*restarting means pressing the power button to turn off, then on again. Not exiting the application.
IMPORTANT: Implemented by AEDController also implies AEDTransmitter, as it is a tightly coupled extension of AEDController.
IMPORTANT: Compression button simulates both compressions and the necessary breaths in between.

ID	Requirement	Use Case	Implemented by	Test	Description
1	Concurrent operation: User will be able to interact with the application with little to no input latency.	None	Design / Architecture	Run the application and observe any lag in interactions	By design, the AEDController separates decisions made by the AED system from the decisions the user makes, as AEDController runs in a different thread than the UI.
2	Dual display: The main window displays that it has power on and shock buttons, a message display, indicator lights, and an LCD displaying the heart rate.	None	AEDWindow.h /cpp/ui	Run the application and observe that the main AED window is showing	Implemented using QT's UI designer and using extracted assets from the AED-Plus, the AEDWindow represents almost the entire system. It has a button that will allow the user to power the system on, a button to administer a shock, and a button to replace a battery at the back of the AED.
3	Dual display: The secondary window displays that it has buttons and a slider that interact with the AED and the patient directly, and a graph showing the ECG coming from the pads.	None	testwindow.h/cpp/ui	Run the application and observe that the test window is showing.	Testwindow represents the extended capabilities of the user to place different types of pads on the patient, to disconnect the pads from the AED, and to give compressions to the patient. It also contains buttons for testing different scenarios and safety features.
4	Power on: The AED will power on and go through its operation sequence when the power button is clicked	UC1	AEDWindow, AEDController, AEDState	Press the power on button and observe changes in the AED's window.	The AED has a self test to go through when it powers up, to make sure the AED can operate safely.
5	Power off: The AED will turn off and save its state during	None	AEDWindow, AEDController,	Press the power off button when the AED is on, and observe that the UI resets to its	The AED will clear the UI and save it's state to be used later on when the AED is powered on again.

	operation.		AEDState	off state.	
6	Saved state: The AED will start at different appropriate points in the program when it is turned off and on again.	None	AEDWindow, AEDController, AEDState, Patient	Multiple tests: see below.	The AED will use the saved state of its patient and its controller's state to make a decision of where to start the program.
7	Same as 6	None	Same as 6	With pads off, restart* the AED and observe that the program starts at the beginning and halts at the pads placement step.	AEDController and AEDState looks at the state of the patient (whether or not the pads are on) and if they're not on, it assumes that the Patient has not started treatment yet, so it starts AEDState at the beginning and asks for pads to be placed.
8	Same as 6	None	Same as 6	With pads properly on, restart* the AED and observe that the program starts at the analysis step.	If the pads are on, it assumes that we are mid-treatment and the AED has lost power for whatever reason. However, it will skip the time-consuming steps and go straight to analysis for expedited treatment.
9	Heart rate and ECG is only displayed when the pads are on the patient .	UC2	AEDWindow, AEDController, AEDState, TestWindow	Observe that the heart rate LCD and the ECG graph are not reading the patient's heart rate when pads are not on.	AED and Test windows' respective heart measuring elements are not updated unless they detect that the Patient class has pads on.
10	Shock Safety: The patient will not be shocked before analysis and/or while the rhythm is unshockable.	None	AEDWindow, AEDController. AEDState, AED, Patient	Click the shock button before analysis and when the rhythm is found to be unshockable. Observe that nothing happens.	AED will not receive a shock signal unless the system is in the shock state, which is determined by AEDController and AEDState, which ultimately calls that function. The AEDController will not progress to analysis unless the patient has pads on.
11	Compression Safety: The patient cannot receive compressions during any step outside of CPR.	None	TestWindow, AEDController. AEDState, Patient	Click the compression button during analysis and shock states. Observe that it has no effect.	Testwindow will not allow the compression signal to propagate if the system is not in the CPR state.

12	Proper Analysis: The AED will correctly classify SVT and VF into shockable rhythms and normal heart rates and PEA/asystole into unshockable rhythms. It will also tell the user when the patient is asystolic.	UC3	AEDWindow, AEDController. AEDState, AED, Patient	Move the heart rate slider to the left and right red regions, and see what the centre console says. Also observe that the state changes to either shock or CPR accordingly. Set the slider to 0 and observe that the AED switches between “Patient is asystolic” and compression feedback.	The AEDController’s state will observe the heart rate and, depending on its value, classify it as one of the two rhythm types. Asystolic patients will be classified differently.
13	Shocks: AED will deliver a shock once a shockable rhythm is found and the button is pressed, stabilising the patient.	UC4	AEDWindow, AEDController, AEDState	Move the heart rate slider to the right, wait for the AED to light up the shock button, and press. Observe the effect on the heart rate.	The shock is delivered by the AED, and how much it stabilises the patient depends on how far away from baseline they are, with extreme values inducing a stronger stabilising effect. It is also affected by a random factor.
14	Compressions/real-time feedback: Program will allow user to perform CPR on the patient, stabilising them when at the proper rhythm. It will also give feedback on compression rhythm.	UC5	AEDState, AEDController, TestWindow, TestController, AEDWindow	Go into the compression stage, then start compressing. Pay attention to the feedback given and the heart rate value depending on if the AED says “Steady Compression Rhythm”, “Compressions too fast…” or “Compressions too slow…”.	The AEDState, through AEDController, only stabilises the patient if the rate of compressions is optimal, and bases the optimal heart rate based on the number of clicks achieved at each point in time. The clicks are taken in from TestWindow and the total clicks are stored in TestController. The user is given about 12 seconds to simulate the standard 120 seconds given to do CPR.
15	Battery depletion: AED’s battery will deplete at each shock.	None	Battery, AED, AEDController, AEDWindow	Get into a shockable state, shock, and observe battery level.	A press of the shock button will travel through AEDWindow to AEDController, which sets a flag in the AED to true which is later checked by AEDState to start the countdown to shock the patient. AEDState actually calls the shock function, which depletes the battery by 10%.
15.1	Battery critical: If battery level is critical (<30) then we must	UC1, UC8	Same as 15	Deplete battery using “Deplete Battery” button until we are below 30 and observe	AEDController does a system check which turns an error flag true if battery is <30, and will in turn set its state to

	replace the battery to continue effective treatment.			that we are in the Power On state. Then, use “Replace Battery” button to replenish above 30%. Observe that we pass the safety test.	the Power On state, which does not exit until the condition (battery > 30) is satisfied.
16	Patient Evaluation: The health outcome of the patient can be determined at any point in the program.	UC6, UC7	TestWindow, Patient, TestController	Press the button when the patient is dead / nominal / shockable / unshockable / rapidly deteriorating.	The evaluation will display the health state of the patient and acts as feedback on treatment. The “Patient is recovering” range is 60 < x < 150, and anything outside is “Patient is in danger”, unless the heart rate is at 0, which is “Patient is dead”. There is a special case where heart rate goes over 250, then they are “Rapidly Deteriorating”.
17	Audio: Program will play audio messages similar to an AED.	None	MediaPlayer, AED	Power on the AED, attempt shocking, to hear the audio play.	The MediaPlayer uses alsa-utils, an audio package on linux and a QProcess to run the package given an audio file stored in the resources folder of this project.
18	Pad Incompatibility: Shocking a child with adult pads gives an increased chance to kill them, while the reverse will barely affect the adult. Only the proper pads will reliably stabilize the patient.	UC2	Patient, AED, AEDController, Testwindow, Testcontroller	Test the effects of a shock using: Adult pads on child, child pads on adult, and adult pads on adult. Observe: the first case may overshoot the healthy zone, the second will produce a weak effect, and the last one will stabilize the patient reliably.	The shock function in AED takes into account the type of patient, the type of pads connected, and decides the range for which the heart rate will stabilize by, depending on these variables.
19	Patient Instability: If the patient is outside of the green zone, then they will get increasingly unstable.	None	AEDController, Testcontroller, Testwindow, Patient	Bring the patient’s heart rate slider outside of the green zone, then observe that it goes towards the extremes.	To simulate a deteriorating / recovering patient, AEDController’s simulation tick will update the heart rate either away or towards the median healthy heart rate depending on the initial condition.
20	User Death: If a shock is administered while the user is in a conductive environment, it will kill them and the patient.	None	AEDController, AED, AEDWindow	Toggle the wet patient status in testwindow, then shock. Observe that the AEDWindow is disabled.	AED will make a check for the conductive environment upon shocking, if it is found to be conductive, then the UI will be disabled and the patient will rapidly deteriorate towards asystole.
21	Patient Stability: If the patient is	Same	Same as 19	Bring the patient’s heart rate slider within	Same as 19

	within the green zone, they will recover to 105.	as 19		the green zone, then observe that it goes towards the median of the green zone.	
22	Audio: AED will play audio for certain messages.	UC2, UC4, UC1	MediaPlayer, AEDWindow, AED.	Power on the AED, shock the patient, power off the AED. Listen to the sound.	Using the package alsa-utils, we create a QProcess from MediaPlayer to run “aplay” to play a sound clip, separate from our application’s process.
23	Pad Placement: Pads may be placed incorrectly and show the ECG, but will not proceed past the step until they are placed correctly.	UC2	Pads, Patient, AEDController, AEDState, Testwindow, AEDWindow, Testcontroller.	Place the pads and get the “CHECK ELECTRODE PADS” message. Observe that the program does not progress until the message “PADS SUCCESSFULLY ATTACHED”. Observe that in either case, heart rate and ECG are updated.	There is a 50% chance to place the pads incorrectly as defined by AEDController. AEDController is constantly checking for the status of the pads, and only updates the LCD and ECG when it is true. However, AEDState never moves onto the next state if it finds that the pads are improperly placed.
24	Pad Connection Safety: AED will not proceed from powering on until the pad is connected to the AED.	UC1, UC9	Testwindow, TestController, AEDWindow, Pads, AEDController	At any point after the AED is on, click the “disconnect pads from AED” button, then observe that the AED goes back to the power on step, and says “PLUG IN ELECTRODE CABLE”, and does not proceed until the button is clicked again.	The Pad class stores a variable to check whether or not the pads are connected to the AED, and can be toggled by the disconnect button. For safety, if this is not connected, then AEDController will detect an error and override the state to the power on state.
25	Arrhythmia distinction: ECG will clearly show what is a normal heart rate, a PEA, SVT, VF and Asystole.	None	HeartRateGenerator, Patient, Testcontroller, Testwindow	When pads are connected: Move the slider to each region of the slider. Observe that each region, including the leftmost slider position has a different signal. When heart rate is lower than 60, a simple sine wave is produced, simulating PEA. When at 0, a flat line is produced, simulating asystole. When above 150, a rapid but regular compound wave is produced, simulating VT.	The Testcontroller has a function updateHeartRate() which updates the ECG with values generated from HeartRateGenerator. HeartRateGenerator will produce different waves depending on the heart rate. Noise is added using a random normal distribution function.

				When above 200, the wave varies in frequency and complexity, simulating VF. When between 60 and 150, a regular compound wave is produced with a moderate frequency.	