Deliverables (5 parts)

- Use cases
    - Create profiles – Matt
    - Update profiles - Matt
    - Delete profiles - Matt
    - Data collection, processing, and visualization - Rachel
    - Viewing recommendations - Jaedong
    - Storing data/data history - Evan
    - Low battery - Evan
        - Graceful shutdown

- Design documentation – structure and behavior
    - UML Class diagram - Matt
    - UML Sequence diagrams for scenarios covering normal and safety operation, to be determined
    - UML State machine diagrams
        For any control entities
    - Textual explanation of your design decisions

- Implementation
    - Source code of your Qt C++ project that builds and runs on the course VM (COMP3004-F24.ova found at https://carleton.ca/scs/tech-support/virtual-machines/
    - Execution of tests based on the above scenarios

Video: record a video of running your simulation through the above specified scenarios
Requirements traceability matrix

Objectives
Analyze requirements based on provided material and develop use case(s).
Develop a software architecture that supports data collection, processing, and visualization.
Test your implementation.
Relate the modeling, design, implementation and testing through a Requirements Traceability Matrix.

Project Requirements

1. Functional Requirements:
    - User Management: Create, update, and delete user profiles. Each device should support multiple profiles (up to five).
    - Data Collection: Interface with RaDoTech hardware to collect health data.
    - Data Processing: Process raw data using algorithms based on Ryodoraku technology to generate health metrics.
    - Data Visualization: Display health metrics in an easy-to-understand, visual format within the application.
    - Recommendations: Provide a place-holder for specialists' recommendation.
    - Historical Data: Store and allow users to access historical health data for trend analysis.
    - Battery power: charge depletion, low power indication, and graceful shutdown.
2. Non-Functional Requirements:

    Scalability: Ensure the system can handle a growing number of users and data points.

    Usability: Ensure the application is intuitive and user-friendly.

Use Cases:

Use Case: Create Profile

Primary Actor: User

Scope:

Level:

Stakeholders and Interests:

- User: Wants to create a profile

Preconditions:

- The system is powered on
- The user presses the “Create Profile” button

Success guarantee:

- A profile is created and saved in the system.

Main Success Scenario:

1. The user selects “Create Profile”
2. The system prompts the user to enter the required information. (Full name, Age, Gender, Height, Weight,)
3. The user inputs the required information into the designated fields.
4. The user selects “Submit” to finalize the profile creation.
5. The system validates the input.
6. If validation passes, the system creates the user profile and displays a confirmation message.

Extensions:

5. a. The system fails to validate the input:
    1. The system highlights the relevant input fields indicating the issue.
    2. The user corrects the error.
    3. Resume at step 4.

Use Case: Update Profile

Primary Actor: User

Scope:

Level:

Stakeholders and interests:

- User: Wants to update their information.

Preconditions:

- The system is powered on.
- The user already has a profile created.
- The user presses the “Update Information” button.

Success guarantee:

- The profile’s information is updated and saved in the system.

Main Success Scenario:

1. The user selects “Profile”.
2. The system displays the current user profile information.
3. The user selects “Edit Information”.
4. The system enables the editing of the user’s information.
5. The user makes the desired changes to the profile information.
6. The user selects “Save Changes”.
7. The system validates the input.
8. If validation passes, the system updates the user profile and displays a confirmation message.

Extensions:

7. a. The system fails to validate the input:
    1. The system highlights the relevant input fields indicating the issue.
    2. The user corrects the error.
    3. Resume at step 6.

Use Case: Delete Profile

Primary Actor: User

Scope:

Level:

Stakeholders and interests:

- User: Wants to delete a profile.

Preconditions:

- The system is powered on.
- The user presses the “Delete Profile” button.

Success guarantee:

- The profile is permanently deleted from the system.
- All associated data is removed from the system.

Main Success Scenario:

1. The user selects “Profile”.
2. The user selects “Delete Profile”.
3. The system prompts the user with a warning message.
4. The user presses the “Yes, delete my profile permanently” button.
5. The system removes the profile and all associated data.
6. The system displays a confirmation message.
7. The system returns to the login screen.

Extensions:

4. a. The user presses the “No, don’t delete my profile” button:
    1. The system displays a confirmation message.
    2. The system returns to the “Profile” screen.

Use Case: Collecting Data

Primary Actor(s): User

Stakeholders and interests:

- User: wants to collect data from 24 measurement points.

Pre-condition(s):

- User profile exists and is selected for the session.
- The device is powered on and ready for the scanning.

Success guarantee(s) :

- Data for all 24 points are collected and stored in the system.
- Data is ready for processing

Main success scenario:

1. User starts a new health scanning session.
2. System prompts the user to position the device at the measurement point.
3. System verifies skin contact at the specified point.
4. System collects the data for the measurement point.
5. User lifts the device off the skin.
6. User positions the device at the next measurement point.
7. Repeat step 2-6 until all 24 measurement points are scanned.
8. User can enter any additional information.
9. System stores the collected data in the user's profile.

Extensions:

3a. User does not lift the device off the skin :

1. System waits for the user to lift the device off the skin and provides a notification.

4a. Skin contact is not detected:

1. System waits until proper skin contact is established and provides a notification.

Use Case: Processing Data

Primary Actor(s): System

Stakeholders and interests:

- User: wants accurate health metrics derived from collected data.

Pre-condition(s):

- Data for all 24 measurement points has been successfully collected and stored.

Success guarantee(s) :

- Raw data is accurately processed into health metrics
- Procssed data is securely stored and ready for visualization.

Main success scenario:

1. System retrieves all 24 measurement points data.
2. System compares the value to the predefined thresholds for normal,below norm and above norm ranges.
3. System classifies each data point into one of these categories.
3. System stores the processed data and classifications in the user's profile.

Use Case: Visualizing Reading Data

Primary Actor(s): User

Stakeholders and interests:

- User: wants to see their health from understandable visual representation

Pre-condition(s):

- Data must be collected and stored.

Success guarantee(s) :

- The user is able to view historical readings through a visually-pleasing chart.

Main success scenario:

1. User clicks on the Charts tab.
2. The system generates and displays a detailed graph reflecting the selected area of interest.

Extensions:

2a. There is not enough data to fill the chart:

1. Visually distinct empty bars are added for padding, wth a “No Data” tooltip

2b. There is too much data to fill the chart:

1. Surplus data points are not shown, and the chart maintains a 20-session rolling data window.

Use Case: Displaying and Updating Past Notes

Primary Actor(s): User

Stakeholders and interests:

- User: wants to view and edit their notes from past sessions

Pre-condition(s):

- None

Success guarantee(s) :

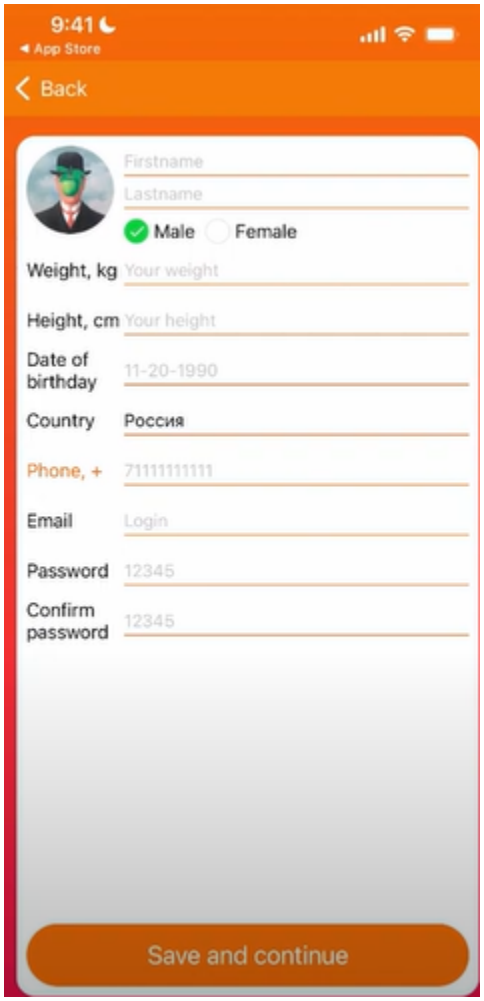
- The user is able to view and modify any note they have previously saved.

Main success scenario:

1. User clicks on the Notes tab.
2. The system displays the latest note, with editable fields.
3. The user uses the arrow buttons to browse through to a note of interest
4. Should they wish, user edits any or all of the notes fields
5. Should they wish, the user presses the save button to update the note

Design documentation:

UML Class Diagram



User:

- userID : int
- fName : string
- lName : string
- age : int
- sex : SEX (Enum)
- height : float
- weight : float

+ createProfile()
+ updateProfile()
+ deleteProfile()
+ Get functions?

Scan:

- scanID : int
- userID : int
- timestamp : Timestamp????
- bodyPoint[] : BodyPoint
- measurements[] : Measurement
- healthInsight : HealthInsight

Measurement:

- measurementID : int
- bodyPointID : int
- value :
- unit :

BodyPoint:

- bodyPointID : int
- name : string
- location : LOCATION (Enum)

Device: (MainWindow)

- batteryLevel : int
- users[NUM_USERS] : User

MainWindow:

- ranges : QMap<QString,QPair<int,int>>
- lastState : bool
- scanCheckboxes : QList<QCheckBox*>
- tagButtonGroup : QList<QPushButton*>
- spotValues: QMap<QString,int>

+ addData(Profile*)
+ processRyodorakuData(Profile*)
+ saveNotes(Profile*)
+ display_note(Profile*)
+ update_chart(Profile*)
- handleCheckboxToggled(bool)
- trackScanning()
- checkAllScansCompleted()
- calculateAverage() : int
- PrintDia()

Battery ✅:

- charging_ports : QVector<QCheckBox*>
- charging_indicators : QVector<QProgressBar*>
- charge_indicator_palette : QPalette

- in_use : bool
- plugged_in : bool
- charge_level : double
+ turn_on_or_off(bool)
+ battery_sim()
+ add_battery_UI(QCheckBox*, QProgressBar*)
+ update_battery_UIs()
+ plug_change(bool)
+ die()

ReadingStorage ✅:
- readings : QMap<QString,int>
- note : Note*
- body_parts_info : QMap<QString,QPair<int,int>>*
+ log_data_point(QString, int)
+ retrieve_data_point(QString) : int
+ retrieve_data_point_percent(QString) : int
+ retrieve_data_point_range(QString) : QPair<int,int>
+ retrieve_session_average() : int

Note ✅:
- bodyTemp : double
- tempUnit : TEMP (Enum)
- bloodPressureLeftSystolic : int
- bloodPressureLeftDiastolic : int
- bloodPressureRightSystolic : int
- bloodPressureRightDiastolic : int
- heartRate : int
- sleepHrs : int
- sleepMins : int
- weight : float
- weightUnit : MASS (Enum)
- emotionalState : MOOD (Enum)
- overallFeeling : MOOD (Enum)
- tags : QStringList
- notes : QString

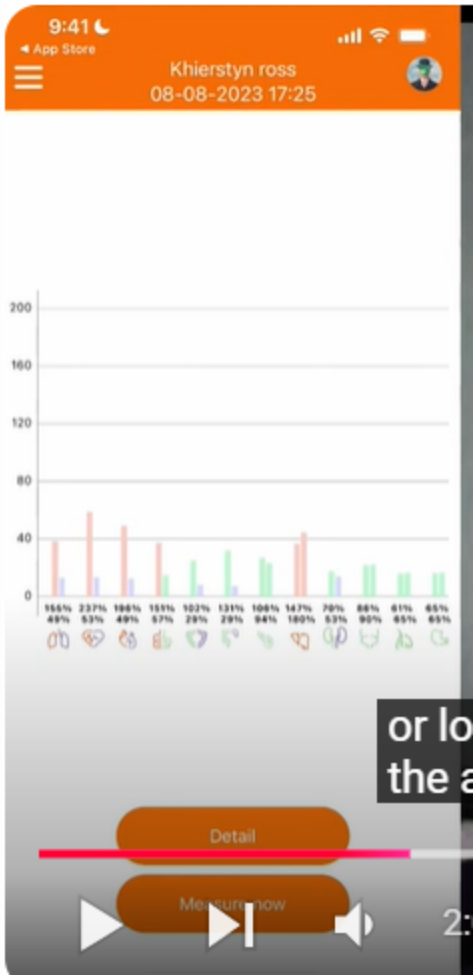
HistoryViewer ✅:
- Num_bars : int
- Chart_bars : QVector<QProgressBar*>
- note_previous_button QPushButton*
- note_next_button QPushButton*
- note_counter : QLabel*
- chart_selector : QComboBox*
- chart_avg : QLabel*
- chart_max : QLabel*
- chart_min : QLabel*
- note_index : int
+ update_chart(QVector<ReadingStorage*>&)

+ next_note()
+ previous_note()

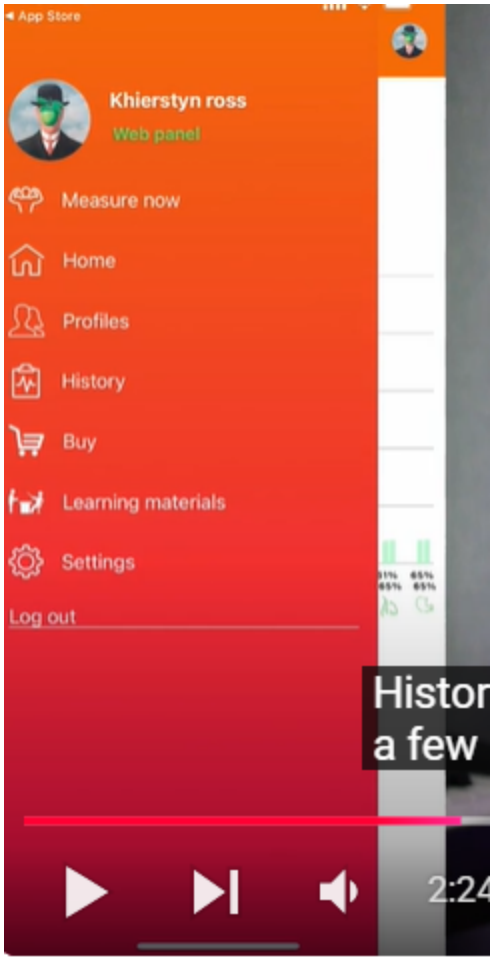
Recommendation
- part_average : int
- higher : int
- lower : int
- wrong_part : QQueue<QString>
- result : QQueue<int>
+ AddAbnormalPartinQ(QString left, QString right, QMap<QString,int>& data)
+ GetAverage(QMap<QString,int>& data) : int
+ GetWrong_partSize() : int
+ GetWrong_part() : QString
+ Getresult() : int
+ Reorganize(Qstring left, QString right) : QString

Radotech

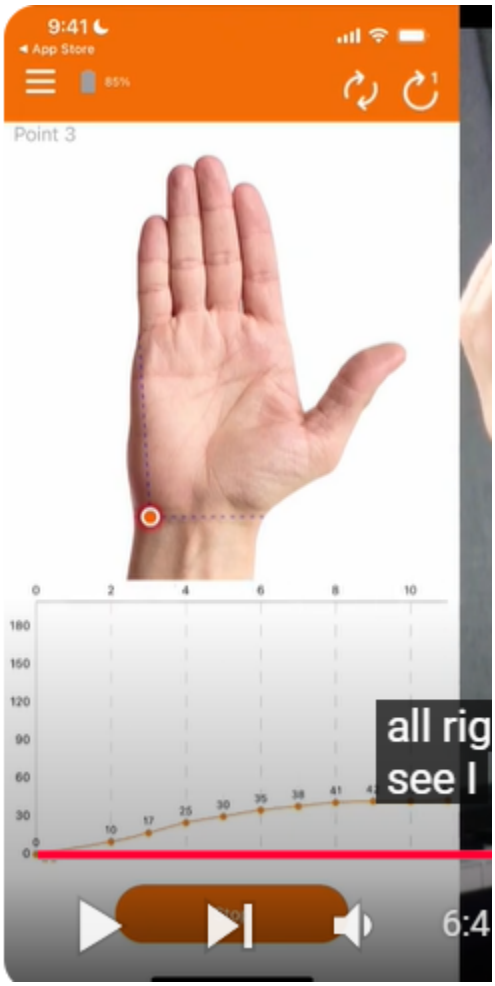
1. health tracker, little spray bottle with tap water, charging cable
2. download app -radotech-
3. log in or create an account
4. first, last name, male or female, weight (KG), height(cm), date of birth, country, phone number, email, password, and confirm password in the create account section.

5. 

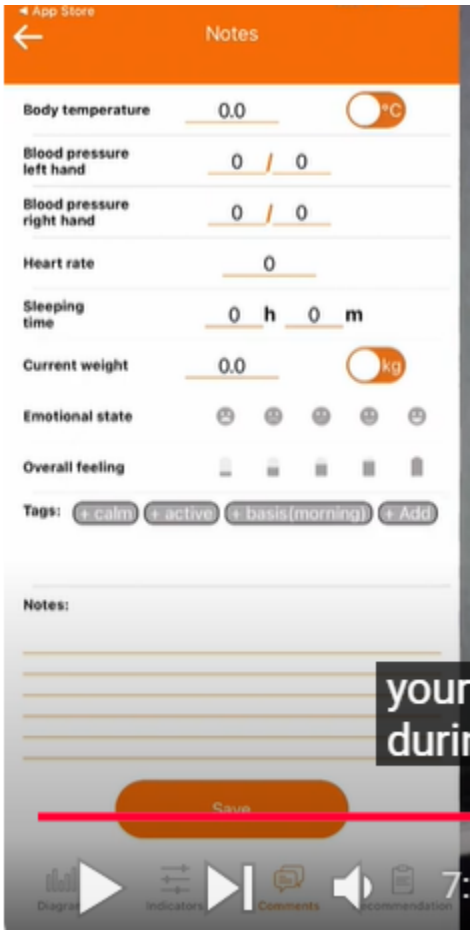
6. Once you create or log in to the account



7.
8. in the menu, measure now, home, profiles, history, buy, learning materials, settings, log out
9. in history, once the user takes a few scans over and over, the user will be able to see a history of the user's past scans and see what trends are starting to form over time with multiple scans.
10. In learning materials, the user starts to get more scans and take more snapshots, to help the user understand what some of the readings mean, and how to take action on making improvements or optimizing the user's current state. And then if the user wants to check out or supplement the library, the user can head over to the official website.
11. first scan
12. first measurement will prompt the user to connect via Bluetooth, the user will be able to connect a machine in the app before making the user's first scan.
13. Android devices, need location services in the phone settings to use the radio tech device and take a proper measurement
14. the node at the top the of device will be applied to certain meridians on the user's body. the app walks the user through a step by step, it is an electrical node, user needs to set the test up for success
15. take off any metal jewelry for success (earrings, neckless, rings, watches, whatever)

16. to start measuring select Measure now
17. and give some practice tips
18. do not hold it like a pencil
19. holds it overhand, the user's palm is over the metal of the machine.
20. it doesn't matter what way you touch your skin, do not need to hold it at the machine to a specific angle
21. moisturizing where the user will scan. (Using tap water is fine)
22. finish moisturizing -> click close
23. at first time, it will ask you to pair the radotech device with your phone.
24. at the bottom of the machine, power button -> click that for a few seconds
25. power flashing blue (ready to pair)
26. start
27. choose profile
28. start scanning
29. show where you should scan in the app, once the scanning is done, a ring sound
30. 

31. also while the user scanning the target place, a graph should start drawing
32. as soon as it beeps, the user should pull it off from the user's skin immediately to not get a misread
33. make sure the surface of the node is touching the skin.

34. top pulls out giving a bit more length for the feet when the user is scanning
35. scan is done
36. user has a chance to take some notes on the user's state, and how you were feeling during the scan
37. 

38. Save -> success message.
39. head back -> see history
40. day-month-year time format
41. first screen -> overall body shot with showing which organs are above normal(red), normal(green), below normal(blue)
42. click chart -> what it means at the organ level
43. for more in-depth information with practical and actionable tips
44. go to indicator gives a snapshot of where how the user’s systems overall are functioning from energy level, immune system, metabolism, user’s psycho-emotional state
45. if want to see what means -> click on it -> display what it means
46. recommendations
47. in recommendation -> give detailed insight for the snapshot
48. any follow-up questions for how to best read the data for yourself or supplement recommendations -> email radotech.com

Design Choices Explained

Data Processing Explanation:

After conducting research, we realized there are two approaches for generating health metrics from raw data collected:

1. Fixed Threshold Method:
    a. If the measured value is below 40μA, it shows the measurement is below norm.
    b. If the measured value is above 80μA, it shows the measurement is above norm.
    c. Otherwise, if it is within the range of 40μA to 80μA, it shows the measurement is normal.
2. Dynamic Range Method:
    a. The average Ryodoraku is calculated by summing all 24 measurement and dividing by 24.
    b. If the measured valued is $average + 20\%$, the measurement is above norm.
    c. If the measured valued is $average + 20\%$, the measurement is below norm
    d. Otherwise, the measurement is normal.

Source:https://pmc.ncbi.nlm.nih.gov/articles/PMC6831389/#:~:text=First%2C%20less%20than%2040%20%CE%BCA,an%20individual's%20average%20Ryodoraku%20current.

Combining our research with the Ryodoraku Chart from the specification, we determined that the second approach aligns with RaDoTech's data processing algorithm. Therefore, we chose to use the dynamic range method.

Graceful Shutdown Explanation:

A main goal of graceful shutdown is that no data gets partially or incompletely saved. To ensure this, all data gets saved in batches rather than as it is read. For example, all electrical readings are saved into a temporary array. Only when all readings are completed are they saved to the user's profile all at once. This prevents power failures from causing partial logs, which would lead to data corruption and disrupt any future trend analysis or history viewing.