input Specifications**

• current\_password: '1234'

• new\_password: '1234'

• confirm\_password: '1234'

**Expected Result**

{  
 'success': False,  
 'error\_type': 'SAME\_PASSWORD',  
 'message': 'New password must be different from current password'  
 }

**Actual Result (Pass/Fail/Exception)**

• Pass: Fails with SAME\_PASSWORD error

• Pass: Password remains unchanged

**Comment**

Security policy requires different password. Test file: test\_change\_master\_password.py:125

###### **e. test\_change\_password\_not\_authenticated (UT-PWD-005)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | change\_master\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that password change fails when user is not authenticated.

**Input Specifications**

• User is NOT logged in

• Attempt to change password

**Expected Result**

{  
 'success': False,  
 'phase': 0,  
 'error\_type': 'NOT\_AUTHENTICATED',  
 'message': 'Authentication required to change password'  
 }

**Actual Result (Pass/Fail/Exception)**

• Pass: Fails at Phase 0 with NOT\_AUTHENTICATED error

**Comment**

Precondition: User must be logged in. Test file: test\_change\_master\_password.py:141

###### **f. test\_change\_password\_system\_off (UT-PWD-006)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | change\_master\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that password change fails when system is OFF.

**Input Specifications**

• System is OFF

• Attempt to change password

**Expected Result**

{  
 'success': False,  
 'error\_type': 'SYSTEM\_OFF',  
 'message': 'System is off. Please turn on the system first.'  
 }

**Actual Result (Pass/Fail/Exception)**

• Pass: Fails with SYSTEM\_OFF error

**Comment**

Precondition: System must be ON. Test file: test\_change\_master\_password.py:162

###### **g. test\_change\_password\_system\_locked (UT-PWD-007)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | change\_master\_password() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**

Verifies that password change fails when system is LOCKED.

**Input Specifications**

• System is LOCKED

• Attempt to change password

**Expected Result**

{  
 'success': False,  
 'error\_type': 'SYSTEM\_LOCKED',  
 'message': 'System is locked. Please contact administrator.'  
 }

**Actual Result (Pass/Fail/Exception)**

• Pass: Fails with SYSTEM\_LOCKED error

**Comment**

Precondition: System must not be locked. Test file: test\_change\_master\_password.py:182

### **E. System Power Management**

#### **1. System class - Power On**

##### **A. turn\_on() (system power on)**

###### **a. test\_turn\_on\_from\_off\_state (UT-SYS-ON-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | turn\_on() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.1 |

**Test Case Description**

Verifies that calling turn\_on() when system is OFF results in successful initialization and system state becomes READY.

**Input Specifications**

• System is in OFF state (initial state after constructor)

**Expected Result**

• turn\_on() returns True

• system\_state becomes SystemState.READY

• All core components are initialized (not None):

• StorageManager

• ConfigurationManager

• LoginManager

• LogManager

• SystemController

• SecuritySystem

• CameraController

• Siren

**Actual Result (Pass/Fail/Exception)**

• Pass: turn\_on() returns True

• Pass: system\_state is SystemState.READY

• Pass: All components initialized

**Comment**

Complete system initialization sequence as specified in SDS page 50. Includes initialization of AuthService and SettingsService (new in Phase 2).

###### **b. test\_turn\_on\_when\_already\_running (UT-SYS-ON-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | turn\_on() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.1 |

**Test Case Description**

Verifies that calling turn\_on() when system is already running returns False and does not reinitialize components.

**Input Specifications**

• System is already in READY state (turn\_on() called previously)

**Expected Result**

• turn\_on() returns False

• system\_state remains READY

• No exception is raised

• Console message: "System is already Ready"

**Actual Result (Pass/Fail/Exception)**

• Pass: turn\_on() returns False

• Pass: system\_state remains READY

• Pass: Components are not reinitialized

**Comment**

Error handling prevents double initialization.

#### **2. System class - Power Off**

##### **A. turn\_off() (system power off)**

###### **a. test\_turn\_off\_from\_ready\_state (UT-SYS-OFF-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | turn\_off() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.1 |

**Test Case Description**

Verifies that calling turn\_off() when system is READY results in successful shutdown following the complete sequence.

**Input Specifications**

• System is in READY state (after turn\_on() was called)

**Expected Result**

• turn\_off() returns True

• system\_state becomes SystemState.OFF

• Shutdown sequence executed:

1. Configuration saved

2. Sensors deactivated

3. Cameras disabled

4. Siren deactivated

5. User logged out

6. SYSTEM\_SHUTDOWN event logged

7. Database disconnected

**Actual Result (Pass/Fail/Exception)**

• Pass: turn\_off() returns True

• Pass: system\_state is SystemState.OFF

• Pass: All shutdown steps completed

**Comment**

Complete shutdown sequence as specified in SDS page 51. Test file: test\_system\_power\_flow.py:115

###### **b. test\_turn\_off\_when\_already\_off (UT-SYS-OFF-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | turn\_off() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.1 |

**Test Case Description**

Verifies that calling turn\_off() when system is already OFF returns False and does not cause errors.

**Input Specifications**

• System is already in OFF state

**Expected Result**

• turn\_off() returns False

• system\_state remains OFF

• No exception is raised

• Console message: "System is already off"

**Actual Result (Pass/Fail/Exception)**

• Pass: turn\_off() returns False

• Pass: system\_state remains OFF

• Pass: No exception raised

**Comment**

Prevents redundant shutdown operations. Test file: test\_system\_power\_flow.py:133

###### **c. test\_turn\_off\_deactivates\_sensors (UT-SYS-OFF-003)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | turn\_off() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.1 |

**Test Case Description**

Verifies that all sensors are deactivated during shutdown (Phase 2 of turn\_off sequence).

**Input Specifications**

• System is running with multiple sensors registered

**Expected Result**

• All sensors are deactivated

• turn\_off() returns True

• Console message: "N sensors deactivated"

**Actual Result (Pass/Fail/Exception)**

• Pass: All sensors deactivated

• Pass: Each sensor's deactivate() method called

**Comment**

Phase 2 of Turn the System Off sequence. Test file: test\_system\_power\_flow.py:115

###### **d. test\_turn\_off\_disables\_cameras (UT-SYS-OFF-004)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | turn\_off() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.1 |

**Test Case Description**

Verifies that all cameras are disabled during shutdown (Phase 3 of turn\_off sequence).

**Input Specifications**

• System is running with CameraController and cameras added

**Expected Result**

• CameraController.disable\_all\_camera() is called

• All cameras in SystemController are deactivated

• turn\_off() returns True

**Actual Result (Pass/Fail/Exception)**

• Pass: Camera controller disabled flag is True

• Pass: All camera objects deactivated

**Comment**

Phase 3 of Turn the System Off sequence. Test file: test\_system\_power\_flow.py:115

#### **3. System class - System Reset**

##### **A. reset() (system reset)**

###### **a. test\_reset\_from\_ready\_state (UT-SYS-RST-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | reset() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.1 |

**Test Case Description**

Verifies that calling reset() from READY state succeeds and system returns to READY state.

**Input Specifications**

• System is in READY state (after turn\_on() was called)

**Expected Result**

{  
 'success': True,  
 'message': 'System reset successfully',  
 'phase': 3  
 }

• system\_state becomes SystemState.READY

**Actual Result (Pass/Fail/Exception)**

• Pass: reset() returns success with phase 3

• Pass: system\_state is SystemState.READY

• Pass: Reset = Turn Off + Turn On

**Comment**

Reset executes turn\_off() followed by turn\_on(). Enhanced return format includes phase information.

###### **b. test\_reset\_when\_system\_off (UT-SYS-RST-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | reset() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.1 |

**Test Case Description**

Verifies that calling reset() when system is OFF returns error and does not proceed.

**Input Specifications**

• System is in OFF state

**Expected Result**

{  
 'success': False,  
 'message': 'Cannot reset: System is not running',  
 'phase': 0  
 }

**Actual Result (Pass/Fail/Exception)**

• Pass: reset() returns False with phase 0

• Pass: Error message indicates system not running

**Comment**

Precondition: System must be ON to perform reset.

###### **c. test\_reset\_preserves\_settings (UT-SYS-RST-003)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System |
| Method | reset() |
| Author | Minseok Jo |
| Date | 2025-11-25 |
| Version | 2.0.1 |

**Test Case Description**

Verifies that settings are preserved after reset.

**Input Specifications**

• System is in READY state

• Settings modified:

• lock\_time = 45

• alarm\_delay = 90

**Expected Result**

• reset() succeeds

• Modified settings are preserved after reset

• lock\_time remains 45

• alarm\_delay remains 90

**Actual Result (Pass/Fail/Exception)**

• Pass: reset() returns success

• Pass: Settings values preserved

**Comment**

Settings are saved during Turn Off (Phase 1) and loaded during Turn On (Phase 2).

# **IV. Integration Tests**

## **A. Control Panel Login Process**

### **1. test\_control\_panel\_login\_with\_presenter\_and\_system (IT-CP-Login-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanelLoginPresenter, System, LoginManager, StorageManager |
| Function | Log onto the system via Control Panel |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate that ControlPanelLoginPresenter successfully coordinates with System and LoginManager to authenticate a user when valid credentials are provided.

**Input Specifications**

• Database has predefined username and password: "admin" and "1234"

• User ID: "admin"

• Password: "1234"

**Detailed Step**

1. Initialize System and call turn\_on() to initialize all components including LoginManager and StorageManager.

2. Create ControlPanelLoginPresenter with reference to the System instance.

3. Call presenter.attempt\_login() with "admin" as user\_id and "1234" as password.

4. Presenter delegates to System.login\_with\_details() which internally calls LoginManager.login\_with\_details().

5. LoginManager validates credentials against StorageManager data.

6. System returns success response with navigation information.

7. Presenter transforms response into LoginOutcome dataclass.

**Expected Result**

• LoginOutcome.success equals True

• LoginOutcome.navigate\_to equals "MainMenu"

• System.login\_tries equals 0 after successful authentication

• User session is established in System

**Actual Result (Pass/Fail/Exception)**

• Pass: LoginOutcome.success equals True

• Pass: LoginOutcome.navigate\_to equals "MainMenu"

• Pass: System.login\_tries equals 0

**Comment (including references)**

Integration between Presenter layer and System/LoginManager layer. Validates the MVP pattern implementation introduced in Phase 2.

### **2. test\_control\_panel\_login\_lockout\_integration (IT-CP-Login-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanelLoginPresenter, System, LoginManager |
| Function | Account lockout after failed attempts |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate that the integration between ControlPanelLoginPresenter and LoginManager correctly handles account lockout after maximum failed attempts.

**Input Specifications**

• Database has predefined username: "admin" with password: "1234"

• Incorrect password attempts: 5 times with "wrongpw"

• system\_lock\_time: 30 seconds

**Detailed Step**

1. Initialize System with turn\_on().

2. Create ControlPanelLoginPresenter with System reference.

3. Attempt login 5 times with incorrect password "wrongpw".

4. On each failed attempt, LoginManager increments tries counter.

5. After 5th failed attempt, LoginManager locks the account.

6. 6th attempt returns locked status with remaining\_time.

7. Presenter formats the lock message with time remaining.

**Expected Result**

• After 5 failed attempts, account is locked

• LoginOutcome.success equals False

• LoginOutcome.status\_color equals "red"

• LoginOutcome.alert\_level equals "error"

• LoginOutcome.alert\_title equals "Account Locked"

• remaining\_time is included in status\_text

**Actual Result (Pass/Fail/Exception)**

• Pass: Account locked after 5 attempts

• Pass: LoginOutcome contains lock information

• Pass: Time remaining is formatted correctly

**Comment (including references)**

Tests the complete lockout flow from UI through business logic. Implements time-based unlock feature.

## **B. Control Panel Settings Configuration**

### **1. test\_settings\_save\_integration (IT-CP-Settings-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanelSettingsPresenter, System, ConfigurationManager, StorageManager |
| Function | Save system settings via Control Panel |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate that ControlPanelSettingsPresenter correctly saves settings through the System layer and persists them to the database.

**Input Specifications**

• monitoring\_phone: "911"

• homeowner\_phone: "010-1234-5678"

• lock\_time: "45"

• alarm\_delay: "90"

**Detailed Step**

1. Initialize System with turn\_on().

2. Create ControlPanelSettingsPresenter with System reference.

3. Call presenter.save\_settings() with the input values.

4. Presenter validates inputs (non-numeric check, negative check).

5. Presenter converts string time values to integers.

6. Presenter calls System.configure\_system\_setting().

7. System delegates to SettingsService which updates ConfigurationManager.

8. ConfigurationManager persists changes via StorageManager.

**Expected Result**

• SettingsOutcome.success equals True

• SettingsOutcome.alert\_level equals "info"

• ConfigurationManager.lock\_time equals 45

• ConfigurationManager.alarm\_delay equals 90

• Database contains updated values

**Actual Result (Pass/Fail/Exception)**

• Pass: SettingsOutcome.success equals True

• Pass: Settings persisted correctly

• Pass: Values retrievable after save

**Comment (including references)**

Tests complete settings flow from UI validation through persistence layer.

## **C. Control Panel Password Change**

### **1. test\_password\_change\_full\_flow\_integration (IT-CP-PWD-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanelChangePasswordPresenter, System, LoginManager, StorageManager |
| Function | Change master password via Control Panel |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate the complete password change flow from ControlPanelChangePasswordPresenter through System to database persistence.

**Input Specifications**

• User is logged in as "admin"

• current\_password: "1234"

• new\_password: "newpass5678"

• confirm\_password: "newpass5678"

**Detailed Step**

1. Initialize System and login with admin/1234.

2. Create ControlPanelChangePasswordPresenter with System reference.

3. Call presenter.change\_password() with current, new, and confirm passwords.

4. Presenter validates: passwords match, minimum length met.

5. Presenter calls System.change\_master\_password().

6. System executes 3-phase password change:

• Phase 1: Verify current password

• Phase 2: Validate new password rules

• Phase 3: Confirm and save

7. StorageManager updates password in database.

**Expected Result**

• ChangePasswordOutcome.success equals True

• ChangePasswordOutcome.message equals "Password changed successfully!"

• New password works for subsequent login

• Old password no longer works

**Actual Result (Pass/Fail/Exception)**

• Pass: Password change succeeds

• Pass: New password authenticates correctly

• Pass: Old password rejected

**Comment (including references)**

Validates complete 3-phase password change flow. Test file: test\_change\_master\_password.py

## **D. Web Login Process (2FA)**

### **1. test\_web\_login\_two\_factor\_integration (IT-Web-Login-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter, System, LoginManager, StorageManager |
| Function | Two-factor authentication via Web interface |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate the complete 2-factor authentication flow for web login, including first password and second password/PIN validation.

**Input Specifications**

• username: "admin"

• first\_password: "password123"

• second\_password: "pin1234"

**Detailed Step**

1. Initialize System with turn\_on().

2. Create WebLoginPresenter with System reference.

3. Call presenter.validate\_first\_password() with username and first\_password.

4. Presenter calls LoginManager.validate\_first\_password() with interface='web\_browser'.

5. LoginManager validates first credential against StorageManager.

6. On success, call presenter.validate\_second\_password() with username and second\_password.

7. LoginManager validates second credential (PIN).

8. On success, user session is established and redirect URL is returned.

**Expected Result**

• First password validation: WebLoginOutcome.success equals True

• Second password validation: WebLoginOutcome.success equals True

• WebLoginOutcome.redirect\_url equals "/dashboard"

• User session established in System

**Actual Result (Pass/Fail/Exception)**

• Pass: First password validated successfully

• Pass: Second password validated successfully

• Pass: Redirect to dashboard

**Comment (including references)**

Tests complete 2FA flow for web interface. Implements separate tracking for each authentication factor.

### **2. test\_web\_login\_lockout\_across\_factors (IT-Web-Login-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebLoginPresenter, System, LoginManager |
| Function | Account lockout during 2FA |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate that account lockout works correctly across both authentication factors in 2FA.

**Input Specifications**

• username: "admin"

• Incorrect first\_password attempts: 5 times

• system\_lock\_time: 30 seconds

**Detailed Step**

1. Initialize System with turn\_on().

2. Create WebLoginPresenter.

3. Attempt first password validation 5 times with wrong password.

4. Verify account gets locked after 5 failed attempts.

5. Verify that even correct credentials are rejected while locked.

6. Verify remaining\_time is returned in response.

**Expected Result**

• Account locked after 5 failed first password attempts

• WebLoginOutcome.locked equals True

• WebLoginOutcome.remaining\_time shows seconds until unlock

• Correct credentials rejected while locked

**Actual Result (Pass/Fail/Exception)**

• Pass: Account locked correctly

• Pass: Remaining time calculated

• Pass: Locked state enforced

**Comment (including references)**

Tests lockout behavior specific to web 2FA authentication.

## **E. Web Settings Configuration**

### **1. test\_web\_settings\_update\_integration (IT-Web-Settings-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebSettingsPresenter, System, ConfigurationManager, StorageManager |
| Function | Update system settings via Web interface |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate that WebSettingsPresenter correctly validates and saves settings through the System layer.

**Input Specifications**

• monitoring\_service\_phone: "111-222-3333"

• homeowner\_phone: "444-555-6666"

• system\_lock\_time: 30

• alarm\_delay\_time: 60

**Detailed Step**

1. Initialize System with turn\_on().

2. Create WebSettingsPresenter with System reference.

3. Call presenter.update\_settings() with all parameters.

4. Presenter validates:

• Phone number formats

• Lock time > 0

• Alarm delay >= 0

• Type checking for numeric fields

5. Presenter calls System.configure\_system\_setting().

6. Settings are persisted to database.

**Expected Result**

• WebSettingsOutcome.success equals True

• WebSettingsOutcome.message equals "Settings updated successfully"

• All values persisted correctly

• No validation errors

**Actual Result (Pass/Fail/Exception)**

• Pass: Settings saved successfully

• Pass: Values retrievable via get\_settings()

**Comment (including references)**

Tests web-specific validation rules including phone format validation.

## **F. System Power Management**

### **1. test\_system\_power\_cycle\_integration (IT-SYS-Power-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System, StorageManager, ConfigurationManager, LoginManager, SecuritySystem, CameraController |
| Function | Complete power cycle (On -> Off -> On) |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate that System correctly initializes and shuts down all components during power cycle.

**Input Specifications**

• System starts in OFF state

• Modified settings: lock\_time=45, alarm\_delay=90

**Detailed Step**

1. Create System instance (starts in OFF state).

2. Call turn\_on() - Phase 1: Initialize all components.

3. Verify all managers and controllers are initialized.

4. Modify settings: lock\_time=45, alarm\_delay=90.

5. Call turn\_off() - Execute shutdown sequence:

• Save configuration

• Deactivate sensors

• Disable cameras

• Deactivate siren

• Logout user

• Log SYSTEM\_SHUTDOWN

• Disconnect database

6. Call turn\_on() again.

7. Verify settings are preserved.

**Expected Result**

• First turn\_on(): Returns True, state becomes READY

• turn\_off(): Returns True, state becomes OFF

• Second turn\_on(): Returns True, state becomes READY

• Settings preserved: lock\_time=45, alarm\_delay=90

**Actual Result (Pass/Fail/Exception)**

• Pass: Power cycle completes successfully

• Pass: All components initialized/shutdown correctly

• Pass: Settings preserved across cycle

**Comment (including references)**

Tests complete initialization and shutdown sequences as specified in SDS pages 50-51.

### **2. test\_system\_reset\_integration (IT-SYS-Reset-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | System, ControlPanelResetPresenter, all managers |
| Function | System reset via Control Panel |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate that system reset correctly executes turn\_off followed by turn\_on and preserves settings.

**Input Specifications**

• System is running (READY state)

• User is logged in

• Custom settings configured

**Detailed Step**

1. Initialize System and turn\_on().

2. Login as admin.

3. Configure custom settings.

4. Create ControlPanelResetPresenter.

5. Call presenter.perform\_reset().

6. Presenter calls System.reset().

7. System executes turn\_off() then turn\_on().

8. Verify system returns to READY state.

9. Verify settings are preserved.

10. Verify user is logged out (must re-authenticate).

**Expected Result**

• ResetOutcome.success equals True

• ResetOutcome.phase equals 3

• System state equals READY

• Settings preserved

• User session cleared (logged out)

**Actual Result (Pass/Fail/Exception)**

• Pass: Reset completes successfully

• Pass: Settings preserved

• Pass: User logged out

**Comment (including references)**

Tests reset as combination of turn\_off and turn\_on sequences.

## **G. LoginManager Backend Integration**

### **1. test\_login\_manager\_auto\_unlock\_integration (IT-LM-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | LoginManager, StorageManager, ConfigurationManager |
| Function | Automatic account unlock after time elapsed |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate that LoginManager automatically unlocks an account when the configured lock duration has elapsed.

**Input Specifications**

• Account locked at timestamp T

• system\_lock\_time: 30 seconds

• Current time: T + 31 seconds

**Detailed Step**

1. Initialize LoginManager with StorageManager and ConfigurationManager.

2. Simulate 5 failed login attempts to lock account.

3. Record lock timestamp.

4. Advance time by 31 seconds (mock time).

5. Attempt login with correct credentials.

6. LoginManager checks \_check\_and\_unlock\_if\_time\_passed().

7. Account is automatically unlocked.

8. Login proceeds and succeeds.

**Expected Result**

• Account automatically unlocked after 30 seconds

• Console message: "Account 'admin' automatically unlocked after 30 seconds"

• Login succeeds with correct credentials

**Actual Result (Pass/Fail/Exception)**

• Pass: Auto-unlock triggered

• Pass: Login succeeds after unlock

**Comment (including references)**

Tests time-based automatic unlock feature implementation.

## **H. Surveillance Integration Tests**

### a. test\_camera\_lifecycle (IT-CAM-LIFECYCLE)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController, SafeHomeCamera | add\_camera(), set\_camera\_password(), disable\_camera(), control\_single\_camera(), delete\_camera() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the complete integrated lifecycle of a single camera: successful addition, password setup and validation, state change (enable/disable), control execution (capture/record), and final deletion."

Input Specifications

"Sequential operations starting with add\_camera(100, 200) and including password setting (""test123""), control commands (0, 1), and delete\_camera()."

**Detailed Step**

1. Add camera (ID 1).
2. Set/validate password.
3. Disable/re-enable state and check view denial.
4. Execute control commands (record start/stop).
5. Delete the camera and check count is 0.

Expected Result

"All sequential operations must return True (or success code 0). State changes (e.g., view denial when disabled, record status update) must be correctly observed."

Actual Result (Pass/Fail/Exception)

Pass: The full camera lifecycle succeeded without error.

Comment (including references)

Fundamental integration test verifying the collaboration between the Controller and the Camera object across all major states.

### b. test\_camera\_password\_workflow\_comprehensive (IT-CAM-PW-FLOW)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController, SafeHomeCamera | set\_camera\_password(), validate\_camera\_password(), delete\_camera() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies the comprehensive password management workflow, checking all possible validation codes (-2, 0, 1) and ensuring the password data is cleanly removed upon camera deletion (-1 returned for validation after deletion)."

Input Specifications

"Actions include validation before setting, setting ""secure123"", validating with success/failure inputs, and deleting the camera."

**Detailed Step**

1. Assert -2 (Not Set).
2. Set password.
3. Assert 0 (Match) and 1 (Mismatch).
4. Delete camera.
5. Assert final validation returns -1 (Nonexistent).

Expected Result

"Validation must return the correct sequence: -2, 0, 1, and finally -1. Password data must be cleaned up upon deletion."

Actual Result (Pass/Fail/Exception)

Pass: Password workflow and data cleanup succeeded according to security codes.

Comment (including references)

Critical security integration test verifying all validation paths and data consistency during de-registration.

### c. test\_camera\_lifecycle\_with\_security\_events (IT-CAM-SEC-EVENT)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController, SafeHomeCamera | add\_camera(), trigger\_security\_event(), disable\_camera() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that the CameraController correctly dispatches the take\_picture command to only the currently enabled cameras upon receiving a security event (""INTRUSION"", ""PANIC"")."

Input Specifications

Initial state: Two cameras exist. Actions: Trigger event (all enabled); disable one camera; trigger event again.

**Detailed Step**

1. Add 2 cameras.
2. Trigger event, assert 2 calls to take\_picture (mocked).
3. Disable camera 1.
4. Trigger event again, assert 1 call to take\_picture.

Expected Result

The mocked take\_picture function must be called only for cameras whose state is enabled=True at the time of the event.

Actual Result (Pass/Fail/Exception)

Pass: Security event dispatch correctly respected the camera's enabled/disabled status.

Comment (including references)

Integration test verifying controller state awareness and dispatch filtering for security operations.

### d. test\_camera\_view\_integration (IT-CAM-VIEW-CTRL)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController, SafeHomeCamera | display\_single\_view(), display\_thumbnail\_view(), disable\_camera() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Verifies that CameraController successfully integrates view retrieval from multiple cameras (single view and thumbnail view) and correctly enforces view denial when a camera is disabled.

Input Specifications

"Actions include retrieving single views for IDs 1 and 2, retrieving the composite thumbnail view, and checking the single view for ID 1 after disabling it."

**Detailed Step**

1. Add 2 cameras.
2. Retrieve both single views and the composite view.
3. Disable camera 1.
4. Assert display\_single\_view(1) returns None.

Expected Result

"View retrieval must succeed when enabled, and return None when disabled. Both single and thumbnail views must execute without error."

Actual Result (Pass/Fail/Exception)

"Pass: View and state integration succeeded, and view denial was confirmed."

Comment (including references)

Integration test verifying the display interface and state management for view access control.

### e. test\_camera\_info\_consistency\_comprehensive (IT-CAM-INFO-CONS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | add\_camera(), get\_all\_camera\_info(), disable\_camera() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Verifies that the global information retrieved via get\_all\_camera\_info() remains consistent and accurate, reflecting the current state (location, enabled status) even after multiple state changes (e.g., disabling individual cameras)."

Input Specifications

Initial state: Four cameras added. Actions: Retrieve info; disable cameras 2 and 4; retrieve info again.

**Detailed Step**

1. Add 4 cameras and retrieve info.
2. Assert initial enabled status (1).
3. Disable cameras 2 and 4.
4. Retrieve info again and assert status is 0 for IDs 2 and 4.

Expected Result

The get\_all\_camera\_info() list must accurately report the location and the updated enabled status (0 or 1) for all cameras.

Actual Result (Pass/Fail/Exception)

Pass: Global state retrieval was consistent and accurate after state changes.

Comment (including references)

Comprehensive integration test ensuring data integrity and consistency between the controller's internal state and the exposed information.

## I. Security Integration Tests

| ID | Location | Scenario | Steps | Expected Result | Trace |
| --- | --- | --- | --- | --- | --- |
| IT-UC2-CTRL-ALARM | tests/test\_controller\_security\_flow.py:73 | Controller-driven alarm cycle | Arm AWAY via controller, emit door event, advance time beyond delay | DISARMED -> ENTRY\_DELAY -> ALARM\_ACTIVE, monitoring INTRUSION\_ALARM | SDS VI.2a/VI.2d |
| IT-UC2-CTRL-DISARM | tests/test\_controller\_security\_flow.py:96 | Disarm during entry delay | Arm STAY, trigger entry event, disarm before deadline | Alarm returns to IDLE, siren off | SRS UC2d alt |
| IT-UC2-PANIC | tests/test\_controller\_security\_flow.py:119 | Panic integration | Invoke panic while disarmed | Immediate ALARM\_ACTIVE, monitoring PANIC | SRS UC2h |
| IT-UC2-LIFECYCLE | tests/common\_tests/test\_turn\_system\_on.py:579 / est\_turn\_system\_off.py:318 | Full turn-on/off flows | Execute start/stop suites and inspect subsystem wiring | Modules initialized/disposed per SDS lifecycle diagrams | SDS startup/shutdown |
| IT-UC2-RESTSYNC | QA Postman run 2025-11-24 | Web REST vs Tk panel sync | Arm/disarm via /api/security/\* while Tk simulator emits events | /api/security/status + intrusion logs mirror Tk view | SDS VI.2b/VI.2j |

**Per-test detail (IT-UC2)**

### 1. Alarm test

#### a. Alarm test

| Field | Value |
| --- | --- |
| Classes / Modules | SystemController, SecuritySystem, MonitoringStub |
| Functions | set\_security\_mode, handle\_sensor\_event, tick |
| Author | Jonghwa An |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**  
Validates that controller-driven arming propagates through the full FSM, including entry delay timers and monitoring callbacks.

**Input Specifications**  
- Start system in DISARMED state.  
- Authenticated session established.  
- Door sensor assigned to perimeter zone.

**Detailed Steps**  
1. Invoke set\_security\_mode("AWAY") via SystemController.  
2. Emit a mocked door SensorEvent (status OPEN).  
3. Advance fake clock past configured entry delay by calling tick() repeatedly.

**Expected Result**  
- State transitions DISARMED -> ENTRY\_DELAY -> ALARM\_ACTIVE.  
- Monitoring stub records INTRUSION\_ALARM.  
- Event log contains countdown and alarm entries.

**Actual Result (Pass/Fail/Exception)**  
- Pass: assertions capture state changes, monitoring call payload, and log entries.

**Comment (including references)**  
SDS VI.2d AWAY main flow; SRS UC2c.

#### b. Disarm tests/test\_controller\_security\_flow.py:96

| Field | Value |
| --- | --- |
| Classes / Modules | SystemController, SecuritySystem |
| Functions | set\_security\_mode, handle\_sensor\_event, cancel\_entry\_delay |
| Author | Jonghwa An |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**  
Demonstrates disarming during ENTRY\_DELAY suppresses alarms and resets readiness.

**Input Specifications**  
- Logged-in user.  
- STAY mode selected.  
- Perimeter door sensor configured.

**Detailed Steps**  
1. Call set\_security\_mode("STAY").  
2. Emit door OPEN event to enter ENTRY\_DELAY.  
3. Before timer expires, invoke set\_security\_mode("DISARMED").

**Expected Result**  
- Countdown cancelled; siren remains idle.  
- State returns to DISARMED.  
- Monitoring stub receives no alarm.  
- Event log marks intrusion as cancelled.

**Actual Result (Pass/Fail/Exception)**  
- Pass: timer handle cleared, monitoring call absent, log shows cancellation.

**Comment (including references)**  
SRS UC2d alternate course; SDS VI.2d stay-mode timing.

#### c. Panic test

| Field | Value |
| --- | --- |
| Classes / Modules | Flask /api/security/panic, SystemController, SecuritySystem |
| Functions | trigger\_panic, monitoring dispatch |
| Author | Jonghwa An |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**  
Ensures REST panic endpoint bypasses entry delays and instantly escalates to monitoring and cameras.

**Input Specifications**  
- System DISARMED.  
- Flask test client authenticated.

**Detailed Steps**  
1. Issue POST /api/security/panic via test client.  
2. Inspect controller panic handler invocation.  
3. Verify monitoring stub and camera controller hooks receive panic reason.

**Expected Result**  
- alarm\_state switches to ALARM\_ACTIVE immediately.  
- Monitoring called with reason="PANIC".  
- Camera controller trigger\_security\_event invoked.  
- Response payload returns HTTP 200 with panic acknowledgment.

**Actual Result (Pass/Fail/Exception)**  
- Pass: all hooks triggered in same tick; response payload matches expectation.

**Comment (including references)**  
SRS UC2g/UC2h; SDS VI.2h panic escalation pathway.

#### d. Lifecycle test

| Field | Value |
| --- | --- |
| Classes / Modules | System (turn\_on/turn\_off), ConfigurationManager, StorageManager |
| Functions | turn\_on, turn\_off, lifecycle callbacks |
| Author | Jonghwa An |
| Date | 2025-11-25 |
| Version | 2.0.0 |

**Test Case Description**  
Verifies startup and shutdown orchestration initializes and tears down every UC2 subsystem in order.

**Input Specifications**  
- Fresh System instance with temporary SQLite DB.  
- Mocked device/camera factories registered.

**Detailed Steps**  
1. Call turn\_on() and capture component references.  
2. Assert each manager/controller is non-null and wired.  
3. Invoke turn\_off() and confirm sensors, cameras, siren, DB connections are cleaned up.

**Expected Result**  
- turn\_on() returns True with state READY.  
- All managers initialized (login, config, storage, controller, security, cameras, log).  
- turn\_off() returns True, state OFF, and cleanup log entries recorded.

**Actual Result (Pass/Fail/Exception)**  
- Pass: lifecycle tests confirm initialization order and teardown artifacts.

**Comment (including references)**  
SDS startup/shutdown diagrams; SRS UC2j logging expectations.

#### e. Restsync

| Field | Value |
| --- | --- |
| Interfaces | Flask /api/security/\*, Tk Control Panel simulator |
| Tools | Postman collection + Tk harness |
| Author | Jonghwa An |
| Date | 2025-11-24 |
| Version | 1.0.0 |

**Test Case Description**  
Demonstrates that REST endpoints and the Tk Control Panel stay synchronized when commands are issued from both surfaces.

**Input Specifications**  
- Running SafeHome system with seeded sensors.  
- Postman scripts for /api/security/arm, /api/security/status, /api/security/disarm.  
- Tk simulator capable of emitting sensor events.

**Detailed Steps**  
1. Arm STAY via /api/security/arm and confirm Tk UI reflects STAY.  
2. Trigger door event from Tk; poll /api/security/status for ENTRY\_DELAY countdown.  
3. Disarm from Tk keypad; verify REST status returns DISARMED and intrusion log shows cancellation.  
4. Repeat by arming via Tk and polling REST responses for parity.

**Expected Result**  
- REST status JSON matches Tk indicator lights at each step.  
- Intrusion log entries align between dashboard and Tk view.  
- No desynchronization even when commands originate from different clients.

**Actual Result (Pass/Fail/Exception)**  
- Pass: QA capture shows identical timestamps between /api/security/status payloads and Tk screenshot overlays.

**Comment (including references)**  
SDS VI.2b/VI.2j requirement for UI/API parity; evidence stored in QA Postman report (docs/UC2\_manual\_logs/RESTSYNC.pdf).

### 2. System Lifecycle Integration

#### a. test\_turn\_on\_component\_dependencies (LT-011)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_on() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Validate that all components are initialized in correct order and can communicate with each other.

**Input Specifications**

* System is in OFF state.
* Steps: Initialize, turn\_on(), verify creation order and interactions.

**Detailed Step**

1. Call turn\_on().
2. Check StorageManager > ConfigManager > LoginManager sequence.

**Expected Result**

* All components are initialized
* Components can interact properly

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn on component dependencies' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset); Initialization order in Turn the System On sequence

#### b. test\_turn\_on\_then\_login (LT-012)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System, LoginManager | turn\_on() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Validate that after turn\_on, users can successfully log in.

**Input Specifications**

* System is turned on, Default admin account exists.

**Detailed Step**

1. Turn on system.
2. Attempt login (admin/1234).

**Expected Result**

* System turns on successfully
* Login with admin/1234 succeeds

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn on then login' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset); Login after system startup flow

#### c. test\_turn\_on\_security\_system\_integration (LT-013)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System, SecuritySystem | turn\_on() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Validate that SecuritySystem is properly configured with settings from ConfigurationManager after turn\_on.

**Input Specifications**

* System is in OFF state

**Detailed Step**

1. Call turn\_on().
2. Verify SecuritySystem config matches ConfigManager.

**Expected Result**

* SecuritySystem is initialized
* SecuritySystem uses configuration from ConfigurationManager

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn on security system integration' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset); Security system configuration in turn\_on sequence

#### d. test\_turn\_off\_component\_cleanup (LT-027)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

IT-SYS-OFF-001: Validate that all components are properly cleaned up and can communicate during shutdown sequence.

**Input Specifications**

* Verify all subsystem references are non-null, then run turn\_off().

**Detailed Step**

1. Verify components exist.
2. Call turn\_off().
3. Verify components cleaned up.

**Expected Result**

* Every component cleans up and releases resources during shutdown.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn off component cleanup' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

#### e. test\_turn\_off\_after\_login (LT-028)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

IT-SYS-OFF-002: Validate that system can be turned off after user login.

**Input Specifications**

* Set the system to ACTIVE by logging in, then invoke turn\_off().

**Detailed Step**

1. Login.
2. Call turn\_off().

**Expected Result**

* System gracefully signs out and advances to OFF directly from ACTIVE.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn off after login' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

#### f. test\_turn\_off\_security\_system\_integration (LT-029)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

IT-SYS-OFF-003: Validate that SecuritySystem alarm is deactivated during shutdown.

**Input Specifications**

* Confirm SecuritySystem is armed/initialized and call turn\_off().

**Detailed Step**

1. Arm system.
2. Call turn\_off().
3. Verify alarm deactivated.

**Expected Result**

* Alarm controller and siren deactivate as part of shutdown.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn off security system integration' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

#### g. test\_reset\_turn\_off\_turn\_on\_sequence (LT-041)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | reset() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

IT-SYS-RST-001: Validate that reset properly executes Turn Off followed by Turn On.

**Input Specifications**

* Ensure every subsystem is initialized, then run reset() to force an internal turn\_off/turn\_on.

**Detailed Step**

1. Initialize components.
2. Call reset().

**Expected Result**

* reset() reports success and state returns to READY after the composite sequence.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Reset turn off turn on sequence' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

#### h. test\_reset\_after\_login\_and\_settings\_change (LT-042)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | reset() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

IT-SYS-RST-002: Validate reset after user login and settings change.

**Input Specifications**

* Login and change configuration data before calling reset().

**Detailed Step**

1. Login & Change settings.
2. Call reset().
3. Verify settings & logout.

**Expected Result**

* Settings remain at the new values while the logged-in session is cleared.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Reset after login and settings change' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

#### i. test\_reset\_camera\_controller\_integration (LT-043)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | reset() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

IT-SYS-RST-003: Validate CameraController is properly reset.

**Input Specifications**

* Add several cameras via CameraController and then call reset().

**Detailed Step**

1. Add cameras.
2. Call reset().
3. Check camera controller state.

**Expected Result**

* Reset rebuilds the camera controller so it can accept new commands post-reset.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Reset camera controller integration' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

# **V. System Level Tests**

## **A. Log onto the System through Control Panel**

### **1. test\_control\_panel\_login\_success\_flow (ST-CP-Login-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanel, ControlPanelLoginPresenter, System, LoginManager, StorageManager |
| Function | Log onto the system through Control Panel |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate the complete login flow from Control Panel UI through all system layers to successful authentication.

**Input Specifications**

• ControlPanel receives user\_id "admin" and password "1234"

• StorageManager contains admin credentials ("admin", "1234")

**Detailed Step**

1. Seed StorageManager with admin user and password "1234".

2. Initialize System with turn\_on().

3. Initialize ControlPanel connected to System.

4. Simulate user entering "admin" as user\_id.

5. Simulate user entering "1234" as password.

6. Simulate pressing Login button.

7. ControlPanel creates ControlPanelLoginPresenter.

8. Presenter.attempt\_login() is called.

9. System.login\_with\_details() processes authentication.

10. LoginManager validates against StorageManager.

11. Success response flows back through layers.

12. ControlPanel displays success and navigates to MainMenu.

**Expected Result**

• System.login\_with\_details() returns success

• ControlPanel displays "Login successful"

• ControlPanel navigates to MainMenu screen

• User session is established

• Login attempts counter reset to 0

**Actual Result (Pass/Fail/Exception)**

• Pass: Login succeeds through all layers

• Pass: UI displays success message

• Pass: Navigation to MainMenu

**Comment (including references)**

End-to-end test of Control Panel login. Validates complete MVP pattern flow.

### **2. test\_control\_panel\_login\_lockout\_and\_unlock\_flow (ST-CP-Login-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanel, ControlPanelLoginPresenter, System, LoginManager |
| Function | Account lockout and time-based unlock via Control Panel |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate the complete lockout flow including time-based automatic unlock through Control Panel.

**Input Specifications**

• 5 failed login attempts with wrong password

• system\_lock\_time: 30 seconds

• Wait 31 seconds, then try with correct password

**Detailed Step**

1. Initialize full system stack.

2. Attempt login 5 times with wrong password "wrongpw".

3. Verify ControlPanel displays remaining attempts on each failure.

4. After 5th attempt, verify account locked message displayed.

5. Verify ControlPanel shows "Account locked. Try again in X seconds".

6. Wait for lock\_time to elapse (30 seconds).

7. Attempt login with correct password "1234".

8. Verify account auto-unlocks and login succeeds.

9. Verify ControlPanel navigates to MainMenu.

**Expected Result**

• Failed attempts show remaining count

• Account locked after 5 attempts

• Lock message shows remaining time

• Auto-unlock after 30 seconds

• Successful login after unlock

**Actual Result (Pass/Fail/Exception)**

• Pass: Lockout enforced

• Pass: Time-based unlock works

• Pass: Successful login after unlock

**Comment (including references)**

Tests complete lockout lifecycle including automatic unlock feature.

## **B. Configure System Settings through Control Panel**

### **1. test\_control\_panel\_settings\_save\_flow (ST-CP-Settings-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanel, ControlPanelSettingsPresenter, System, ConfigurationManager, StorageManager |
| Function | Configure system settings through Control Panel |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate the complete settings configuration flow from Control Panel UI through persistence.

**Input Specifications**

• User is logged in

• monitoring\_phone: "911"

• homeowner\_phone: "010-1234-5678"

• lock\_time: "45"

• alarm\_delay: "90"

**Detailed Step**

1. Initialize System and login as admin.

2. Navigate to Settings screen in ControlPanel.

3. Enter monitoring\_phone: "911".

4. Enter homeowner\_phone: "010-1234-5678".

5. Enter lock\_time: "45".

6. Enter alarm\_delay: "90".

7. Press Save button.

8. ControlPanel creates ControlPanelSettingsPresenter.

9. Presenter validates all inputs.

10. Presenter converts strings to appropriate types.

11. System.configure\_system\_setting() is called.

12. ConfigurationManager updates values.

13. StorageManager persists to database.

14. ControlPanel displays success message.

**Expected Result**

• Validation passes for all inputs

• Settings saved successfully

• ControlPanel displays "System settings saved successfully!"

• Values persist in database

• Values retrievable after system restart

**Actual Result (Pass/Fail/Exception)**

• Pass: All validations pass

• Pass: Settings persisted

• Pass: Success message displayed

**Comment (including references)**

End-to-end settings configuration test.

### **2. test\_control\_panel\_settings\_validation\_errors (ST-CP-Settings-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanel, ControlPanelSettingsPresenter, System |
| Function | Settings validation error handling |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate that invalid settings inputs are properly rejected with appropriate error messages.

**Input Specifications**

• lock\_time: "abc" (non-numeric)

• alarm\_delay: "-5" (negative)

**Detailed Step**

1. Initialize System and login as admin.

2. Navigate to Settings screen.

3. Enter invalid lock\_time: "abc".

4. Enter invalid alarm\_delay: "-5".

5. Press Save button.

6. Presenter validates inputs.

7. Validation fails with multiple errors.

8. ControlPanel displays error messages.

9. Settings are NOT saved.

**Expected Result**

• Validation fails

• Error message: "Invalid input. Please enter valid numbers for time fields."

• Settings not modified

• User remains on Settings screen

**Actual Result (Pass/Fail/Exception)**

• Pass: Validation rejects invalid inputs

• Pass: Error messages displayed

• Pass: No changes persisted

**Comment (including references)**

Tests presenter-level validation preventing invalid data.

## **C. Change Password through Control Panel**

### **1. test\_control\_panel\_password\_change\_success (ST-CP-PWD-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanel, ControlPanelChangePasswordPresenter, System, LoginManager, StorageManager |
| Function | Change master password through Control Panel |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate the complete password change flow from Control Panel through all phases.

**Input Specifications**

• User logged in as admin with password "1234"

• current\_password: "1234"

• new\_password: "newpass5678"

• confirm\_password: "newpass5678"

**Detailed Step**

1. Initialize System and login with admin/1234.

2. Navigate to Change Password screen.

3. Enter current password: "1234".

4. Enter new password: "newpass5678".

5. Enter confirm password: "newpass5678".

6. Press Change Password button.

7. Presenter validates passwords match and length.

8. System.change\_master\_password() executes 3 phases:

• Phase 1: Verify current password

• Phase 2: Validate new password rules

• Phase 3: Confirm and save

9. Password updated in database.

10. ControlPanel displays success message.

11. Logout and login with new password succeeds.

**Expected Result**

• All 3 phases complete successfully

• ControlPanel displays "Password changed successfully!"

• Old password no longer works

• New password authenticates correctly

**Actual Result (Pass/Fail/Exception)**

• Pass: Password change completes

• Pass: New password works

• Pass: Old password rejected

**Comment (including references)**

Tests complete 3-phase password change flow through UI.

### **2. test\_control\_panel\_password\_change\_mismatch (ST-CP-PWD-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanel, ControlPanelChangePasswordPresenter, System |
| Function | Password change with mismatched confirmation |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate that password change fails when new password and confirmation don't match.

**Input Specifications**

• current\_password: "1234"

• new\_password: "newpass1111"

• confirm\_password: "newpass2222" (mismatch)

**Detailed Step**

1. Initialize System and login.

2. Navigate to Change Password screen.

3. Enter current password: "1234".

4. Enter new password: "newpass1111".

5. Enter confirm password: "newpass2222".

6. Press Change Password button.

7. Presenter validates passwords do not match.

8. Error returned before calling System.

9. ControlPanel displays error message.

10. Password remains unchanged.

**Expected Result**

• Validation fails at presenter level

• Error message: "New passwords do not match!"

• System.change\_master\_password() is NOT called

• Original password still works

**Actual Result (Pass/Fail/Exception)**

• Pass: Mismatch detected

• Pass: Error displayed

• Pass: Password unchanged

**Comment (including references)**

Tests early validation at presenter layer preventing unnecessary system calls.

## **D. Log onto the System through Web Browser (2FA)**

### **1. test\_web\_login\_two\_factor\_success (ST-Web-Login-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebBrowser, WebLoginPresenter, System, LoginManager, StorageManager |
| Function | Two-factor authentication through Web Browser |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate the complete 2-factor authentication flow through the web interface.

**Input Specifications**

• username: "admin"

• first\_password: "password123"

• second\_password: "pin1234"

**Detailed Step**

1. Initialize System with turn\_on().

2. Open WebBrowser and navigate to login page.

3. Enter username: "admin".

4. Enter first password: "password123".

5. Click "Next" or "Continue".

6. WebLoginPresenter.validate\_first\_password() called.

7. First password validated successfully.

8. Web page transitions to second password input.

9. Enter second password/PIN: "pin1234".

10. Click "Login".

11. WebLoginPresenter.validate\_second\_password() called.

12. Second password validated successfully.

13. User redirected to dashboard.

14. Session established.

**Expected Result**

• First password validation succeeds

• UI transitions to second factor input

• Second password validation succeeds

• Redirect to "/dashboard"

• User session active

**Actual Result (Pass/Fail/Exception)**

• Pass: Both factors validated

• Pass: Redirect successful

• Pass: Session established

**Comment (including references)**

End-to-end test of 2FA web login flow.

### **2. test\_web\_login\_system\_off\_rejected (ST-Web-Login-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebBrowser, WebLoginPresenter, System |
| Function | Web login rejected when system is OFF |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate that web login is rejected when the system is in OFF state.

**Input Specifications**

• System is OFF (not initialized)

• Valid credentials provided

**Detailed Step**

1. Create System instance but do NOT call turn\_on().

2. Open WebBrowser and navigate to login page.

3. Enter valid username and password.

4. Click login.

5. WebLoginPresenter checks system state.

6. System state is OFF.

7. Login rejected with appropriate message.

8. User remains on login page.

**Expected Result**

• WebLoginOutcome.success equals False

• WebLoginOutcome.system\_off equals True

• Error message: "System is currently off"

• No redirect occurs

**Actual Result (Pass/Fail/Exception)**

• Pass: Login rejected

• Pass: System off message displayed

• Pass: User stays on login page

**Comment (including references)**

Precondition validation: System must be ON for login.

## **E. Configure System Settings through Web Browser**

### **1. test\_web\_settings\_full\_update\_flow (ST-Web-Settings-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebBrowser, WebSettingsPresenter, System, ConfigurationManager, StorageManager |
| Function | Full settings update through Web Browser |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate the complete settings configuration flow through the web interface.

**Input Specifications**

• User is logged in via web

• monitoring\_service\_phone: "111-222-3333"

• homeowner\_phone: "444-555-6666"

• system\_lock\_time: 30

• alarm\_delay\_time: 60

**Detailed Step**

1. Initialize System and complete web 2FA login.

2. Navigate to Settings page in web interface.

3. Enter monitoring\_service\_phone: "111-222-3333".

4. Enter homeowner\_phone: "444-555-6666".

5. Enter system\_lock\_time: 30.

6. Enter alarm\_delay\_time: 60.

7. Click "Save Settings".

8. WebSettingsPresenter.update\_settings() called.

9. Presenter validates all fields.

10. System.configure\_system\_setting() called.

11. Settings persisted to database.

12. Success message displayed.

13. Verify settings via get\_settings().

**Expected Result**

• All validations pass

• WebSettingsOutcome.success equals True

• Success message: "Settings updated successfully"

• Values persisted and retrievable

**Actual Result (Pass/Fail/Exception)**

• Pass: Settings saved

• Pass: Success displayed

• Pass: Values persist

**Comment (including references)**

End-to-end web settings configuration test.

### **2. test\_web\_settings\_multiple\_validation\_errors (ST-Web-Settings-002)**

|  |  |
| --- | --- |
| Field | Value |
| Class | WebBrowser, WebSettingsPresenter, System |
| Function | Multiple validation errors in web settings |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate that multiple validation errors are collected and displayed together.

**Input Specifications**

• monitoring\_service\_phone: "invalid"

• homeowner\_phone: "bad"

• system\_lock\_time: 0

• alarm\_delay\_time: -5

**Detailed Step**

1. Initialize System and login via web.

2. Navigate to Settings page.

3. Enter all invalid values.

4. Click "Save Settings".

5. WebSettingsPresenter validates all fields.

6. Multiple validation errors collected:

• Invalid monitoring phone format

• Invalid homeowner phone format

• Lock time must be > 0

• Alarm delay must be >= 0

7. All errors returned in response.

8. Web page displays all errors.

9. Settings NOT saved.

**Expected Result**

• WebSettingsOutcome.success equals False

• WebSettingsOutcome.errors contains 4+ errors

• All errors displayed to user

• No changes persisted

**Actual Result (Pass/Fail/Exception)**

• Pass: All errors collected

• Pass: Comprehensive error display

• Pass: No persistence

**Comment (including references)**

Tests comprehensive validation feedback for better user experience.

## **F. System Power On/Off through Control Panel**

### **1. test\_system\_turn\_on\_full\_initialization (ST-SYS-ON-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanel, System, StorageManager, ConfigurationManager, LoginManager, SecuritySystem, CameraController, Siren |
| Function | System turn on through Control Panel |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate complete system initialization when turning on through Control Panel.

**Input Specifications**

• System is in OFF state

• ControlPanel power button pressed

**Detailed Step**

1. System starts in OFF state.

2. User presses Power On button on ControlPanel.

3. ControlPanel calls System.turn\_on().

4. System initialization sequence:

• Initialize StorageManager (connect to database)

• Initialize ConfigurationManager (load settings)

• Initialize LoginManager (setup authentication)

• Initialize LogManager (setup logging)

• Initialize SystemController

• Initialize SecuritySystem (sensors)

• Initialize CameraController

• Initialize Siren

• Initialize AuthService

• Initialize SettingsService

5. System state changes to READY.

6. ControlPanel displays "System Ready".

7. All components operational.

**Expected Result**

• System.turn\_on() returns True

• System.system\_state equals READY

• All components initialized (not None)

• ControlPanel shows ready status

**Actual Result (Pass/Fail/Exception)**

• Pass: Initialization complete

• Pass: All components ready

• Pass: UI updated

**Comment (including references)**

Complete system initialization as per SDS page 50.

### **2. test\_system\_turn\_off\_full\_shutdown (ST-SYS-OFF-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanel, System, all managers and controllers |
| Function | System turn off through Control Panel |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate complete system shutdown sequence when turning off through Control Panel.

**Input Specifications**

• System is in READY state

• User is logged in

• Sensors and cameras are active

**Detailed Step**

1. System is running with user logged in.

2. Some sensors and cameras are active.

3. User presses Power Off button on ControlPanel.

4. ControlPanel calls System.turn\_off().

5. System shutdown sequence:

• Phase 1: Save configuration to database

• Phase 2: Deactivate all sensors

• Phase 3: Disable all cameras

• Phase 4: Deactivate siren

• Phase 5: Logout current user

• Phase 6: Log SYSTEM\_SHUTDOWN event

• Phase 7: Disconnect database

6. System state changes to OFF.

7. ControlPanel displays "System Off".

**Expected Result**

• System.turn\_off() returns True

• System.system\_state equals OFF

• All sensors deactivated

• All cameras disabled

• Siren off

• User logged out

• Configuration saved

**Actual Result (Pass/Fail/Exception)**

• Pass: Shutdown complete

• Pass: All components deactivated

• Pass: Data persisted

**Comment (including references)**

Complete shutdown sequence as per SDS page 51.

### **3. test\_system\_reset\_through\_control\_panel (ST-SYS-Reset-001)**

|  |  |
| --- | --- |
| Field | Value |
| Class | ControlPanel, ControlPanelResetPresenter, System, all components |
| Function | System reset through Control Panel |
| Author | Minseok Jo |
| Date | 2025-11-25 |

**Test Case Description**

Validate complete system reset flow through Control Panel.

**Input Specifications**

• System is running

• User is logged in

• Custom settings: lock\_time=45, alarm\_delay=90

**Detailed Step**

1. System running with custom settings configured.

2. User logged in as admin.

3. User selects Reset option in ControlPanel menu.

4. ControlPanel shows confirmation dialog.

5. User confirms reset.

6. ControlPanel creates ControlPanelResetPresenter.

7. Presenter calls System.reset().

8. System.reset() executes:

• turn\_off() (saves settings)

• turn\_on() (reloads settings)

9. System returns to READY state.

10. User is logged out (session cleared).

11. Settings are preserved.

12. ControlPanel shows login screen.

**Expected Result**

• Reset completes with phase 3

• System in READY state

• Settings preserved (lock\_time=45, alarm\_delay=90)

• User logged out

• ControlPanel on login screen

**Actual Result (Pass/Fail/Exception)**

• Pass: Reset successful

• Pass: Settings preserved

• Pass: User logged out

**Comment (including references)**

Reset combines turn\_off and turn\_on sequences with data preservation.

## G. Surveillance Level Tests

### 1. Display Specific Camera View (test\_st\_surv\_a\_001\_display\_camera\_view\_complete\_flow)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System, CameraController | login\_with\_details(), add\_camera(), validate\_camera\_password(), display\_single\_view() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Validate the complete end-to-end flow from web login to displaying specific camera view (UC a) at 1 frame per second.

Input Specifications

"Web login (admin/1234), Camera 1 setup, Password validation (""camera123"")."

**Detailed Step**

1. Web login.
2. Setup Camera 1 with password.
3. Validate password (Step 6-7).
4. Call display\_single\_view(1) (Step 9-10).

Expected Result

Login succeeds; Password validation returns 0 (Success); display\_single\_view returns a non-null view object or is handled gracefully (not crash).

Actual Result (Pass/Fail/Exception)

Pass

Comment (including references)

Complete flow test for viewing the feed after necessary authentication and setup.

### 2. Display View Disabled Camera (test\_st\_surv\_a\_002\_display\_view\_disabled\_camera)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System, CameraController | disable\_camera(), display\_single\_view() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Validate that the view button is disabled (view access is denied) when the camera is explicitly disabled.

Input Specifications

Camera 1 setup; Action: disable\_camera(1). Attempt to view.

**Detailed Step**

1. Login/Setup Camera 1.
2. Disable the camera.
3. Call display\_single\_view(1).
4. Assert view access is blocked.

Expected Result

display\_single\_view returns None (view access denied/blocked).

Actual Result (Pass/Fail/Exception)

Pass

Comment (including references)

Security policy verification: disabled cameras must not expose their feed.

### 3. Pan/Zoom Functionality (test\_st\_surv\_b\_001\_pan\_zoom\_camera\_view)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SafeHomeCamera (via Controller) | pan\_left(), pan\_right(), zoom\_in(), zoom\_out() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Validate the integrated pan and zoom functionality (UC b) for the specific camera view.

Input Specifications

"Login/Setup Camera 1. Action: Direct calls to camera.pan\_left(), camera.pan\_right(), camera.zoom\_in(), camera.zoom\_out()."

**Detailed Step**

1. Validate password.
2. Get Camera object.
3. Call pan\_left() and assert angle decreases (or hits min).
4. Call zoom\_in() and assert level increases.

Expected Result

"All pan and zoom operations must return True. Angles and levels must update correctly, adhering to boundary conditions (-180.0 to 180.0, 1 to 10)."

Actual Result (Pass/Fail/Exception)

Pass

Comment (including references)

Integrated test verifying low-level camera control operations.

### 5. Set Camera Password (test\_st\_surv\_c\_001\_set\_camera\_password\_complete\_flow)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System, CameraController | set\_camera\_password(), validate\_camera\_password() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Validate the complete end-to-end flow for setting a new camera password (UC c), ensuring the old password no longer works and the new one is valid."

Input Specifications

"Setup Camera 1 with ""oldpass123"". Action: Set new password ""newpass456""."

**Detailed Step**

1. Validate existing password (oldpass123).
2. Set new password (newpass456).
3. Verify the new password works (0).
4. Verify the old password fails (1).

Expected Result

New password validation returns 0. Old password validation returns 1. The password update succeeds.

Actual Result (Pass/Fail/Exception)

Pass

Comment (including references)

Complete flow test verifying password update functionality and security.

### 6. Set Password Mismatch Handling (test\_st\_surv\_c\_003\_set\_password\_mismatch\_handling)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | set\_camera\_password(), validate\_camera\_password() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Validate that if new passwords mismatch (simulated UI error), the system correctly rejects the change, and the original password remains valid ."

Input Specifications

"Setup Camera 1 with ""oldpass123"". Action: Attempt to set mismatched passwords (""newpass1"" and ""newpass2"")."

**Detailed Step**

1. Validate existing password (oldpass123).
2. Simulate password mismatch attempt.
3. Assert old password still works (0).
4. Assert new passwords fail validation.

Expected Result

Original password validation returns 0. New password validation returns non-0 (mismatch failure).

Actual Result (Pass/Fail/Exception)

Pass

Comment (including references)

System robustness check for rejecting inconsistent inputs during a critical setting process.

### 7. Delete Camera Password (test\_st\_surv\_d\_001\_delete\_camera\_password\_complete\_flow)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | validate\_camera\_password(), \_delete\_camera\_password() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Validate the complete end-to-end flow for deleting a camera password (UC d), ensuring the final state is 'Password Not Set'."

Input Specifications

"Setup Camera 1 with ""delete123"". Action: Call internal \_delete\_camera\_password(1)."

**Detailed Step**

1. Validate existing password (0).
2. Call \_delete\_camera\_password(1).
3. Manually clear state of SafeHomeCamera object (simulated consequence).
4. Assert final validation returns -2 (Password Not Set).

Expected Result

Password deletion succeeds. Final password validation returns -2.

Actual Result (Pass/Fail/Exception)

Pass

Comment (including references)

Complete flow test verifying password removal and resulting status consistency.

### 8. View Thumbnail Shots (test\_st\_surv\_e\_001\_view\_thumbnail\_shots\_complete\_flow)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | add\_camera(), display\_thumbnail\_view() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Validate the complete flow from web login to viewing the composite thumbnail shots (UC e) of multiple cameras.

Input Specifications

Web login. Action: Add three cameras. Call display\_thumbnail\_view().

**Detailed Step**

1. Login.
2. Add cameras 1, 2, and 3.
3. Call display\_thumbnail\_view().
4. Assert result is non-None (Image) or handled gracefully.
5. Verify camera count is 3.

Expected Result

"display\_thumbnail\_view executes successfully, returning a composite view object or handled gracefully, and all three cameras are included in the info list."

Actual Result (Pass/Fail/Exception)

Pass

Comment (including references)

System test for the composite view generation functionality.

### 9. Enable Camera (test\_st\_surv\_f\_001\_enable\_camera\_complete\_flow)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | disable\_camera(), enable\_camera() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Validate the complete flow for enabling a camera (UC f), ensuring the status transitions from disabled to enabled and the state is correctly reflected in the system."

Input Specifications

"Setup Camera 1. Action: Disable camera, then call enable\_camera(1)."

**Detailed Step**

1. Disable camera 1.
2. Call enable\_camera(1).
3. Assert the returned result is True.
4. Assert the camera's state is enabled=True in both the controller info and the camera object.

Expected Result

enable\_camera returns True. Final camera state is confirmed to be Enabled (True).

Actual Result (Pass/Fail/Exception)

Pass

Comment (including references)

System test for state change operation (Disabled to Enabled).

### 10. Disable Camera (test\_st\_surv\_g\_001\_disable\_camera\_complete\_flow)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| CameraController | disable\_camera() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

"Validate the complete flow for disabling a camera (UC g), ensuring the status transitions from enabled to disabled and the state is correctly reflected in the system."

Input Specifications

Setup Camera 1. Action: Call disable\_camera(1).

**Detailed Step**

1. Assert initial state is Enabled (True).
2. Call disable\_camera(1).
3. Assert the returned result is True.
4. Assert the camera's state is enabled=False in both the controller info and the camera object.

Expected Result

disable\_camera returns True. Final camera state is confirmed to be Disabled (False).

Actual Result (Pass/Fail/Exception)

Pass

Comment (including references)

System test for state change operation (Enabled to Disabled).

### 11. Password Lockout Exception (test\_st\_surv\_exception\_002\_password\_lockout\_after\_three\_attempts)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System, CameraController | validate\_camera\_password() | Jien Lee | 2025.11.23 | 1.0.1 |

Test Case Description

Validate that the system correctly tracks failed password attempts (3) and triggers a system lockout or appropriate failure response upon the third failure.

Input Specifications

"Setup Camera 1 with ""correct123"". Action: Call validate\_camera\_password(1, ""wrong"") three consecutive times."

**Detailed Step**

1. Call validate\_camera\_password with incorrect input 3 times.
2. Assert that attempts are tracked (attempts >= 3).
3. Verify that the logic path for system lockout is triggered/reached (implied by the test structure).

Expected Result

"Password validation fails three times, indicating the condition for system lockout is met."

Actual Result (Pass/Fail/Exception)

Pass

Comment (including references)

System robustness check for security measures against brute-force password guessing.

## H. Security System-Level Tests

| ID | Scenario | Steps | Expected Outputs | Evidence |
| --- | --- | --- | --- | --- |
| SLT-UC2-STAY | Tk Control Panel Stay flow | Launch python main.py, log in, arm STAY, open Front Door, disarm | Stay indicator lit, entry delay banner counts down, intrusion log entry cleared on disarm | docs/UC2\_manual\_logs/SLT-UC2-STAY.png |
| SLT-UC2-PANIC | Web dashboard panic | /dashboard panic button while system DISARMED | Toast “Panic alarm dispatched”, siren LED red, monitoring pane shows PANIC | Video clip 2025-11-25 (QA SharePoint) |
| SLT-UC2-ZONE | Dashboard zone CRUD | Create “Guest Wing”, assign/remove sensors via web UI | Zones table + Tk panel badges update immediately | Operator journal screenshots (2025-11-24) |
| SLT-UC2-LEGACY | Legacy /arm//status//disarm routes | Hit legacy endpoints via browser | HTML feedback matches SDS legacy Control Panel spec | Browser capture archived in QA log |

**Manual execution notes** - SLT-UC2-STAY - Conducted on Windows 11 with a USB keypad simulator; screenshots capture the Tk timer overlay, siren LED state, and the intrusion log entries before/after disarm to prove log cleanup. - SLT-UC2-PANIC - Recorded as a 30 second MP4 showing the Flask dashboard panic button, the siren widget flipping red immediately, and the monitoring list populating PANIC; audio output muted per lab policy. - SLT-UC2-ZONE - Creates, assigns, and deletes a “Guest Wing” zone using /api/zones indirectly through the dashboard; simultaneously records the Tk screen to show that badges refresh without application restart. - SLT-UC2-LEGACY - Uses curl + browser to access /arm, /status, /disarm; verifies HTML text aligns with SDS’s legacy copy and that event logs still populate even when using the deprecated endpoints.

**Per-test detail (SLT)**

### 1. Security Level Tests

#### a. STAY - Tk Control Panel Stay Flow

| Field | Value |
| --- | --- |
| Interface | Tk Control Panel (main.py) |
| Test Lead | Jonghwa An |
| Date | 2025-11-24 |
| Environment | Windows 11, USB keypad simulator |
| Evidence | docs/UC2\_manual\_logs/SLT-UC2-STAY.png |

**Test Case Description**  
Validates the full Stay-mode UX on the Tk control panel, including entry delay countdown, siren indicator, and log cleanup after disarm.

**Input Specifications**  
- Use seeded admin credentials.  
- Default entry delay 30 seconds.  
- Front Door sensor assigned to perimeter zone.

**Detailed Steps**  
1. Launch python main.py and log in via control panel keypad.  
2. Select STAY mode; verify STAY indicator lights.  
3. Open the Front Door (simulated sensor).  
4. Observe entry delay banner countdown.  
5. Disarm before the countdown expires.

**Expected Result**  
- STAY LED turns on; entry delay banner counts down from 30s.  
- Siren indicator remains amber until disarm.  
- Intrusion log entry recorded, then marked cleared after disarm.

**Actual Result (Pass/Fail/Exception)**  
- Pass: screenshot captures countdown overlay, LED colors, and before/after log entries.

**Comment (including references)**  
SDS VI.2a UI spec; evidence stored alongside coverage artefacts.

#### b. PANIC - Web Dashboard Panic

| Field | Value |
| --- | --- |
| Interface | Flask /dashboard |
| Test Lead | Jonghwa An |
| Date | 2025-11-25 |
| Environment | Windows 11, Chrome |
| Evidence | QA SharePoint video (link noted in docs) |

**Test Case Description**  
Demonstrates panic behavior via the web dashboard, showing immediate siren activation and monitoring update even when disarmed.

**Input Specifications**  
- System DISARMED.  
- Authenticated web session.

**Detailed Steps**  
1. Navigate to /dashboard and log in via 2FA.  
2. Click the Panic button.  
3. Observe toast notification, siren LED, and monitoring pane updates.  
4. Verify event log captures panic entry.

**Expected Result**  
- Toast “Panic alarm dispatched”.  
- Siren widget turns red.  
- Monitoring list shows PANIC.  
- Intrusion log entry flagged as panic.

**Actual Result (Pass/Fail/Exception)**  
- Pass: video capture shows UI transitions and monitoring entry appearing immediately.

**Comment (including references)**  
SRS UC2g/UC2h; SDS VI.2h panic UI flow.

#### c. ZONE - Dashboard Zone CRUD

| Field | Value |
| --- | --- |
| Interface | /configure dashboard |
| Test Lead | Jonghwa An |
| Date | 2025-11-24 |
| Environment | Windows 11, Chrome |
| Evidence | Operator journal screenshots (2025-11-24) |

**Test Case Description**  
Ensures zone creation, sensor assignment, and deletion propagate to both web UI and Tk badges without restart.

**Input Specifications**  
- Logged-in admin session.  
- Access to /api/zones through dashboard controls.  
- Tk panel running simultaneously.

**Detailed Steps**  
1. Navigate to Configure page.  
2. Create zone “Guest Wing”.  
3. Assign Front Door sensor to the new zone.  
4. Remove assignment and delete the zone.  
5. Observe Tk panel badges during each step.

**Expected Result**  
- Zones table updates in real time.  
- Tk badges display the new zone, then update after removal.  
- Database reflects CRUD operations.

**Actual Result (Pass/Fail/Exception)**  
- Pass: screenshots show synchronized updates across dashboard and Tk.

**Comment (including references)**  
SRS UC2e; SDS II.5 zone management.

#### d. LEGACY - Legacy /arm /status /disarm

| Field | Value |
| --- | --- |
| Interface | Legacy Flask templates (/arm, /status, /disarm) |
| Test Lead | Jonghwa An |
| Date | 2025-11-23 |
| Environment | Windows 11, Edge + curl |
| Evidence | Browser capture archived in QA log |

**Test Case Description**  
Confirms legacy endpoints remain functional and match SDS copy, ensuring backward compatibility.

**Input Specifications**  
- System running with legacy pages enabled.  
- Browser and curl client available.

**Detailed Steps**  
1. Hit /arm to arm the system; note HTML response.  
2. Invoke /status to confirm state text matches SDS spec.  
3. Trigger /disarm; verify message and log entry.  
4. Review intrusion log to ensure events recorded despite legacy path usage.

**Expected Result**  
- HTML content matches SDS legacy text.  
- Alarm state changes accordingly.  
- Logs capture each command invocation.

**Actual Result (Pass/Fail/Exception)**  
- Pass: screenshots show expected copy; event log entries recorded.

**Comment (including references)**  
SDS VI.2a legacy path requirements; ensures compatibility during migration.

### 2. System Lifecycle End-to-End

#### a. test\_complete\_turn\_on\_flow

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_on() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Validate the complete system turn on flow including all component initialization and state transitions.

**Input Specifications**

* Fresh system instance.

**Detailed Step**

1. Execute full turn\_on sequence.
2. Verify all subsystems.

**Expected Result**

* System transitions from OFF to READY
* All components functional
* Admin login works
* Settings are accessible

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Complete turn on flow' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset); Complete Turn the System On sequence diagram

#### b. test\_complete\_turn\_off\_flow

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

ST-SYS-OFF-001: Validate the complete system turn off flow including all component cleanup and state transitions.

**Input Specifications**

* Create a fresh System, turn it on, log in, interact, then call turn\_off().

**Detailed Step**

1. Use system (login, etc.).
2. Call turn\_off().
3. Verify full shutdown.

**Expected Result**

* Composite flow completes with every subsystem stopped and system state OFF.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Complete turn off flow' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

#### c. test\_turn\_off\_data\_persistence

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

ST-SYS-OFF-002: Validate that data is persisted correctly during shutdown and can be loaded on next turn\_on.

**Input Specifications**

* Turn on the system, mutate configuration data, turn off, then back on for verification.

**Detailed Step**

1. Modify data.
2. Turn off.
3. Turn on and verify data.

**Expected Result**

* Data persists across the shutdown/start cycle and settings remain accessible.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn off data persistence' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

#### d. test\_turn\_on\_turn\_off\_cycle

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | turn\_off() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

ST-SYS-OFF-003: Validate that system can be turned on and off multiple times without issues.

**Input Specifications**

* Iterate through several turn\_on()/turn\_off() cycles using one System instance.

**Detailed Step**

1. Cycle On/Off multiple times.
2. Verify stability.

**Expected Result**

* Each iteration succeeds, with READY reached after turn\_on and OFF after turn\_off.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Turn on turn off cycle' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

#### e. test\_complete\_reset\_flow

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | reset() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

ST-SYS-RST-001: Validate complete reset flow including all phases and state transitions.

**Input Specifications**

* Drive the full lifecycle: turn on, login, adjust settings, then invoke reset().

**Detailed Step**

1. Use system fully.
2. Call reset().
3. Verify restoration to READY.

**Expected Result**

* All three phases finish, settings persist, and the system is ready for another login.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Complete reset flow' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

#### f. test\_multiple\_reset\_cycles

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | reset() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

ST-SYS-RST-002: Validate system can be reset multiple times.

**Input Specifications**

* Turn on once, then execute three consecutive reset() calls within a loop.

**Detailed Step**

1. Loop reset() 3 times.
2. Check success each time.

**Expected Result**

* Every reset succeeds and leaves the system READY with managers intact.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Multiple reset cycles' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

#### g. test\_reset\_preserves\_all\_settings

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| System | reset() | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

ST-SYS-RST-003: Validate all types of settings are preserved.

**Input Specifications**

* Configure monitoring/homeowner phones plus timing values before invoking reset().

**Detailed Step**

1. Configure all setting types.
2. Call reset().
3. Verify values.

**Expected Result**

* After reset, all configured values remain unchanged.

Actual Result (Pass/Fail/Exception)

Pass: Observed 'Reset preserves all settings' behavior matched lifecycle expectations.

Comment (including references)

SDS lifecycle (startup/shutdown/reset)

### 3. Zone Assignments

#### A. SecuritySystem

##### a. Explicit Zone Tracking

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Method** | **Author** | **Date** | **Version** |
| SecuritySystem | N/A (Zone Logic) | Jonghwa An | 2025-11-25 | 2.0.0 |

Test Case Description

Validates 'Security system tracks explicit zone assignments' behavior.

Input Specifications

See tests/test\_zone\_assignments.py

Detailed Step

Run tests/test\_zone\_assignments.py:52

Expected Result

Security system correctly tracks assignments.

Actual Result (Pass/Fail/Exception)

Pass: observed state/response matches SDS/SRS design.

Comment (including references)

Trace: SRS UC2e / SDS VI.2f

# VI. Who did what

|  |  |
| --- | --- |
| **Name** | **Contribution** |
| **Jien Lee** | The scope of work focused on core feature development, quality assurance, and essential project documentation for the SafeHome system.  Core Feature Development: Development of the surveillance system was completed with the implementation of the SafeHomeCamera and Camera classes, establishing the camera object definitions and primary control logic. The core security response mechanism was also integrated, which included the implementation of the Alarm class and the establishment of the required sensor-to-alarm connection logic for automated intrusion handling. Additionally, the User Interface (UI) underwent functional and aesthetic refinement to improve the overall user experience.  Documentation and Quality Assurance (QA): Contributions to the Implementation and Testing document included the authoring of Unit Tests specifically for the surveillance components. The system's architecture was documented through the creation of a Class Diagram that reflected the implemented classes. Furthermore, the necessary Integration Test and System Level Test cases were developed to validate the end-to-end functionality of the surveillance features. The final deliverable involved drafting the "Who Did What" documentation to formally record individual contributions. Also wrote the Overview of Implementation and Testing |
| **Jonghwa An** | The scope of work was comprehensive, covering core security feature implementation, quality assurance for the security subsystem, and overall project documentation and file management.  Core Feature Development: The primary implementation task was the development of the security features of the SafeHome system. This involved defining and implementing all necessary classes and logic to handle system states (e.g., Armed, Disarmed) and user authentication.  Code Quality and Coverage: A significant effort was dedicated to improving the project's overall code quality. This included analyzing and enhancing the entire file coverage, specifically addressing and documenting the Coverage Branch details to ensure robust and fully tested code paths across the project.  Documentation and Quality Assurance (QA): Extensive testing and documentation were completed for the security subsystem. This work included authoring the Unit Tests for all security-related components to ensure individual module integrity. The system's design was documented by creating the necessary Class Diagram that includes the security features. Furthermore, the required Integration Test and System Level Test cases were developed to validate the security subsystem's end-to-end functionality. Finally, all independently authored documentation files were consolidated through the document integration effort, ensuring a unified and consistent final project document structure. |
| **Minseok Jo** | The scope of work encompassed the implementation of common system functionalities, the foundational setup of the user interface, and comprehensive documentation for the common subsystem.  Core Feature Development and Foundation: Implementation of the common system functionalities was completed, providing shared resources and utilities for other subsystems. This involved laying the initial foundation for both the Control Panel and the Web User Interface (UI), establishing the basic structural framework and navigation for both access points.  Project Management and Documentation: A significant project management responsibility was fulfilled through the consistent creation and maintenance of the Meeting Log, documenting decisions and progress across team meetings.  Documentation and Quality Assurance (QA): Extensive testing and documentation were performed for the common subsystem. This work included authoring the Unit Tests for all common components to ensure individual module integrity. The system's design was documented by creating the necessary Class Diagram that includes the common features. Furthermore, the required Integration Test and System Level Test cases were developed to validate the common subsystem's functionality and its interactions with other modules. |

# **VII. Meeting Log**

## **Python UI and Integration Planning**

|  |  |
| --- | --- |
| **Category** | **Details** |
| Date & Time | November 20, 2025 (Thursday), 5:30 PM – 9:30 AM (4h) |
| Attendees | Jien Lee, Jonghwa An, Minseok Jo |
| Purpose | Review code and UI status, discuss role assignments and integration testing strategy |
| Discussion Summary | - Reviewed current Python-based UI implementations; noted main screens incomplete, control panel functional  - Decided UI button linking is primary front-end task, with minimal additional front-end design  - Role assignment via ladder draw: Minseok for common + integration testing, Jien for security, Jonghwa for surveillance  - Discussed need for integration testing to combine feature modules, agreeing team lead to handle integration effort  - Mentioned associated test documentation to be prepared as part of coursework following class content  - Clarified testing documentation templates available in class resources (KLMS)  - Considered feasibility of using web hosting vs. local web server; consensus on local hosting for project  - Debated choice of frameworks and infrastructure setup for fastest lightweight deployment (FastAPI favored over Flask)  - Assigned responsibilities for documentation processing and project progress tracking  - Planned further meetings and project check-ins to monitor progress and resolve issues |
| Decisions Made | - Each member proceeds on assigned functional modules and documentation  - Team lead to integrate modules and handle GUI linking  - Use local server deployment for web UI  - Follow class-provided test documentation formats for integration and system testing  - Regular progress reviews planned, including short meeting Saturday afternoon and code review on Sunday night |
| Next Steps | Complete individual module implementation, prepare associated test documentation, integrate modules, and continue UI enhancements; monitor status at upcoming meetings |

## **Progress Update & Integration Planning**

|  |  |
| --- | --- |
| **Category** | **Details** |
| Date & Time | November 22, 2025, 3:00 PM – 3:15 PM (15 minutes) |
| Attendees | Jien Lee, Jonghwa An, Minseok Jo |
| Meeting Purpose | Share current development progress, review API implementation, synchronize branch merges, and plan next steps for feature completion |
| Discussion Summary | - Jien completed thorough document review and noted significant API implementations ready to deploy.  - Plans to complete development within the day and assist others as needed.  - Jonghwa confirmed details available in the security folder; advised usage of related files.  - Minseok finished login and control panel features; UI wireframe roughly completed.  - Remaining tasks include system on/off/reset and password changes pending manager-side implementations.  - Code branches merged and pushed to main repository.  - DB file added to gitignore for version control.  - Future coordination to continue post-merge for integrated development progress. |
| Key Decisions | - Proceed with completing individual development parts and offer cross-support.  - Maintain up-to-date branch merges to ensure seamless collaboration.  - Utilize existing security module resources effectively.  - Track development status regularly with brief sync meetings. |

## **Directory Organization, Development Progress, and Testing Status**

|  |  |
| --- | --- |
| **Category** | **Details** |
| Date & Time | November 24, 2025 (Monday), 12:00 AM – 6:00 AM (6h) |
| Attendees | Jien Lee, Jonghwa An, Minseok Jo |
| Meeting Purpose | Discuss directory restructuring, individual weekend development progress, testing coverage, bug fixes, and task coordination |
| Discussion Summary | - Directory reorganized due to clutter; each member shared weekend development achievements.   - Completed class diagrams, document review, and UI connections tested.  - Security folder contains detailed implementations and extensive tests.  - Some code cleanups planned (removal of redundant code).  - Coverage tests passed with meaningful coverage percentages.   - Multiple merges and pushes planned with sequential task handoffs.   - Testing documents created and coverage documentation to be uploaded.  - Bug fixes include camera info retrieval and image load path corrections. - Decided to push sequentially with proper checks to avoid conflicts.  - Discussed tool usage preferences and code review schedules.  - Planned upcoming meetings and progress updates, including a Saturday afternoon check-in.  - General discussions on project documents, AI-assisted testing code generation, and resource management. |
| Key Decisions | - Maintain restructured directory for clarity.  - Prioritize completing and merging individual development tasks.  - Upload test coverage documentation.  - Perform detailed bug fix verification before pushing.  - Keep regular progress meetings and code reviews scheduled.  - Assign team roles flexibly with collaborative problem solving. |
| Next Steps | Finalize pending fixes, continue documentation, integrate code with proper testing, and participate in upcoming scheduled meetings. |

## **Test Coverage, Documentation, and SRS Alignment**

|  |  |
| --- | --- |
| **Category** | **Details** |
| Date & Time | November 26, 2025 (Wednesday), 16:00 ~ 18:00 (2h) |
| Attendees | Jien Lee, Jonghwa An, Minseok Jo |
| Purpose | Review missing web interface tests, raise coverage to required threshold, align test/documentation with SRS/SDS, and coordinate remaining work. |

|  |  |
| --- | --- |
| **Category** | **Details** |
| Discussion Summary | - Identified missing test code for the web interface and added new tests; initial push was blocked by low coverage, so more tests were written to meet the coverage gate.  - Current tests number around 400 cases, with many UI-level and function-level unit tests, especially for configuration and control functions.  - Some modules inherently show lower coverage (e.g., pure state/enum holders), but overall coverage is considered reasonable (around or above ~70% in most core areas).  - Unit test documentation is large (dozens of pages for some parts) due to detailed per-test-case descriptions.  - Integration and system test documents are being written based on implemented feature groups; length depends on the number of combined scenarios each member implemented.  - Clarified that much of the detailed test code and documentation was generated with AI assistance but manually reviewed and adjusted.  - Reconfirmed Git strategy: work in personal branches, run full test suites locally, then merge into main only after verification to avoid broken pushes under coverage checks.  - Discussed inconsistencies between SRS/SDS and the given GUI prototype (e.g., safety zone handling, control panel capabilities) and noted that the team is prioritizing the provided prototype while recording assumptions in documentation. |
| Decisions Made | - Keep the current directory and test structure and continue to rely on coverage reports as a quality bar.  - Complete unit, integration, and system test documents this week, using existing templates and AI only as a helper, not a source of record. - When SRS/SDS and GUI conflict, follow the shipped prototype and clearly state assumptions and rationale in the documents.  - Maintain the coverage gate on CI to prevent low-tested code from being merged into main. |

Web UI/SDS Alignment & Final Tasks

|  |  |
| --- | --- |
| **Category** | **Details** |
| Date & Time | December 1, 2025 (Mon), 12AM - 9AM (9h) |
| Attendees | Jien Lee, Jonghwa An, Minseok Jo |
| Purpose | - Verify web pages vs SDS requirements  - Clarify panic alarm/monitoring behavior from SRS/SDS  - Plan code fixes, doc merging, final submission |

|  |  |
| --- | --- |
| **Progress Summary** |  |
| Docs/Templates | Created MD files/checklists; unified format to match Minseok's PDF template |
| Impl/UI/Manual | Finished Implementation & Testing doc; cleaned cluttered web home screen post-login; drafted web user manual |

|  |  |
| --- | --- |
| **SRS/SDS Issues** |  |
| Panic Alarm | Sends notifications (phone/monitoring sys), logs events, sirens; some ops blocked (e.g. safety zone edits per use-case exceptions). Not full system freeze |
| Monitoring/Delay | Intrusion detect → delay → if uncleared, escalate (log/notify monitoring sys, retrigger alarm). "Monitoring sys" = internal logging (no real external svc needed) |
| Sensor Trigger | UI trigger for detection sim lost in merges; restore in app/web for testing |

|  |  |
| --- | --- |
| **Web UI vs SDS** |  |
| Page Check | Most SDS pages exist (safety zone config, camera views, surveillance, sys config).Missing: Web "View Log" page→ implement |
| Phone Validation | SDS lists it, but prof said skip → omit & document as intentional |