Project Title: System Verification and Validation Plan for 4TB6 - Mechatronics Capstone Smart Bike Lock

Abi Nevo
Elsa Bassi
Steffi Ralph
Abdul Iqbal
Stephen De Jong
Anthony Shenouda

November 2, 2022

1 Revision History

Date	Version	Name	Notes
20-10-22	1.0	Steffi	Section 2, 3 & 4
01-11-22	1.1	Stephen, Elsa, Abi, & Anthony	Section 5
02-11-22	1.2	Abdul	Section 4

Contents

List of Tables

2 Symbols, Abbreviations and Acronyms

Refer to Section 1 of the SRS documentation for a full reference section on units, symbols and abbreviations/acronyms.

This document will outline the verification and validation plans we have created for the purpose of proving that the SmartLock device is a successful product. The tests mentioned in this document will validate our product against the various requirements which we have outlined in the SRS document. At the completion of the plan in this document, we will have the knowledge required to form an iterative design process which improves into a successful device at its completion.

3 General Information

3.1 Summary

The smart lock can be broken down into two main components, the software (ie. the smartphone application) and the hardware (ie. the physical lock).

The smartphone application, SmartLock, will be a basic UI that our users can interact with to send a signal to unlock/lock the locking mechanism, check in on the battery of the device, and remember where they left their bike when they locked it.

The physical locking device will be used to secure the wheels to the bike frame, which can also be connected to an external mount.

These components will be tested through various different methods, to determine if they meet all the requirements outlined in the SRS.

3.2 Objectives

The objective that our project hopes to accomplish is to create a simple-to-use smartphone app that allows users to lock the most important features of their bike with confidence and to allow them to find their bike with ease.

3.3 Relevant Documentation

For more information on the project breakdown, planning or delivery refer to the following documentation: SRS, HA, MG, and MIS.

4 Plan

In this section, various aspects of the verification plan will be illustrated explaining the process from concept to implementation. Beginning with an outline of the Validation team and their roles in guiding the process, this section will cover the verification of our SRS documentation as well as how we plan to design, implement and test our problem and eventually prototype our solution. It also covers any automation tools used as well as a software validation plan.

4.1 Verification and Validation Team

Table 1: Verification and Validation Team Table

VnV Team Members	Role
Group 5 Members	Responsible for quality control on their topic of expertise, keeping the other group members aware of any changes or progress and reflecting that knowledge in the documentation
Gr 5: Abi Nevo	Wireless Communication Expert
Gr 5: Elsa Bassi	Microcontroller Expert
Gr 5: Steffi Ralph	Documentation Expert/ Faux Marker
Gr 5: Abdul Iqbal	Lock Mechanism Expert
Gr 5: Stephen De Jong	Lock Frame Design Expert
Gr 5: Anthony Shenouda	Software Expert
Dr. Sirouspour	From our supervisor, Dr. Sirouspour, we are looking for technical feedback on components that we might not have considered or fully grasped as well as feedback on any design, materials or documentation considerations
Classmates	Peer review from the perspective of someone who knows what is required of the documentation
Nicholas Annable	Provide specific feedback and grades on the work we have completed so that we can take the appropriate action moving further into the project
Dr. Smith	Provide information and guidance on the project goals/deliverables and documentation requirements

4.2 SRS Verification Plan

Our plan for verification of the SRS includes the following steps:

1. Keep the Document Alive

   By keeping the document alive we will be able to continually update any requirements/constraints/objectives that adapt as our project comes to life. This will ensure that all the documentation is accurate.

2. Peer Review

   We will use the peer review to verify our work and to look for inconsistencies or aspects that we didn't consider.

3. TA Review

   The TA review will be used to validate what we have done and what needs to be reworked as the project progresses.

4. SRS Checklist

   With every update of the SRS document and progression of the SmartLock we will continue to reference the SRS Checklist to ensure that we continue to meet all the requirements and produce complete documentation.

4.3 Design Verification Plan

Our plan for verification of the Design Plan includes the following steps:

1. Review with Supervisor (Dr. Sirouspour)

   As a novice design team, we will use the meetings with Dr. Sirouspour to review, primarily the hardware components of, our design and understand if we are using the right technical pieces and if our approach can lead to success.

2. Proof of Concept Demo

   The proof of concept demo will be a crucial part of our validation plan. With the intent to be able to demonstrate both a portion of our physical component and a preliminary software UI, the presentation and feedback will provide us with an opportunity to validate our objectives and verify the next steps and problem areas.

3. Testing

   Testing, which will be discussed in detail below, will be used to understand if our requirements can be met given the scope of this project and help us to determine what needs to change for our final product.

4. MIS & MG Checklists

   The MIS Checklist and the MG Checklist will be used to review the tests that are created and check that they follow the guidelines so that we can properly test the work product. Tests will be reviewed and updated as the project progresses.

4.4 Implementation Verification Plan

Our plan to verify the implementation is outlined below:

1. Functional Requirements Verification

   The verification of our functional requirements will be done by performing the system tests discussed in Tests for Functional Requirements. The success rate of these system tests will help the team to confirm and validate these requirements and therefore will provide the team with good metrics on the core functionality of the product.

2. Non-Functional Requirements Verification

   The verification of our non-functional requirements will be done by performing the system tests discussed in Tests for Nonfunctional Requirements. The success rate of these system tests will help the team to confirm and validate these requirements and therefore will provide the team with good metrics on the overall quality and class of the product. This will help us in verifying other aspects of the product including the App features, durability and ease of use.

3. Code Implementation Verification

   In order to verify the implementation of our code-base, we will make use of both code walk-throughs and inspection by fellow peers as well as by performing unit tests on our system. Having a peer review done would not only illuminate any shortcomings in our implementation but also help us in identifying more efficient techniques within our code-base. By performing unit tests, which are discussed in Unit Test Description, we can verify different modules within our system to see if each is performing as expected and is robust. This will help us maintain best practices while implementing our code-base.

4.5 Automated Testing and Verification Tools

The tools used for automated testing and verifying coding standards are outlined below:

1. SwiftLint

   SwiftLint will be used in order to enforce common Swift style and naming conventions.

2. CodeFactor

   CodeFactor will be used for automated code analysis on our repo on Github.

3. Github

   Github will be used to host our codebase including all staging branches and the production branch. All changes will need to be approved and reviewed before merging to production.

4. Git

   Git will be used for version control, resolving merge conflicts and interfacing with different branches.

4.6 Software Validation Plan

[If there is any external data that can be used for validation, you should point to it here. If there are no plans for validation, you should state that here. —SS]

5 System Test Description

5.1 Tests for Functional Requirements

The tests in this section will be used to confirm and validate our functional requirements from the SRS document. Completing these tests will prove the functionality of our product

5.1.1 Area of Testing: User Input Related

1. test-id1: FR1: LockEngage input must engage the lock on the bike.

   Control: Automatic

   Initial State: Bike lock is disengaged

   Input/ Condition: Signal from App to engage the lock

   Output/ Results: Pass/fail if lock is successfully engaged

   How test will be performed: The bike lock will be disengaged. The user will use the App to engage the lock. If successful, lock becomes engaged.

2. test-id2: FR2: LockDisengage input must disengage the lock on the bike.

   Control: Automatic

   Initial State: Bike lock is engaged

Input/ Condition: Signal from App to disengage the lock

Output/ Results: Pass/fail if lock is successfully disengaged

How test will be performed: The bike lock will be engaged. The user will use the App to disengage the lock. If successful, lock becomes disengaged.

3. test-id3: FR3: When lock is engaged, change lock status to engaged.

   Control: Automatic

   Initial State: Bike lock is engaged

   Input/ Condition: Signal from microcontroller to App stating lock is engaged

   Output/ Results: Pass/fail if App correctly displays lock is engaged

   How test will be performed: The bike lock will be engaged.

4. test-id4: FR4: When lock is disengaged, change lock status to disengaged.

   Control: Automatic

   Initial State: Bike lock is disengaged

   Input/ Condition: Signal from microcontroller to App stating lock is disengaged

   Output/ Results: Pass/fail if App correctly displays lock is disengaged

   How test will be performed: The bike lock will be disengaged.

5. test-id5: FR5: Moving locking bar closed must move OpenClosedStatus to Closed.

   Control: Automatic

   Initial State: Locking bar is open

   Input/ Condition: Locking bar is moved closed

   Output/ Results: Pass/fail if App correctly displays OpenClosedStatus to Closed

How test will be performed: The locking bar will be moved closed and the App will be checked if the OpenClosedstatus to Closed.

6. test-id6: FR6: Moving locking bar open must move OpenClosedStatus to Open.

   Control: Automatic

   Initial State: Locking bar is closed

   Input/ Condition: Locking bar is pushed open

   Output/ Results: Pass/fail if App correctly displays OpenClosedStatus to Open

   How test will be performed: The locking bar will be pushed open and the App will be checked if the OpenClosedStatus to Open.

7. test-id7: FR7: Location (coordinates) of user's phone must be able to be saved in the smartphone application as UserPosition.

   Control: Automatic

   Initial State: No location

   Input/ Condition: User engages the Bike lock using the App

   Output/ Results: Pass/fail if App places a geotag within 10 metres of the Users current location

   How test will be performed: The Bike lock will be engaged on the App. The App will be monitored to ensure the geotag is stored and correctly placed on the App.

8. test-id8: FR8: Effective Bike Lock: The lock is sturdy and cannot be manually opened by the average human once engaged.

   Control: Manual

   Initial State: Lock is engaged, and bike is locked.

   Input/Condition: A prying, pulling, kicking, etc. force between 200-400N.

Output/Results: A pass/fail as well as a score from 1-4 for the following cases; a fail if the lock disengages and breaks, a fail if the lock disengages, a pass if the lock stays engaged but breaks, and a pass if the lock system can stay engaged with out breaking.

How test will be performed: A test group of 2-3+ adults, completing 50+ trials, each giving the required input forces.

9. test-id9: FR9: Lock must only be engaged/disengaged by the intended user(s).

   Control: Manual

   Initial State: Lock is engaged, and bike is locked.

   Input/Condition: Iphone Bluetooth signal.

   Output/Results: A fail if they are able to connect to the smart bike lock and disengage it and a pass if they can't connect and disengage the lock.

   How test will be performed: A test group will attempt to connect via Bluetooth from their smartphones.

5.1.2 Area of Testing: Bike Input Related

1. test-id10: FR10: The lock can be mounted to the bike's frame.

   Control: Manual

   Initial State: Lock is in disengaged state and not mounted to a bike.

   Input/Condition: Users force to mount the SmartLock.

   Output/Results: A pass/fail for each bike if the lock is/isn't able to mount in to the bike as intended. A score from 0-the number of bikes tested will also be given.

   How test will be performed: Users will attempt to mount the SmartLock onto 1-3 bikes each from 3-5 categories of bikes including (adult standard/hybrid, road, mountain, kids standard/hybrid).

5.1.3 Area of Testing: Output Related

1. test-id11: FR11: Battery percentage must be shown on the phone app.
   Control: Manual

   Initial State: Phone on with App downloaded and open

   Input/Condition: observation of app

   Output/Results: user must be able to view the battery percentage of the lock

   How the test will be performed: apply input condition, observe expected output

2. test-id12: FR12: Location (coordinates) of bike must be shown on the app as BikePosition.
   Control: Manual

   Initial State: Phone on with App downloaded and open

   Input/Condition: observation of app

   Output/Results: user must be able to view the saved coordinates

   How the test will be performed: apply input condition, observe expected output

3. test-id13: FR13: Battery must output enough power to engage the lock.
   Control: Manual

   Initial State: Locking mechanism disengaged, electro magnet has ability to get power from battery (functional circuit)

   Input/Condition: supply power to electromagnet

   Output/Results: lock mechanism is engaged

   How the test will be performed: apply input condition, observe expected output

5.2 Tests for Nonfunctional Requirements

This subset of tests will be used to validate the nonfunctional requirements of our product. Completing these tests will prove various aspects of our products needs. these aspects include smart phone app features, physical design attributes, accuracy, and the usability of our product.

5.2.1 Area of Testing: Smart Phone

1. test-id14: NFR1: Can be used by people who speak any native language.

   Control: Manual

   Initial State: Lock is engaged and phone is on with App downloaded and open.

   Input/Condition: Non-English-speaking test user has no prior knowledge of how to use the System.

   Output/Results: The lock is successfully engaged and the bike is securely locked by test user.

   How test will be performed: User attempts to disengage the mechanism through the App, attach bike to external frame and re-engage mechanism.

2. test-id15: NFR2: Can reasonably be used without requiring an instruction manual.

   Control: Manual

   Initial State: Lock is engaged and phone is on with App downloaded and open.

   Input/Condition: A group of test users with no prior knowledge of how to use the System.

   Output/Results: The lock is successfully engaged, and the bike is securely locked by all test users.

   How test will be performed: Users are told to disengage the mechanism through the App, attach the bike to an external frame and re-engage the mechanism without any instruction. The time required for full usage per test is measured using a stopwatch and must below ten minutes. Learning period is subsequently calculated and plotted.

3. test-id16: NFR3: App storage under 50 megabytes. A small mobile app should not take significant space on the user's phone.

Control: Static

Initial State: App development IDE is open and up to date.

Output/Results: App storage is less than 50 megabytes.

How test will be performed: App storage displayed on the IDE is kept below the upper threshold of 50 megabytes.

5.2.2 Area of Testing: Physical Lock

1. test-id17: NFR4: The design must be visually appealing.
   Control: Manual

   Initial State: Locking mechanism assembled

   Input/Condition: Survey users on their opinions of the visual appeal of the device; see Usability Survey Questions.

   Output/Results: Visual appeal is rated 7 or higher (on a scale of 1-10)

   How the test will be performed: Apply input condition, survey 50 users and observe expected output.

2. test-id18: NFR5: The lock must not impede normal bike functions.
   Control: Manual

   Initial State: Locking mechanism assembled and mounted on bike frame

   Input/Condition: Normal bike operation (ride forward, turn right, turn left, ride downhill, ride uphill)

   Output/Results: Device does not impede bicycle functioning

   How the test will be performed: apply input condition, observe expected output

3. test-id19: NFR6: The lock must be waterproofed to withstand normal rainfall. And NFR7: The lock must be waterproofed to withstand normal splashing while riding.

   Stage 1: Empty Housing (no electronics inside)

   Control: Manual

   Initial State: Husing assembled.

Input/Condition: Simulate rain: Hold housing under shower for 5 minutes, slowly rotating.

Output/Results: Receives a pass if the inside of housing is completely dry.

How the test will be performed: apply input condition, observe expected output

Stage 2: Full Mechanism

Control: Manual

Initial State: Locking mechanism, electronics, and housing assembled, microcontroller turned on.

Input/Condition: Simulate rain: hold housing under shower for 5 minutes, slowly rotating.

Output/Results: Receives a pass if the device remains functional and operational.

How the test will be performed: apply input condition, observe expected output

4. test-id20: NFR8: Accuracy of bike lock status must be above 95%.
   Control: Manual

   Initial State: Locking mechanism, electronics, and housing assembled operational and within range.

   Input/Condition: Engage/disengage lock 100 times.

   Output/Results: Ensure engage/disengage status matches current physical state.

   How the test will be performed: apply input condition, observe expected output

5.2.3 Area of Testing: Usability

1. test-id21: NFR11: iPhone app locking must be quicker to use than a typical keyed/combo bike lock.

   Control: Manual

Initial State: The Smart lock is in the transport state with the test user standing beside the bike at a bike rack.

Input/Condition: User engaging and disengaging the SmartLock.

Output/Results: Scores from 1-10 will be given for weighted time from fastest to slowest.

How test will be performed: Users will be timed on locking and unlocking. As a group of 3 people will, remove lock from the transport state, and engage the lock as intended. To lock they will disengage the lock as intended and convert to transport state. This will be done with 3-5 types of locks including, SmartLock, keyed, and combination.

2. test-id22: NFR12: Opening and closing lock must require similar force to a typical keyed/combo.

   Control: Manual.

   Initial State: The Smart Lock will be in transport state with the test user standing beside the bike at a bike rack.

   Input/Condition: Users Engaging and Disengaging the SmartLock.

   Output/Results: Scores from 1-5 will be given from most to least amount of relative force required.

   How test will be performed: Users will measure force while locking and unlocking. As a group of 3 people will, remove lock from the transport state, and engage the lock as intended. To lock they will disengage the lock as intended and convert to transport state. This will be done with 3-5 types of locks including, SmartLock, keyed, and combination.

3. test-id23: NFR13: Battery must last for greater than 1 month and/or 60 rides before needing to be replaced or charged.

   Control: Manual

   Initial State: The Smart Lock is fully charged.

   Input/Condition: Users Engaging, Locating, and Disengaging the SmartLock.

   Output/Results: The amount of time and quantity of lock/unlocks of the Smart Lock. A pass if it meets the required number.

How test will be performed: A group of users will take turns to bike to a new bike lock, lock the bike, mark the bikes location in the app, unlock, and then repeat. This will be done until the battery dies.

4. test-id24: NFR14: Batteries must be accessible to replace or chargeable.

   Control: Manual.

   Initial State: The Smart Lock is in the Transport state.

   Input/Condition: Users will charge/replace batteries as intended.

   Output/Results: Pass if the batteries can be charged/replaced as intended.

   How test will be performed: Users will charge/replace batteries as intended.

5. test-id25: NFR15: The lock must be easily mounted on the bike frame. It does not require special tools, (i.e., those not found in a typical toolbox, such as power tools), to be installed and does not take more than twenty minutes to install.

   Control: Manual.

   Initial State: The System is ready to be installed.

   Input/Condition: A group of test users with an average understanding of how to use typical tools. They have access to those typical tools.

   Output/Results: The System is successfully mounted on the bike frame without special tools for each test user.

   How test will be performed: Each user attempts to install the System, following the procedure outlined in the instructions. This procedure will be developed following the completion of the first stage of prototyping.

6. test-id26: NFR16: The lock can be used for many different models of mountain, city, and road bikes.

   Control: Manual

Initial State: The System is ready to be installed.

Input/Condition: A group of test users with an average understanding of how to use typical tools. They have access to those typical tools. Three different models of bike are tested; one for each type (mountain, city and road).

Output/Results: The System can be mounted on all three bike models successfully by each test user.

How the test will be performed: Three users attempt to install the System, following the procedure outlined in the instructions.

5.2.4 Area of Testing: Accuracy

1. test-id20: NFR10: Battery percentage must be calculated accurately within 10%.

   Control: Manual

   Initial State: Full device assembled, battery percentage calculated is 1%.

   Input/Condition: Engage/disengage lock until battery dies.

   Output/Results: Number of lock engages possible with 1% battery matches our specification for number of lock engages possible with 100% battery (multiply the measured # of lock engages possible with 1% battery by 100), within 10% accuracy.

   How the test will be performed: Apply input condition and observe expected output

5.3 Traceability Between Test Cases and Requirements

Table 2: Traceability Table

Test Case	Functional Requirement(s)	Non-Functional Requirement(s)
test-id1	FR1	
test-id2	FR2	
test-id3	FR3	
test-id4	FR4	
test-id5	FR5	
test-id6	FR6	
test-id7	FR7	
test-id8	FR8	
test-id9	FR9	
test-id10	FR10	
test-id11	FR11	
test-id12	FR12	
test-id13	FR13	
test-id14		NFR1
test-id15		NFR2
test-id16		NFR3
test-id17		NFR4
test-id18		NFR5
test-id19		NFR6, NFR7
test-id20		NFR8
test-id21		NFR11
test-id22		NFR12
test-id23		NFR13
test-id24		NFR14
test-id25		NFR15
test-id26		NFR16
test-id27		NFR10

Note: NFR9 regarding GPS location accuracy has been changed to a constraint due to its dependence on the GPS capabilities of the user's smartphone. These capabilities are not associated with the System, therefore it is not included in the Traceability Table.

6 Unit Test Description

6.1 Unit Testing Scope

Not applicable.

6.2 Tests for Functional Requirements

Not applicable.

6.3 Tests for Nonfunctional Requirements

Not applicable.

6.4 Traceability Between Test Cases and Modules

Not applicable.

7 Appendix

7.1 Usability Survey Questions

1. testid-10: Survey Question

   Rate this device's visual appeal on a scale of 1-10, with 10 being most visually appealing, 1 being least visually appealing.

Appendix — Reflection

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

1. What knowledge and skills will the team collectively need to acquire to successfully complete the verification and validation of your project? Examples of possible knowledge and skills include dynamic testing knowledge, static testing knowledge, specific tool usage etc. You should look to identify at least one item for each team member.

   Abi: In order to perform verification, specifically on accuracy, we need to be able to have an expected output to compare our real output to. This is especially applicable for battery life. We need to first be able to calculate an expected battery life to then validate it, which is a skill we will need to acquire.

   Stephen: We will need critical thinking skills to approach our device in such a way that we can complete a set of tests which have been created critically and will prove our products design and implementation. We will need technical knowledge on how our device works such that these validation tests will be a complete set.

   Elsa: The Team must build our collective knowledge in the area of structural analysis of engineering designs. In order to be able to effectively evaluate our designs for efficacy, robustness, longevity and usefulness, the Team will need to acquire or re-aquire engineering skills related to how to assess the structural integrity of mechanical devices.

2. For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?

   Abi: One approach to learn this skill would be doing research on the internet, and another might be to ask for the help of an expert. We are not familiar with someone knowledgeable on this topic, and therefore we will be learning how to calculate battery life through internet research.

   Stephen: To acquire critical thinking skills we will need to think outside the box and ask others their thoughts so that we can gain multiple

opinions and a different way of thinking. To acquire the technical knowledge about our device, many hours of research as well as talking with industry professionals will be needed.

Elsa: To acquire or re-quire skills related to structure analysis of mechanical designs, the Team can review previous coursework in Materials related to stress and strain. We can also take advantage of McMaster resources for physical testing, including online databases and professors who have experience in the subject.