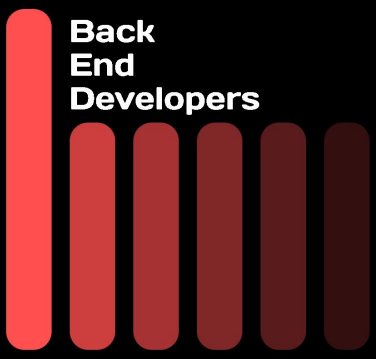
Verification and Validation Report: Mechatronics Engineering



Team #1, Back End Developers
Jessica Bae
Oliver Foote
Jonathan Hai
Anish Rangarajan
Nish Shah
Labeeb Zaker

April 4, 2023

1 Revision History

Date	Version	Notes
2023-03-08	1.0	Initial Documentation
2023-03-17	2.0	Modified requirements and traceability tables to match VnV Plan
2023-04-04	2.1	Incorporated TA feedback
2023-04-04	2.2	Added logo and style to the document

2 Symbols, Abbreviations and Acronyms

Throughout this document SI (Système International d'Unités) is employed as the unit system. In addition to the basic units, several derived units are used as described below. For each unit, the symbol is given followed by a description of the unit and the SI name.

Symbol	Unit	SI
m	length	metre
kg	mass	kilogram
s	time	second
°	angle	degree
rad	angle	radian
V	potential	volts
A	current	ampere
Ω	resistance	ohm (Ω= V A^{-1})
F	capacitance	farad
H	inductance	henry
N	force	newton (N = kg m s^{-2})
Pa	pressure	pascals (Pa = Nm^{-2})
Hz	frequency	hertz
J	energy	joule
W	power	watt (W = J s^{-1})

Table 1: Table of Basic Units

Below are some derived units that do not use a specific SI symbol.

Derived Unit	SI
area	m^2
volume	m^3
velocity	m s^{-1}
acceleration	m s^{-2}

Table 2: Table of Derived Units

2.1 Abbreviations and Acronyms

Symbol	Description
A	Assumption
ACC	Acceleration
BAR	Barometer
CSV	Comma-Separated Values
CV	Controlled Variables
DD	Data Definition
EMA	Ecological Momentary Assessment
FSM	Finite State Machine
GD	General Definition
GS	Goal Statement
GYR	Gyrometer
IM	Instance Model
LC	Likely Change
LSS	Lumbar Spinal Stenosis
MV	Monitored Variables
NFR	Non-Functional Requirements
OOP	Object Oriented Programming
PID	Proportional-integral-derivative
PS	Physical System Description
R	Requirement
SI	Système International d'Unités
SReS	School of Rehabilitation Sciences
SRS	Software Requirements Specification

Table 3: Table of Abbreviations and Acronyms

Contents

List of Tables

List of Figures

3 Design Verification

As outlined in section 4 of the VnV Plan, verification of the SRS and VnV Plan were performed to ensure that deisgn is meeting its requirements.

3.1 SRS Verification

Table 4: SRS Verification Result

Section	Description	Verification
Overall	System constrains, and monitored and controlled variables are consistent. Comparison to existing solutions is unambiguous. Project Goals are relevant to description of device.	Stakeholder Dr. Luciana Macedo approves of function and non function requirements, and requests that the team refines the definition of various activities to eliminate ambiguity.
4.1-4.3	The verification for this section will be performed by checking if system constrains and user characteristics are provided in detail without any ambiguity.	If Accelerometer is greater than or equal to 0.5 and gyroscope is greater than equal to 0.5, then trigeer counter.
5	Terminology and the use of monitored and controlled variables must be verified to be consistent and correct. Assumptions and project goals are clear.	All terminologies and variable names were consistent and correct.
7	Functional and non-functional requirements must be verified.	Speific tests for functional and non functional requirements are included as part of section 4 and 5.
8-12	Matrices and graphs must be verified according to requirements. In addition, verification must be performed to see if normal operation covers the required goals of the device.	Graph needs to be separated into each functionality. UI team to modify code such that the graphing functionality allows researchers todisplay selective information if necessary.
14-16	The FSM, Legal factors and Phase in Plan must be verified.	Data storage is changed from cloud service to local database to account for medical information privacy. FSM is verified and updated to reflect the new version of the design.

3.2 Design Verification Plan

Table 5: Design Verification Result

Test Categories	**Description**	**Verification**
Response Feedback Test	This test will test the majority of the sensors of the system. Sensor input will be pushed beyond the established boundaries in different testing scenarios, with feedback being observed for erraneous or strange readings.	Included as part of unit testing in 7
User Interface Test	Different elements of the user interface will tested by manipulating said elements in all possible scenarios. In addition, characteristics such as screen size, brightness, and visibility will be tested in different visual environments to determine whether or not they are visible in the correct ranges of use.	Included as part of non functional requirements test in 5
Durability Test	Each sensor will be tested for failure independence. Should one sensor fail, it must not impact the functionality of any other sensors of the system. To test this, individual sensors will be rendered non-functional, and other sensor values will be checked for erroneous readings.	None of the sensors are dependent of each other. Thus, failing one sensor will not impact the performance of other sensors. Specific dependencies of each module can be viewed in the MIS.
Database Scalability Test	This system's database will be tested for volume overload issues. Postman (a request tool) will be used to feed as many requests as possible to the database. The database will then be monitored for faults and errors.	No exception nor error arised.
Hardware Communication Test	Valgrind will be used at every interaction between hardware components to determine whether or not memory has been allocated properly. An execution tracker will also be used to ensure the workflow is functioning as expected.	Not applicable anymore, bluetooth functionality is implemented.

4 Functional Requirements Evaluation

These tests were performed to ensure that the functional requirements are met and that the results are recorded.

4.1 Duration Test

The following tests were performed to check that the device maintaines an ”ON” state throughout the entire duration of the monitoring period.

1. **DT_1: Regular Inputs**

   Initial State: Device waits for the monitoring period to be set up in the configuration of the device.

   Input: Monitoring Period [”Date”,”Time”] : [”03-11-2022”, ”05:30:PM”].

   Expected Results: Device turns off after monitoring period.

   How test was performed: The test was performed by passing in a valid monitoring period and ensuring that the device maintains power throughout this period.

   Observed Results: The device can last for around 2 days with the current battery circuitry. Any monitoring period of 2 days or less had inputs being collected as seen in the serial logs.

2. **DT_2: Invalid Inputs**

   Initial State: Device waits for the monitoring period to be set up in the configuration of the device.

   Input: Monitoring period [”Date”,”Time”] : [”3rd November 2022”, ”Five Thirty PM”].

   Expected Results: Device returns an error code to the error handler and asserts the Invalid Data Error (BED_ERR_INVALID_DATA).

   How test was performed: The test was performed by passing an invalid input and ensuring that the appropriate error code with its description is returned.

   Observed Results: When an unexpected input (wrong type) was passed, an error code with description (BED_ERR_INVALID_DATA) showed up on the screen.

3. **DT_3: Earlier Date**

   Initial State: Device waits for the monitoring period to be set up in the configuration of the device.

   Input: Monitoring period [”Date”,”Time”] : [”01-1-1999”, ”05:30:PM”].

   Expected Results: Device returns an error code to the error handler and asserts the Invalid Data Error (BED_ERR_INVALID_DATA).

How test was performed: The test was performed by passing an older date than the current date for configuration.

Observed Results: When an older date was passed, an error code with description (BED_ERR_INVALID_DATA) showed up on the screen.

4. **DT_4: Date Beyond Capabilities**

   Initial State: Device waits for the monitoring period to be set up in the configuration of the device.

   Input: Monitoring period [”Date”,”Time”] : [”9-12-2300”, ”05:30:PM”].

   Expected Results: Device returns an error code to the error handler and asserts the Invalid Data Error (BED_ERR_INVALID_DATA).

   How test was performed: The test is performed by passing a date greater than what the battery life of the device can support.

   Observed Results: When a date later than the predetermined maximum date was passed on, an error code with description (BED_ERR_INVALID_DATA) showed up on the screen.

4.2 Device should track Minor Movements

The following tests were performed to analyze the device's sensitivity.

Minor Tracking Test

1. **MTT_1: Regular Movement**

   Initial State: Device is worn with all systems working.

   Input: Tester performs activities at a regular/normal pace (regular walking is roughly 100 steps per minute).

   Expected Results: The steps taken should be incremented everytime the tester walks.

   How test was performed: The test was performed by attaching the sensor onto the tester's finger while they walk in normal pace. This was done to ensure that the device can work under normal scenarios.

   Observed Results: The steps counter was incrementing as expected and the count of total number of steps was almost very accurate, with regular walking step rate of 105 steps per minute.

2. **MTT_2: Slow Movement**

   Initial State: Device is worn with all systems working.

Input: Tester performs activities at a very slow pace.

Expected Results: Sudden changes in heart rate would be detected and displayed.

How test was performed: The test was performed by attaching the sensor onto the tester's finger and having them limp. This was done to ensure that the device can work for people with limited mobility.

Observed Results: Test Failed. The device failed to detect heart rate reliably (heart rate was expected to be roughly 70 bpm based on a third party heart rate sensor, but the device returned values greater than 200 bpm). Review of the code and selection of heart rate sensor is required.

4.3 Prompt Tests

The following tests were done to ensure that the proper prompts were displayed to the user when desired stimuli were detected. Tests involve generating the default and specific prompts for different detections.

1. **PRT_1: Test Prompt**

   Initial State: Device has just been reconfigured.

   Input: Device is turned on for the first time.

   Expected Results: Test prompt is displayed.

   How test was performed: The test was performed by turning on the device for the first time after reconfiguration. Upon turning on the device, a test prompt was generated to confirm that the prompting system was working correctly. The test prompt is the default prompt.

   Observed Results:Test Failed. Device failed to recognize the activity appropriately, and microcontroller (Seeeduino XIAO) failed to respond to any input. Hardware issue is yet to be resolved, consider using an alternative microcontroller.

2. **PRT_2: Activity Prompt**

   Initial State: Device is in the idle state.

   Input: Activity has been detected.

   Expected Results: Specific activity prompt is displayed.

   How test was performed: The test was done by having a tester perform one of the registered activities while the device was idle. This should result in a prompt generation for the tester that is based on the specific activity performed.

   Observed Results: The device displayed the specific prompts as shown below.

   Tracked activity: Tester slows down or comes to a stop.

Activity prompt: Are you in pain?
Possible answers: Yes or No

4.4 Customizable Thresholds

The following tests were done to ensure that the proper prompts were displayed to the user when activities are detected at configured thresholds. Tests involve generating the test prompt, and generating prompts for different activities.

1. **TT_1: Regular Inputs**

   Initial State: Device is in configuration mode.

   Input: Regular values for thresholds within limits.

   Expected Results: Configuration file generated successfully.

   How test was performed: The test was performed by setting the device to configuration mode and then setting valid values for the thresholds.

   Observed Results: The tester has the desired configuration thresholds for respective parameters. A configuration file was also created successfully.

   E.g. Speed Threshold Limits: 0 m/s $<$threshold $<$5 m/s

2. **TT_2: Below Lower Limit**

   Initial State: Device is in configuration mode.

   Input: Values for thresholds below lower limits.

   Expected Results: Configuration file generates BED_ERR_OUT_OF_BOUNDS.

   How test was performed: The test was performed by setting the device to configuration mode and then setting values for the thresholds above allowable limits.

   Observed Results: Configuration file generated an error code for the specific parameter in the file.

3. **TT_3: Above Upper limit**

   Initial State: Device is in configuration mode.

   Input: Values for thresholds above upper limits.

   Expected Results: Configuration file generates BED_ERR_OUT_OF_BOUNDS.

   How test was performed: The test was performed by setting the device to configuration mode and then setting values for the thresholds below allowable limits.

   Observed Results: Configuration file generated an error code for the specific parameter in the config file.

4. **TT_4: Invalid Value**

   Initial State: Device is in configuration mode.

   Input: Invalid values for thresholds.

   Expected Results: Config File generates BED_ERR_INVALID_DATA.

   How test was performed: The test was performed by setting the device to configuration mode and then setting invalid values for the thresholds.

   Observed Results: Configuration file generated an error code for the specific parameter in the file.

5. **TT_5: No Value**

   Initial State: Device is in configuration mode.

   Input: No values for thresholds.

   Expected Results: Configuration file generates BED_ERR_INVALID_DATA.

   How test was performed: The test was performed by setting the device to configuration mode and then assigning no values for the thresholds.

   Observed Results: Configuration file generated an error code for the specific parameter in the file.

4.5 Data Storage

The following tests were done to ensure that data is stored when necessary. Tests include checking when storage buffer was full, when prompts were generated and when sensor data needed to be logged.

1. **DST_1: Storage Buffer Full**

   Initial State: Device is in an idle state.

   Input: Activity detected causing prompt to be generated (internal storage is full).

   Expected Results: Data storage system generates BED_ERR_MEMORY_FULL

   How test was performed: The test was performed by first loading the internal memory buffer with garbage values so that it was nearly or completely full. Then a registered activity was triggered, generating the prompts.

   Observed Results: Once the prompts were answered, the system did not have enough memory to store the new values thus resulting in the expected error code.

2. **DST_2: Prompt Generated**

   Initial State: Device is in an idle state.

Input: Activity detected causing prompt to be generated.

Expected Results: Prompt response is saved into the internal memory.

How test was performed: The test was executed by performing a registered activity and ensuring that the prompt generated is answered and its result is stored in the internal storage buffer.

Observed Results: The responses were saved in the internal memory which were later logged to check if they were stored correctly.

e.g. Registered Activity: Participant slows down or comes to a stop for an extended period of time.

Prompt Generated: Are you in pain?
Possible Answers: Yes or No

Prompt Generated: Do you need assistance?
Possible Answers: Yes or No

3. **DST_3: Sensor Storage**

Initial State: Device is in an idle state.

Input: Activity detected causing prompt to be generated.

Expected Results: Specific sensor values that triggered a prompt are stored in the internal memory.

How test was performed: The test was performed by performing a registered activity and ensuring that the sensor values that caused the prompt are stored in the internal storage buffer.

Observed Results: The raw sensor data values that triggered the prompts were stored along with the responses from the users.

4.5.1 Data Extraction

The following tests were done to ensure that data can be extracted and presented in a graphical manner deemed acceptable for the purpose of EMA analysis.

1. **DXT_1: Extracting Data**

Initial State: Device is connected to the device manager.

Input: A command that tells the device manager to extract all data from the internal memory.

Expected Results: Extracted data is sent to the host software where it is stored in the database.

How test was performed: The test was performed by first running the device as intended and waiting for a small monitoring period to finish. After this, the device was connected to the host software with the help of the device manager driver. Once connected, the user can start the extraction process and should be able to see all the relevant data in a presentable manner.

Observed Results: The data was read in the host software and updated in the database.

2. **DXT_2: Extracting No Data**

   Initial State: Device is connected to the device manager.

   Input: A command that tells the device manager to extract all data from the internal memory.

   Expected Results: Device Manager returns a BED_ERR_EMPTY_DATA error

   How test was performed: The test was performed by first deleting all the contents of the internal memory prior to connection with the host software. Once connected and the extraction process began, data was attempted to be extracted.

   Observed Results: There was an error code returned on the display.

3. **DXT_3: Extracting Corrupted Data**

   Initial State: Device is connected to the device manager.

   Input: A command that tells the device manager to extract all data from the internal memory.

   Expected Results: Device manager returns a BED_ERR_INVALID_DATA error.

   How test was performed: The test was performed by filling it with garbage values prior to connection with the host software. Once connected, the data was attempted to be extracted from the memory.

   Observed Results: Since the data stored were garbage values and not the expected data types or limits, the device displayed an error.

5 Nonfunctional Requirements Evaluation

5.1 Hardware Safety

These tests were performed to test the safety of the device concerning electrical and hardware components. It is important that the users of the device are kept safe, that any systems in place to protect them from electrical shocks are functioning properly and that the indicator LED functions as expected under certain conditions.

1. **HST_1: High Voltage**

   Initial State: Device begins in ”OFF” mode.

   Input: When the device is turned on, a voltage amount greater than the expected voltage is applied.

   Expected Results: Device detects high voltage and indicates that voltage is high using the indicator LED. The breaker or fuse then trips or the voltage is sent to ground.

   Test Case Derivation: N/A

   How test was performed: Use a power supply to oversupply the system with voltage.

   Observed Results: Respective indicator lit up as expected, corresponding to high voltage. Fuse tripped to prevent electrocution.

2. **HST_2: Normal Voltage**

   Initial State: Device begins in ”OFF” mode.

   Input: When device is turned on a normal voltage is applied.

   Expected Results: Device powers on, no issues are detected.

   Test Case Derivation: N/A

   How test was performed: The power supply will be used to provide the expected voltage to the device.

   Observed Results: The device operated normally.

3. **HST_3: Low Voltage**

   Initial State: Device begins in ”OFF” mode.

   Input: When device is turned on a voltage below the expected amount is applied.

   Expected Results: If device powers on, indicator LED indicates that the voltage is lower than expected, device goes into power saving mode.

   Test Case Derivation: N/A

   How test was performed: Use power supply to provide a voltage to the device in amount lower than the expected voltage.

   Observed Results: As expected, respective LED was lit to indicate low voltage.

5.2 Re-usability

These tests evaluate the re-usability of various hardware systems of the device. The device should be reusable by multiple participants to prevent electronic waste and reduce cost for the researchers.

1. **RT_1: Battery Charge/Discharge**

   Initial State: Battery is fully discharged.

   Input: Connect battery to the charging cable.

   Expected Results: Battery is fully charged without any issue.

   Test Case Derivation: N/A

   How test was performed: Battery component was isolated from the device and fully charged and discharged couple times.

   Observed Results: Battery was fully charged and device was safe to use.

2. **RT_2: Battery Charge Protection**

   Initial State: Battery is fully charged.

   Input: Connect battery to the charging cable.

   Expected Results: Battery will not overcharge.

   Test Case Derivation: N/A

   How test was performed: Battery component was isolated from device and fully charged. The battery was then plugged in to the charger for 5 minutes.

   Observed Results: Battery was not being overcharged nor overheated, thus device was safe to use.

5.3 Accuracy

The following set of tests were executed to ensure that the device can recognize its state and perform its main functionality in accurate manner, based on the state. The following functional requirements test cases are applicable for accuracy:

- MTT_1: Regular movement
- MTT_2: Slow movement
- PRT_1: Test prompt
- PRT_2: Activity prompt

5.3.1 Usability

The following set of tests were done to ensure that the device is usable by target user population is not problematic in terms of maintenance and user interactions.

1. **UT_1: Device UI**

   Initial State: The device is turned on and a question prompt is generated on screen.

Input: Testers are asked to answer the questionnaire set without assistance in less than 1 minute.
Expected Results: All testers are able to answer their prompt successfully.

How test was performed: A tester group of people with age of 40 or older was be asked to receive a powered-on device and asked to respond to a set of 3 questions in 1 minute or less. These questions were randomly selected from the following 10 questions.

- Q1: Are you a dog person or a cat person?
- Q2: Are you currently inside of a building?
- Q3: Are you currently a student?
- Q4: Did you make your bed this morning?
- Q5: Do you feel tired right now?
- Q6: Do you prefer coffee or tea?
- Q7: Do you have a G driver's license?
- Q8: Do you feel sleepy right now?
- Q9: Which is better: Summer vs Winter
- Q10: Is it daytime right now?

Observed Results: The UI was considered to be intuitive and user-friendly, testers were able to navigate through and answer propmpts without any problem. Table 4 lays out all the responses made by the four participating testers.

Table 6: UT_1 Survey Response

	User A	**User B**	**User C**	**User D**
Q1	Dog	Dog	Cat	Dog
Q2	Yes	Yes	Yes	Yes
Q3	No	Yes	No	No
Q4	No	No	No	No
Q5	Yes	Yes	Yes	No
Q6	Coffee	Tea	Tea	Tea
Q7	Yes	No	No	Yes
Q8	No	No	No	No
Q9	Summer	Winter	Summer	Winter
Q10	Yes	Yes	Yes	Yes

2. **UT_2: Comfortable Device**

Initial State: The device is turned on and left on idle mode.

Input: Testers are asked to wear the device for 24 hours.

Expected Results: Testers fill out a survey form regarding the comfort of the device on their body.

How test was performed: At the end of their 1 day cycle, testers will be asked to fill out an online form indicating how comfortable they felt the device was regards to weight, shape, stability, etc. The following questions will be asked.

- Did the device every fall off? If so, please record the following for each case: What you were doing each time? When did the incident happen?
- How do you feel regarding the weight of the device? Was it too heavy or too light?
- How do you feel regarding the texture of the device? Did you find it uncomfortable in any way?

Observed Results: Testers filled out a survey and the responses were mostly positive.

3. **UT_3: Reusability**

Initial State: The device is cleaned using isopropyl alcohol wipes.

Input: Testers visually inspect how clean the device is.

Expected Results: The device is deemed clean and the device is not harmed.

How test was performed: A member of Back End Developers cleaned the device, which was later inspected for safety and usage to check if the device turned on and functioned normally.

Observed Results: The device was functionally as normal despite being cleaned after usage.

4. **UT_4: Host Software UI**

Initial State: Main page is displayed after login page.

Input: Testers are asked to filter out database record of anyone 70 years old and write down the first names.

Expected Results: John, Marry, and Bill.

How test was performed: The software application is presented with its main page. Testers are required to navigate through the software to use the dataview filtering functionality and filter out data as requested by the test. They are then asked to write down their responses on a piece of paper. Later, the papers are collected and compared for validation.

Observed Results: Testers successfully identified the 3 first names, as expected. The following table lays out all their responses.

Table 7: UT_4 Survey Response

	User A	**User B**	**User C**	**User D**
Name1	John	John	Mary	John
Name2	Mary	Mary	Bill	Mary
Name3	Bill	Bill	John	Bill

5.3.2 Performance

The following set of questions was asked to ensure that the device was capable of meeting user expectations.

1. **PT_1: Data Transfer Time**

   Initial State: The device has collected a complete set of data.

   Input: The device is connected to the host software and connection is established for data transfer.

   Expected Results: Data transfer to finish within the specified parameter, TRANSFER_TIME.

   How test was performed: 2 group members of the Back End Developers performed seperate data transfer tests 4 times and recorded the total transfer time for each execution. All 8 tests should result in execution time less than or equal to TRANSFER_TIME.

   Observed Results: The data was transferred within the specified parameter, thus indicating device performed well in terms of transfer rates.

   TRANSFER_TIME: Set to 60 seconds

File Size	Transfer Time	Errors	Test Result	Notes
2 kB	2.66 s	None	Pass	None
10 kB	13.27 s	None	Pass	None
8.5 kB	10.51 s	None	Pass	None
5 kB	6.94 s	None	Pass	None
11 kB	14.10 s	None	Pass	None
18 kB	N/A	COM port 7 not open	Result rejected	Cable was plugged in incorrectly.
20 kB	25.68 s	None	Pass	None
35 kB	36.85 s	None	Pass	None

Table 8: Data Transfer Time Testing Data

2. **PT_2: Battery Life**

Initial State: The device is turned on and left on idle mode.

Input: Testers wear the device for standard monitoring period.

Expected Results: The total amount of time before the device runs out of battery is greater or equal to the parameter, BATTERY_LIFE.

How test was performed: 2 group members of the Back End Developers wore the device for standard monitoring period consecutively in their daily basis. At the end of each calendar day, they are prompted to record their response to a simple question: ”Does the device still have battery life left?”.

Observed Results: The group members answered the prompt. The battery limit was greater than equal to the predetermined parameter, BATTERY_LIFE.

BATTERY_LIFE: 24 Hours

Battery Level	Errors	Prompt Response	Notes
22%	None	”Yes”	None
31%	None	”Yes”	None

Table 9: Battery Performance Testing Data

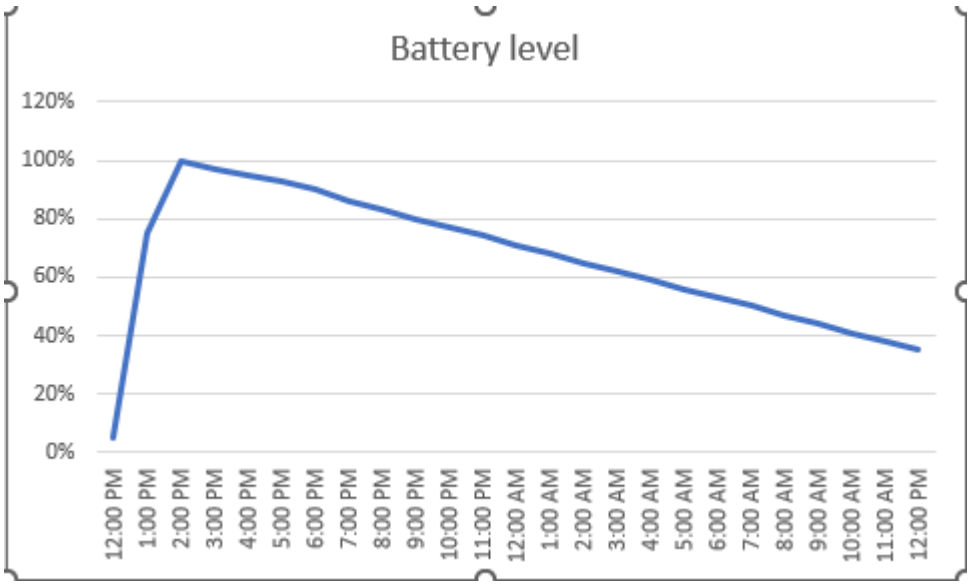


Figure 1: The battery levels in a span of 24 hours

5.3.3 Data Security Tests

These tests were designed to test the security of the system and the accessibility of important medical records. This is important to protect the user's privacy by avoiding it to fall in the wrong hands. Thus only authorised users can access the data.

1. **DSQT_1: Records Safety Test**

   Initial State: Device is powered on.

   Input: Data transfer cable is plugged in from device to a generic PC, a tester tries to access the database and modify the data within.

   Expected Results: Database inaccessible and requests an administrative security key. The tester should not be able to access nor modify the data.

   Test Case Derivation: N/A

   How test was performed: Device was given to a tester with required technical skills but without the administrator security key. The tester attempted to access the database.

   Observed Results: Databases couldn't be accessed and was prompting for a security key, implying the database is safe.

2. **DSQT_2: Data Storage Selection Test**

   Initial State: Device is powered on.

   Input: Less than 3 seconds of limping from tester.

   Expected Results: Corresponding sensor data is not included in sdcard data.

   Test Case Derivation: N/A

How test was performed: One member of Back End Developers wears the device and pretends to ”limp” for 3 seconds, to create garbage sensor data. Then sd card is inserted into a PC to examine the content.

Observed Results: Corresponding sensor data is not included in sdcard data.

6 Comparison to Existing Implementation

Current motion detecting smart watches, such as the Apple watch or the Samsung Galaxy watch, are desgined for healthy and active population. The EMAnator is specifically geared towards older adults who have chronical back pain. Therefore, it is desgined to capture minor and subtle movements accurately.

7 Unit Testing

No.	Name	Ref.	Action	Expected Output	Actual Output	Result
1	bed_HR_setup	R2	Power on the device	”PulseSensor object created”, LED flash per each heartbeat	”PulseSensor object created”, LED flash per each heartbeat	Pass
2	bed_HR_detect	R2	Walk for 10 seconds while wearing the device, then stop	”HeartBeat Detected. BPM: 70”	”HeartBeat Detected. BPM: 130”	Fail
3	bed_HR_detect	R2	Shake the watch without anyone wearing it	”HeartBeat Not Detected.”	”HeartBeat Not Detected. ”	Pass
4	bed_MPU _setup	R2	Power on the device	“Calibrating MPU in 4 seconds! Hold it still! avg X acc: 20, , avg Y acc: 20, avg Z acc: 20”	“Calibrating MPU in 4 seconds! Hold it still! avg X acc: 20, , avg Y acc: 20, avg Z acc: 20”	Pass
5	bed_MPU _detect	R2	Limp	“limp count + limp count + limp count + limp count + limp count + limping. Stepcount: 5”	“limp count + limp count + limp count + limp count + limp count + limping. Stepcount: 5”	Pass
6	bed_MPU _detect	R2	Walk	“Stepcount: 13”	“limp count + limp count + limp count + Stepcount: 10”	Fail
7	bed_MPU _detect	R2	Stay still	“Stepcount: 0”	“Stepcount: 0”	Pass
8	bed_init_display	NFR2	No action	Black screen on	Black screen on	Pass

No.	Name	Ref.	Action	Expected Output	Actual Output	Result
9	bed_scroll_test	NFR2	Scroll through 2 adjacent bezels in counter clock-wise directio.	Selection goes up	Selection goes up	Pass
10	bed_scroll_test	NFR2	Touch 2 non-adjacent bezels in counter clock-wise direction	Selection stays as is	Selection stays as is	Pass
11	bed_scroll_test	NFR2	Scroll through 2 adjacent bezels in counter clock-wise direction with a 2 seconds delay in between	Selection goes up	Selection goes up	Pass
12	bed_splash _screen	NFR2	Power on the device	”Back End Developers”	”Back End Developers”	Pass
13	bed_display _one_line	NFR2	Select a response for generated prompt	Screen clears prompt list and only displays current date and time	Screen clears prompt list and only displays current date and time	Pass
14	OpenFile	R6	Press button and choose the ”data.txt” file	Contents of ”data.txt” is displayed on UI	Contents of ”data.txt” is displayed on UI	Pass
15	bed_display _prompt	R3	Move around for 5 seconds and then stop for 5 seconds	”Are you in pain?”, Yes, No	”Are you in pain?”, Yes, No	Pass
16	bed_format _prompt	R3	Limp with the device to activate prompt generation	Prompt text color is white while background is black	Prompt text color is white while background is black	Pass

No.	Name	Ref.	Action	Expected Output	Actual Output	Result
17	OpenSerial	R6, NFR14	Connect device to computer via USB port	UI displays green dot	UI displays green dot	Pass
18	bed_init_rtc	R6, NFR1	Power on the device	”March 5th, 16:21”	”March 5th, 16:21”	Pass
19	bed_display_info	NFR2	Power on the device	”March 5th, 16:23”	”March 5th, 16:23”	Pass
20	bed_set _explicit_date _time	R6, NFR1	Disconnect power source, reconnect, and power on the device	”March 5th, 16:46”	”March 5th, 16:46”	Pass
21	bed_touch _detect	NFR2	Touch 2 bezels in clockwise direction	1	1	Pass
22	__init__	N/A	Run the host software	Login page appears	Login page appears	Pass
23	saveAsTxt	R6	Press Save as button and choose a directory to save file in.	Config.txt file created and saved	Config.txt file created and saved	Pass
24	BackToMain	NFR2	Press the main button	Main screen appears and current screen closes	Main screen appears and current screen closes	Pass
25	InsertDB	R6, NFR14	Press save button	Data appears on database table	Data appears on database table	Pass
26	LoadDb	R6, NFR14	Press load button	Data appears on software UI table	Data appears on software UI table	Pass
27	dataHead	R6, NFR14	Press load button	Data appears on software UI table	Data appears on software UI table	Pass

No.	Name	Ref.	Action	Expected Output	Actual Output	Result
28	search	R6, NFR14	Filter-search participant first name of “John”	All participant record with first name “John” appears	All participant record with first name “John” appears	Pass
29	OpenGraph	R7	Press graph button.	Empty graph chart appears	Empty graph chart appears	Pass
30	OpenHeart	R7	Press heart rate button	Only heart rate graph appears	Only heart rate graph appears	Pass
31	OpenSteps	R7	Press steps button	Only number-of-steps graph appears	Only number-of-steps graph appears	Pass
32	OpenActivity	R7	Press activity button	Only activity graph appears	Only activity graph appears	Pass
33	plot	R7	Press steps button	Only number-of-steps graph appears	Only number-of-steps graph appears	Pass
34	loginAttempt	NFR8, NFR14	ID: ”admin”, PW: ”capstone”	Main screen appears	Main screen appears	Pass
35	loginAttempt	NFR8, NFR14	ID: ”wr1ong”, PW: ”creden5tials”	”Wrong credentials!”	”Wrong credentials!”	Pass
36	showRecord Window	NFR2	Press records button	Records window appears and main window closes	Records window appears and main window closes	Pass
37	showCreate Records	NFR2	Press create records button	Create Records window appears and main window closes	Create Records window appears and main window closes	Pass

No.	Name	Ref.	Action	Expected Output	Actual Output	Result
38	showConfig View	NFR2	Press configuration button	Configuration window appears and main window closes	Configuration window appears and main window closes	Pass
39	showDataView	NFR2	Press data view button	Data view window appears and main window closes	Data view window appears and main window closes	Pass

8 Changes Due to Testing

8.1 Heart Rate Sensor Changes

After testing out each unit of the device system and failing unit test MTT_2, the type of heartrate sensor has been changed. Below table outlines the rejected heartrate sensor candidates for EMAnator.

Heart Rate Sensor Options:

Table 10: Tested Heart Rate Sensors

Image	Model	Status	Reason for Status
See Figure 2	Chaney Electronics Inc. G21394	Rejected	Due to the Infrared LED's Placement and the photoelectic sensor's placement, it is impossible to integrate the module with a device sitting on the wrist.
See Figure 3	SparkFun Pulse Oximeter and Heart Rate Sensor - MAX30101 & MAX32664 (Qwiic)	Rejected	Chronically out of stock, and requires extra complexity to be built into the PCB in order for this to function.
See Figure 4	Grove - Fingerclip Heart Rate Sensor	Rejected	Size too large, and issues with skin contact to the sensor results in garbage data.
See Figure 5	Grove - Earclip/Finger-clip Heart Rate Sensor	Rejected	Sensor attachment to the ear was considered too disruptive to the participant's daily activities.
See Figure 6	Comimark 2Pcs Heart Rate Pulse Sensor Sensor Module for Arduino Raspberry pi	Rejected	Build quality was unacceptably bad. Ordered name-brand version of this product next.
See Figure 7	PulseSensor.com Pulse Sensor	Accepted	Build quality was better than knock-offs from before. Skin contact issue still present, but attempts are being made to solve this issue using software.

Figure 2: Chaney Electronics Inc. G21394

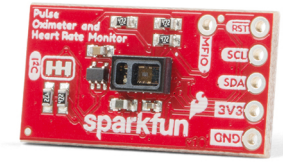


Figure 3: SparkFun Pulse Oximeter and Heart Rate Sensor - MAX30101 & MAX32664 (Qwiic)

Figure 4: Grove - Finger-clip Heart Rate Sensor

Figure 5: Grove - Ear-clip/Finger-clip Heart Rate Sensor

Figure 6: Comimark 2Pcs Heart Rate Pulse Sensor Sensor Module for Arduino Raspberry pi

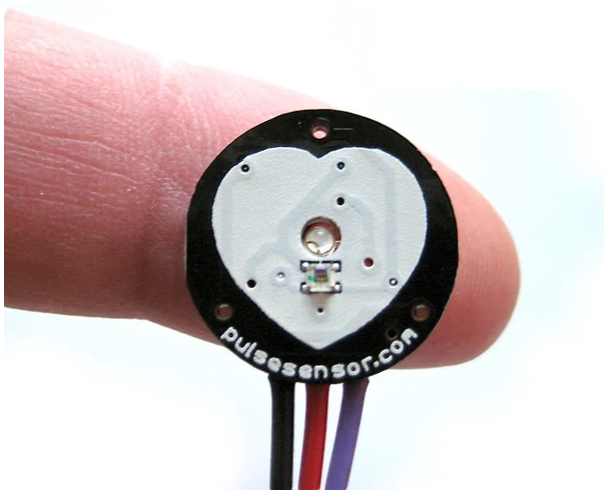


Figure 7: PulseSensor.com Pulse Sensor

8.2 Last Minute Microcontroller Changes

A major change to the design was a last minute switch to the ESP32 microcontroller after originally working with the Seeedunio Xiao microcontroller until Revision 0. The decision to switch had to be made for reasons explained below. But, unfortunately this change was only made after significant progress had already been made by the Back End Developers. Going as far as designing and printing a custom PCB with all required hardware and sensors, and integrating the codebase with the Seeeduino Xiao and PCB prior to the Revision 0 demonstration.

The Seeedunio Xiao was originally chosen because it runs on a similar ARM based platform compared to the Arduino Nano which was used for testing. This made development easier and provided enough processing power to meet the needs of the design. The Seeduino SAMD21 Cortex M0+ chip has as clock speed of 48 MHz which is 4 times faster the clock speed on a typical Arduino Nano of 16 MHz. Additionally, the Seeeduino is a much smaller device than the Nano and could be directly soldered onto the 44mm diameter custom PCB. It would not take up significant space, leaving room for the rest of the sensors and hardware and allowing the design to fit within standard smartwatch dimensions.

During development and testing the Seeeduino did not present any issues and development was progressing well. However, leading up to the Revision 0 demo multiple Seeeduinos entered into an irrecoverable state. The reason for why this occurred is still unclear. However, after reading online the Back End Developers speculate two main reasons. First, it is possible that the bootloader region was not properly protected from the factory and it was accidentally overwritten by the software. Second, it is possible that there was not adequate power/surge protection for the Seeeduino since it seemed to fail some time after being connected to an external power supply for sensor testing but did not fail at all during use with a USB cable connected to a computer.

After more troubleshooting online it was discovered that this failure is a somewhat well documented issue with no easily available solution. This left the Back End Developers with no choice but to find an alternative microcontroller that did not have a major design flaw and would still meet the needs and constraints of the design. The alternative microcontroller that was chosen was the ESP32. The ESP can also run the Back End Developers code which reduces the total amount of refactoring that needs to be done. However, the pinout and size of the device differs from the Seeeduino meaning that the new custom PCB has to be made larger and the routing updated. The expected fabrication and shipping time will take over 1 week based on prior experience.

Due to the size and nature of the problem that was discovered at such a late stage in the project, there is a risk to the completion of the project as a whole. The time to integrate all the sensor code, display code and database code will be very limited once the new PCB arrives. Additionally, there is a risk that the PCB design does not perform the required functionality in the way that was intended. With a lead time of over 1 week if the PCB does not function as required the Back End Developers will have to fall back on a design that focuses on proof of concept rather than a fully fleshed out prototype. The proof of concept design will include functionality of sensing movement, prompting the user, collecting data, and enabling the researcher to view the data as intended by the functional requirements.

Figure 8: Seeeduino Xiao

Figure 9: ESP 32 - WROOM

9 Automated Testing

N/A

10 Trace to Requirements

	DT1	DT2	DT3	DT4	MTT1	MTT2	PRT1	PRT2	TT1
R1	X	X	X	X					
R2					X	X			
R3							X	X	
R4									X
R5									
R6									
NFR1					X	X	X	X	
NFR12							X		

Table 11: Requirements Traceability Matrix Part 1

	TT2	TT3	TT4	TT5	DST1	DST2	DST3	DXT1	DXT2	DXT3
R1										
R2										
R3										
R4	X	X	X	X						
R5					X	X	X			
R6								X	X	X

Table 12: Requirements Traceability Matrix Part 2

	HST1	HST2	HST3	RT1	UT1	UT2	UT3	PT1	PT2	DSQT1
NFR2					X	X	X			
NFR3									X	
NFR4						X				
NFR5							X			
NFR6	X	X	X							
NFR7				X						
NFR8										X
NFR9					X					
NFR10						X				
NFR11	X	X	X	X						
NFR13								X		
NFR14										X

Table 13: Requirements Traceability Matrix Part 3

11 Trace to Modules

	DT1	DT2	DT3	DT4	MTT1	MTT2	PRT1	PRT2	TT1
M1	X								
M2									X
M3									
M4					X	X		X	
M5							X	X	
M6							X	X	
M7	X	X	X	X					
M8									
M9									X
M10					X	X		X	

Table 14: Module Traceability Matrix Part 1

	TT2	TT3	TT4	TT5	DST1	DST2	DST3	DXT1	DXT2	DXT3
M1										
M2	X	X	X	X				X	X	X
M3					X	X	X	X	X	X
M4					X	X	X	X	X	X
M5										
M6					X	X	X			
M7						X	X			
M8										
M9	X	X	X	X						
M10										

Table 15: Module Traceability Matrix Part 2

	HST1	HST2	HST3	RT1	RT2	UT1	UT2	PT1	PT2	DSQT1
M1	X	X	X	X	X				X	
M2								X		X
M3										
M4										
M5						X				
M6						X				
M7										
M8										
M9										
M10						X	X			

Table 16: Module Traceability Matrix Part 3

12 Code Coverage Metrics

N/A

Appendix — Reflection

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

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

The process of developing this project from a concept to its current state has been a humbling experience. We (Team #1: The Back End Developers) believed that we comprehensively covered the timeline of our project in the VnV Plan. We had included test cases that seemed to cover all possible points of failure, took input from the TA comments, and reviewed the document with final unanimous approval from every team member.

We thought we had covered all our bases. We were wrong. Little did we know, that our microcontroller (8.2) and heart rate sensors (Table 8.1) would fail in spectacular ways. These served as a wake-up call for the team; we realized that we were not yet at a point in our project where everything was smooth sailing.

The main changes we made from the VnV Plan were changing the microcontroller we were using, along with the heart rate sensor. However, other changes were made as well. We distributed the team much more dynamically than before. Initially, our members had static roles (either on the hardware aspect of the project, software aspect, UI aspect, etc.) but we realized after the VnV Plan that we had to move our resources around according to the situational needs of the project. The greatest resource we have are our team members, and so we have learned to shuffle them around wherever we need their skills. We now regularly split into sub-teams whenever a task needs reviewing. For example, whenever new documentation has been written, two team members who have more time on their hands pore over the rubric and the documentation to see if there are any missed points.

From the technical side, we have had to search for a new microcontroller, eventually settling on the ESP32 MCU. We previously wondered why few people selected the Seeeduino XIAO (our old microcontroller) for projects similar to ours. Now we know it is because of their notorious unreliability. We have learned our lesson and scrutinized the available choices out there and found that the ESP32 fit our requirements and also was known to have extensive documentation and few reliability problems. We also took every hear rate sensor available to us and rigorously tested the values they gave us, and the results were disappointing; only one sensor available was capable of returning reasonably accurate heart rate values. Once again, a lesson has been learned.

All these changes prompted us to take a look at our project in general; was it up to the standards that we wanted it to be? For many parts, it was. Our team runs smoothly, with members working together as a conflict free, well-oiled machine. Considering the chatter we have been picking up from other teams and the bickering they are occasionally embroiled in, we are thankful that we have never encountered any issues like this. On the organizational side, we also believe we are up to par. Our Trello board remains the crux of our planning, and our use of git has allowed us to work in parallel seamlessly.

However the issues begin to appear when it comes to the product itself. Our goals were too ambitious; we found that many types of biometric sensors were prohibitively disruptive to users of our device. We have then settled on using only the MPU and heart rate sensor for data collection. This severely limits us in our capability to gather diverse sets of data from which we can draw conclusions, but that is just something that our team is going to have to handle (possibly finding a software based analysis solution).

There is also still much work to do, both in terms of developing our solution and in improving our quality of work. Regarding our solution, we have to finalize the PCB of our design and settle on the optimal way on designing the physical form factor. In addition, we have to clean up our codebase so it is up to modern coding standards and enables us to build in more security features into our solution. When it comes to how our team functions, our team members must commit their changes to our GitHub much more frequently, as it allows us to track our changes and revert to previous versions in case of issues with the latest commits. Also, we must be more proactive in documenting our decisions as they are made. We have found that when we review old documents or code from the beginning of our project, we become confused as to why we made certain decisions. We want to leave a paper trail from here on, and document why we choose to take our project in certain directions.

We believe that these changes are a part of the life-long learning process. In future projects, the skills that we have begun to build here will help us spot issues and put out fires before they spin out of control. We have learned to scrutinize off-the-shelf solutions more stringently, to have a wide selection of options when it comes to any form of components that we wish to build into a solution, and to use the organizational tools that we have set up to help us to track our success in meaningful ways.

For now, our focus is entirely on completing this project with quality, speed, and professionalism. We will not settle for any less.