




COMP - 3004

NEURESET

Adam Esanhaji
Anas Ismaili
Joshua Mukasa
Kunal Barua



Use Cases:

UC-01	Base Use Case
Primary Actors	User, Device, Battery
Pre-condition	<ul style="list-style-type: none"> Device is in proper working order and NOT defective.
Success guarantees	<ul style="list-style-type: none"> Device performs actions in response to user commands
Main Sequence	<ol style="list-style-type: none"> Device is powered on Device shows the main menu and provides the user a method of selecting options Users selects an option The power button is pressed, and the device turns off
Extensions	<p>*a. If battery charge is low</p> <p>*a1. Device warns the user of low battery, and must be charged before proceeding</p>

UC-02	Treatment Session
Primary Actors	User, Device, EEG Headset
Pre-condition	<ul style="list-style-type: none"> Device is turned on and working properly
Success guarantees	<ul style="list-style-type: none"> Device performs treatment session properly Treatment details are saved in a session log
Main Sequence	<ol style="list-style-type: none"> User selects option to start new session for treatment User puts on the headset Once contact is initiated, indicated by a blue light, the timer shows the approximate time remaining, and the session progress bar shows a completion percentage Device calculates the “before” baseline frequencies for each EEG site concurrently Device performs treatment round using an offset frequency of 5Hz Device performs treatment round using an offset frequency of 10Hz Device performs treatment round using an offset frequency of 15Hz Device performs treatment round using an offset frequency of 20Hz Device calculates the “after” baseline frequencies for each EEG site concurrently Once all treatment rounds are complete, “before” and “after” baselines for the EEG sites are recorded and saved to a session log
Extensions	<p>*a. Poor/Inadequate contact of the EEG electrodes</p> <p>*a1. The red light on the device flashes and the session is paused and the device starts beeping until contact is re-established.</p> <p>*a2. If the contact is adequate, the device resumes the session.</p> <p>*b. Users wishes to pause the current session and presses the pause button. If the session is not resumed within 5 minutes, the device automatically turns off without saving the session.</p> <p>*c. User wishes to stop the current session and presses the stop button and takes the user to the main menu.</p>

UC-03	Treatment Round
Primary Actors	User, Device, EEG Headset
Pre-condition	<ul style="list-style-type: none"> Device is turned on and working properly Device is currently performing a treatment session
Success guarantees	<ul style="list-style-type: none"> All 21 EEG sites have successfully undergone treatment Treatment round for specified frequency is finished
Main Sequence	<ol style="list-style-type: none"> Device targets the 21 EEG sites for the treatment concurrently The device's green light is flashing during the entire treatment round Device reads signal from each EEG site and establishes baseline average frequencies over period of one minute It adds the offset frequency to the baseline frequencies every 1/16th of a second It then recalculates the brainwave frequencies, adds the offset and repeats the process every 1/16th of a second for a duration of 1 second
Extensions	<p>*a. Poor/Inadequate contact of the EEG electrodes</p> <p>*a1. The red light on the device flashes and the session is paused and the device starts beeping until contact is re-established.</p> <p>*a2. If the contact is adequate, the device resumes the session.</p> <p>*b. Users wishes to pause the current session and presses the pause button. If the session is not resumed within 5 minutes, the device automatically turns off without saving the session.</p> <p>*c. User wishes to stop the current session and presses the stop button and takes the user to the main menu.</p>

UC-04	Session Logs Viewing
Primary Actors	User, Device
Pre-condition	<ul style="list-style-type: none"> Device is turned on and working properly User is at the main menu
Success guarantees	<ul style="list-style-type: none"> Scrollable session log containing time and date is displayed
Main Sequence	<ol style="list-style-type: none"> User selects option to view session log Date and Time of logged sessions are displayed on the device
Extensions	<p>*a. If menu button is pressed</p> <p>*a1. user is returned to the main menu</p> <p>2a. If no sessions are logged</p> <p>2a1. message is displayed informing user</p>

UC-05	Update Time and Date
Primary Actors	User, Device
Pre-condition	<ul style="list-style-type: none"> Device is turned on and working properly User is at the main menu
Success guarantees	<ul style="list-style-type: none"> Time and date on device are updated
Main Sequence	<ol style="list-style-type: none"> User selects the "Date and Time" option from the main menu User inputs a date and time Date and time on the device are updated
Extensions	<p>2a. If date or time is invalid</p> <p>2a1. User is notified date or time was not accepted</p> <p>2a2. Date and time are not updated</p> <p>2a3. User is returned to the main menu</p>

UC-06	PC Upload and Viewing
Primary Actors	User, Device, PC
Pre-condition	<ul style="list-style-type: none"> • User has a PC • Device and PC are both turned on and functional • PC and device are connected to each other
Success guarantees	<ul style="list-style-type: none"> • Before and after baselines are uploaded to the PC, along with time and date information
Main Sequence	<ol style="list-style-type: none"> 1. User selects menu option to upload data to PC 2. Device begins the data transfer process 3. PC receives the data from the device 4. PC displays all the session log data
Extensions	*a. Technical issues during transfer *a1. Device displays error message on the Neureset display

Design explanations:

Our project uses a couple of different design patterns. Firstly, we used the state pattern in our implementation of the Neureset, as many of the different functionalities depended on the state of the Neureset. This was needed so that the user would not be able to use certain buttons/activate certain features when the device could not handle it. For example, whenever the Neureset was in an “off” state (powered off), the buttons in the interface would all be disabled, and the main screen of the device would be off. Also, when the Neureset was in the “treatment” state, the button to return to menu and play button would be disabled and the main display would show a timer and progress bar, rather than the main menu.

The observer pattern was used through our codebase by utilizing Qt’s signals and slots functionality. It is mainly used for connecting our UI widgets to their respective button handlers and for inter-thread communication. In our design, signals acts as publishers and slots acts as subscribers that listen to these signals by using the connect function. This allows us to easily create handlers to catch UI events and perform appropriate functions in return. We also created our own signals and slots for inter-thread communication to allow our therapy and timer threads to interact with the main QT UI thread.

The strategy pattern was used for the generation of accurate waveforms. This pattern allowed us to implement modular and easily switchable algorithms for generating waveforms based on the different frequency bands. The generation strategy is controlled by the Neureset which specifies which algorithm the probes are to currently use.

The mediator pattern was used to simplify interactions between the many components of our codebase. Our Neureset acts as a central mediator between the MainWindow UI, SessionManager, and PC. This pattern allows us to have separation between the components and simplifies the modifying of state because of the centralized nature of the Neureset.

The singleton pattern was used for the SessionManager, both the PC and Neureset class share the same instance of SessionManager. This was necessary because conflicts could arise with more than one instance, having more than one instance means having more than one buffer. In this case the data would not be unified, the data of one SessionManager may not be the same as the data of another SessionManager which in turn could lead to data loss when the separate instances save their buffer to disk.

The PC is designed to handle user inputs through its graphical interface. Users can start new sessions, select existing sessions to view detailed data, and modify settings or parameters of neurofeedback sessions. When a user selects a session from the list, the `onSessionSelected()` slot is triggered, which updates other parts of the UI, such as detail labels or graphs, to reflect the selected session's data. The PC dynamically displays detailed session information and EEG data visualization using detailed labels and custom widgets like `SineWave`. EEG data for the selected session is visualized using the `SineWave` widget, which plots frequency and amplitude over time based on the session's EEG readings.

In the `SineWave` constructor, member variables `frequency1` and `frequency2` are initialized that stores before and after frequencies which is used to define the waveform characteristics. Once the `SineWave` widget is constructed and added to the UI, Qt's event loop will eventually send a `paintEvent` to the widget. This happens automatically when the widget is first shown. When the `paintEvent()` is triggered, the `QPainter` object is created to perform drawing operations within the `SineWave` widget. The `paintEvent()` uses the initialized data (like `frequency1` and `frequency2`) to calculate the points needed to draw the waveform. The `paintEvent()` then creates a `QPainterPath` or uses `drawLine()` functions of the `QPainter` to draw the sine wave on the widget's canvas. It may use loops, mathematical functions like `sin()`, and scale factors to translate the waveform data into pixel coordinates for the drawing. After the drawing commands are issued in the `paintEvent()`, the rendered image is then displayed on the screen.

Tracibility Matrix:

ID	Requirement	Related Use Cases	Implemented By	Tested By
01	The device has a headset with 21 electrodes.	Fulfilled by design	Fulfilled by design	N/A
02	The device has a red light for contact failure, green light for session in progress and blue light for successful contact	Fulfilled by design	<code>MainWindow.blueLight</code> <code>MainWindow.greenLight</code> <code>MainWindow.redLight</code>	UI contains lights that get activated as needed during runtime
03	The user can initiate a session from the menu.	UC-01	<code>Neureset.menuListHandler()</code> <code>Neureset.startButtonHandler()</code> <code>Neureset.startTherapy()</code>	User can select the "New Session" option from the device menu
04	The device must inform the user about the session duration and completion status.	UC-02	<code>Neureset.startTherapy()</code> <code>MainWindow.updateProgressBar()</code> <code>MainWindow.updateTimerDisp()</code>	The mid-treatment display contains a progress bar and timer
05	The device must be able to read EEG signals from each of the 21 sites on the headset.	UC-03	<code>Probe.measureWaveform()</code> <code>Probe.getWaveform()</code>	Each site has a probe, that the device communicates with using the headset
06	The device must calculate a baseline average frequency for each EEG site over approximately one minute.	UC-02, UC-03	<code>Probe.measureWaveform()</code>	N/A
07	The device must deliver treatment by adding an offset frequency to the baseline frequency every 1/16th of a second.	UC-02, UC-03	<code>Probe.applyFeedBack()</code>	N/A

08	The device must indicate the progress of treatment using a progress bar and the green light	UC-02	MainWindow.updateProgressBar() MainWindow.greenLight	A progress bar on the main window fulfils this requirement
09	The user should be able to pause a session.	UC-02, UC-03	MainWindow.pauseButton Neureset.pauseButtonHandler()	A pause button is provided in the device UI to pause the session
10	If contact with the user is lost, the device must pause the session, flash a red light, start beeping, and turn off automatically if contact is not reestablished within 5 minutes.	UC-02, UC-03	Neureset.connectHeadsetHandler() Neureset.connectionLost() Headset.connected() Headset.disconnect()	A red light is shown in the UI which indicates the user about lost connection
11	The device must save a log of session times and dates.	UC-02	SessionManager.writeToBuffer() SessionManager.saveBufferToDisk()	As treatment progresses, session manager writes to buffer with log details which can be accessed using getter functions
12	Before and after baselines for all 21 EEG sites must be recorded, along with session times and dates	UC-02, UC-03	SessionManager.writeToBuffer() SessionManager.saveBufferToDisk()	session manager writes to buffer with details about before and after frequencies of each site
13	The user must be able to set the current date and time on the device.	UC-05	Neureset.updateDateTime() Neureset.saveDateTime()	A main menu option is provided in the UI called “set date and time” which accomplishes this requirement