

Verification and Validation Report: Mechatronics Engineering

Team #1, Back End Developers

Jessica Bae

Oliver Foote

Jonathan Hai

Anish Rangarajan

Nish Shah

Labeeb Zaker

March 6, 2023

1 Revision History

| Date | Version | Notes |
|--------|---------|-------|
| Date 1 | 1.0 | Notes |
| Date 2 | 1.1 | Notes |

2 Symbols, Abbreviations and Acronyms

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

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

Contents

| | | |
|-----------|---|-----------|
| 1 | Revision History | i |
| 2 | Symbols, Abbreviations and Acronyms | ii |
| 3 | Functional Requirements Evaluation | 1 |
| 3.1 | Duration Test | 1 |
| 3.2 | Device should track Minor Movements | 2 |
| 3.3 | Prompt Tests | 3 |
| 3.4 | Customizable Thresholds | 4 |
| 3.5 | Data Storage | 5 |
| 3.5.1 | Data Extraction | 7 |
| 4 | Nonfunctional Requirements Evaluation | 8 |
| 4.1 | Hardware Safety | 8 |
| 4.2 | Re-usability | 9 |
| 4.3 | Accuracy | 10 |
| 4.3.1 | Usability | 10 |
| 4.3.2 | Performance | 12 |
| 4.3.3 | Data Security Tests | 13 |
| 5 | Comparison to Existing Implementation | 13 |
| 6 | Unit Testing | 13 |
| 7 | Changes Due to Testing | 14 |
| 8 | Automated Testing | 17 |
| 9 | Trace to Requirements | 17 |
| 10 | Trace to Modules | 19 |
| 11 | Code Coverage Metrics | 21 |

List of Tables

| | | |
|---|--|----|
| 1 | Data Transfer Time Testing Data | 12 |
| 2 | Tested Heart Rate Sensors Part 1 | 15 |
| 3 | Tested Heart Rate Sensors Part 2 | 16 |
| 4 | Requirements Traceability Matrix Pt1 | 17 |
| 5 | Requirements Traceability Matrix Pt2 | 18 |
| 6 | Requirements Traceability Matrix Pt3 | 18 |
| 7 | Module Traceability Matrix Pt1 | 19 |

| | | |
|---|--------------------------------|----|
| 8 | Module Traceability Matrix Pt2 | 19 |
| 9 | Module Traceability Matrix Pt3 | 20 |

List of Figures

This document ...

3 Functional Requirements Evaluation

This tests were performed to ensure that the functional requirements were met and the results were noted.

3.1 Duration Test

The following tests were performed to check whether the device maintained an "ON" state throughout the 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 in lesser than 2 days 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 the appropriate error code with the description is returned.

Observed Results: When an unexpected input (wrong type) was passed, an error code with a very intuitive 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 a very intuitive 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 newer than the maximum date that can be stored was passed, an error code with a very intuitive description (BED_ERR.INVALID_DATA) showed up on the screen.

3.2 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.

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

How test was performed: The test was performed by attaching the sensor onto the volunteer's finger who walked as usual. This was done to ensure that the device can work under normal scenarios.

Observed Results: The steps were incrementing as expected and almost very accurate value of how many steps were logged in serial.

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 volunteer's finger who walked as if he was limping. This was done to ensure that the device can work for people with limited mobility.

Observed Results: The steps were incrementing as expected and almost very accurate value of how many steps were logged in serial. Since they weren't above threshold, the device detected that the tracker were limping

3.3 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, there was a test prompt that confirmed that the prompting system was working correctly. The test prompt is the default prompt.

Observed Results: The default prompt showed up. When a desired activity was detected, a set of specific prompts were displayed, for which responses were recorded.

Test Prompt: Is this Device on?

Possible Answers: Yes or No

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 volunteer perform one of the registered activities when the device was idle. This should result in a prompt for the volunteer that is generated 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

3.4 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, 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: Config 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 config file was also created successfully.

Eg:

Speed Threshold:

Limits: 0m/s <Threshold <5m/s

2. TT_2: Below Lower Limit

Initial State: Device is in Configuration mode.

Input: Values for thresholds below lower limits.

Expected Results: Config File generates a 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 config file.

3. TT_3: Above Upper limit

Initial State: Device is in Configuration mode.

Input: Values for thresholds above upper limits.

Expected Results: Config File generates a 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 a 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 config file.

5. TT_5: No Value

Initial State: Device is in Configuration mode.

Input: No values for thresholds.

Expected Results: Config File generates a 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 config file.

3.5 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 a 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/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.

eg: 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.

3.5.1 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 on 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.

4 Nonfunctional Requirements Evaluation

4.1 Hardware Safety

These tests were performed to test the safety of the device concerning electrical components and hardware. It is important that the users of the device are kept safe and 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 over expected voltage is applied.

Expected Results: Device detects high voltage and indicates that voltage is high using an 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 to correspond to high voltage and 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:

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

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

4. HST_4: Frequency Detection

Initial State: Device begins in off mode

Input: Device is turned on in normal operating state

Expected Results: Frequency analysis conducted on the circuitry components

Test Case Derivation: N/A

How test was performed: Using a frequency analysis software such as NI Multi-sim and an Oscilloscope, graphs will be generated for different components of the hardware to determine normal operating frequency compared to industry standards.

Observed Results: Based on the graphs generated by Multisim, we were able to determine that frequencies were detected and had expected values.

4.2 Re-usability

These tests basically evaluated 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 issue.

Test Case Derivation: N/A

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

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

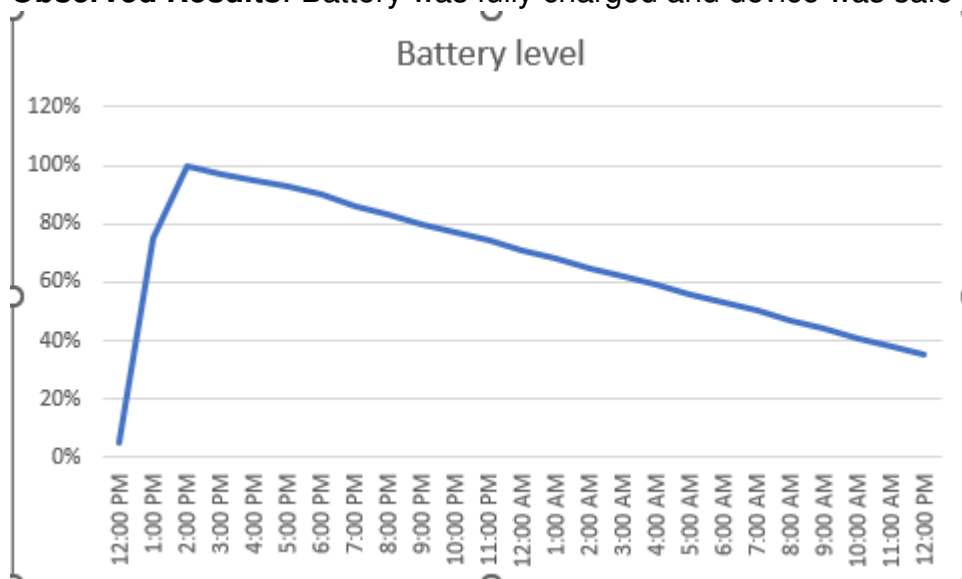


Figure:The battery levels in a span of 24 hours

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, then it wasn't unplugged and left around for couple minutes .

Observed Results: Battery was not being overcharged nor did it heat up, thus device was safe to use.

4.3 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](#)

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

- Are you a dog person or a cat person?
- Are you currently inside of a building?

- Are you currently a student?
- Did you make your bed this morning?
- Do you feel tired right now?
- Do you prefer coffee or tea?
- Do you have a G driver's license?
- Do you feel sleepy right now?
- Which is better: Summer vs Winter
- Is it daytime right now?

Observed Results: The UI was very intuitive and user-friendly. Volunteers were able to navigate through and answer prompts with ease.

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.

4.3.2 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 separate data transfer tests 3 times and recorded the total transfer time for each execution. All 6 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 |
|-----------|---------------|---------------------|-----------------|-----------------------------------|
| 2kB | 2.66s | None | Pass | None |
| 10kB | 13.27s | None | Pass | None |
| 8.5kB | 10.51 | None | Pass | None |
| 5kB | 6.94s | None | Pass | None |
| 11kB | 14.10s | None | Pass | None |
| 18kB | N/A | COM Port 7 Not Open | Result Rejected | Cable was plugged in incorrectly. |
| 20kB | 25.68s | None | Pass | None |
| 35kB | 36.85s | None | Pass | None |

Table 1: 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? If so, how much percentage?".

Observed Results: The group members answered the prompt, thus giving us the time limit on the batteries. It was greater than the parameter set.

4.3.3 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 volunteer tries to access the database and modify the data within.

Expected Results: Database inaccessible and requests an administrative security key. The volunteer should be unable to access or modify the data.

Test Case Derivation: N/A

How test was performed: Device was given to a volunteer with required technical skills who did not have the administrator security key. They attempted to access the database.

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

5 Comparison to Existing Implementation

Current motion detecting smart watches, such as Apple watches or Samsung Galaxy watches, 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.

6 Unit Testing

| | | | | |
|---|---------------------|--------------------------------------|-----|---------------------------------|
| 1 | Void bed_HR_setup() | Set-up function for heartrate module | RQ | User heart rate input from A0 s |
| 2 | name | desc | ref | input |
| 3 | name | desc | ref | input |

7 Changes Due to Testing

After testing out each unit of the device system and failing unit test #2, the type of heartrate sensor has been changed. Now the ... (list out all the potential candidates that were ruled out).

Heart Rate Sensor Options:

Table 2: Tested Heart Rate Sensors Part 1



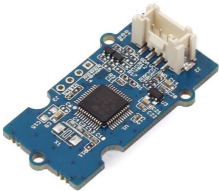


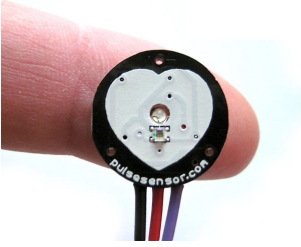
| Image | Model | Status | Reason for Status |
|---|--|----------|--|
|  | Chaney Electronics Inc. G21394 | Rejected | Due to the Infrared LED's Placement and the photoelectric sensor's placement, it is impossible to integrate the module with a device sitting on the wrist. |
|  | SparkFun Pulse Oximeter 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. |
|  | Grove - Finger-clip Heart Rate Sensor | Rejected | Size too large, and issues with skin contact to the sensor results in garbage data. |
|  | Grove - Ear-clip/Finger-clip Heart Rate Sensor | Rejected | Sensor attachment to the ear was considered too disruptive to the participant's daily activities. |
|  | Comimark 2Pcs Heart Rate Pulse Sensor Module for Arduino Raspberry pi | Rejected | Build quality was unacceptably bad. Ordered name-brand version of this product next. |

Table 3: Tested Heart Rate Sensors Part 2

| Image | Model | Status | Reason for Status |
|---|---|----------|---|
|  | 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. |

Last Minute Change to Microcontroller

A major change to the design was a last minute switch to the ESP32 microcontroller after originally working with the Seeeduno 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 Seeeduno Xiao and PCB prior to the Revision 0 demonstration.

The Seeeduno 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 Seeeduno SAMD21 Cortex M0+ chip has a clock speed of 48 MHz which is 4x faster than the clock speed on a typical Arduino Nano of 16 MHz. Additionally, the Seeeduno 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 Seeeduno did not present any issues and development was progressing well. However, leading up to the Revision 0 demo multiple Seeedunos 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 Seeeduno 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.

8 Automated Testing

N/A

9 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 4: Requirements Traceability Matrix Pt1

| | 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 5: Requirements Traceability Matrix Pt2

| | HST1 | HST2 | HST3 | HST4 | RT1 | UT1 | UT2 | UT3 | PT1 | PT2 | DSQT1 |
|-------|------|------|------|------|-----|-----|-----|-----|-----|-----|-------|
| NFR2 | | | | | | X | X | X | | | |
| NFR3 | | | | | | | | | | X | |
| NFR4 | | | | | | | X | | | | |
| NFR5 | | | | | | | | X | | | |
| NFR6 | X | X | X | X | | | | | | | |
| NFR7 | | | | | X | | | | | | |
| NFR8 | | | | | | | | | | | X |
| NFR9 | | | | | | X | | | | | |
| NFR10 | | | | | | | X | | | | |
| NFR11 | X | X | X | X | X | | | | | | |
| NFR13 | | | | | | | | | X | | |
| NFR14 | | | | | | | | | | | X |

Table 6: Requirements Traceability Matrix Pt3

10 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 7: Module Traceability Matrix Pt1

| | 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 8: Module Traceability Matrix Pt2

| | HST1 | HST2 | HST3 | HST4 | RT1 | RT2 | UT1 | UT2 | UT3 | PT1 | PT2 | DSQT1 |
|------------|------|------|------|------|-----|-----|-----|-----|-----|-----|-----|-------|
| M1 | X | X | X | X | X | X | | | | | X | |
| M2 | | | | | | | | | | X | | X |
| M3 | | | | | | | | | | | | |
| M4 | | | | | | | | | | | | |
| M5 | | | | X | | | X | | | | | |
| M6 | | | | | | | X | | | | | |
| M7 | | | | X | | | | | | | | |
| M8 | | | | | | | | | | | | |
| M9 | | | | | | | | | | | | |
| M10 | | | | | | | X | X | | | | |

Table 9: Module Traceability Matrix Pt3

11 Code Coverage Metrics

Appendix — Reflection

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

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