Wearable Tech Device for Measuring Electrodermal Activity and Cardiovascular signals in Cognitive learning Environments

(Software Proposal)

Advisor: Dr. Atchison

Team members:

Terrell Woodard

Jacob Zolda

Josh Kuk

December 5, 2025

Executive Summary

The current landscape of physiological research is limited by a tradeoff between usability and transparency. Commercial wearables often lock raw data behind black box algorithms while validated academic tools generally lack the capability for real time analysis. To address this gap, our team proposes a transparent software platform designed to capture and process Electrodermal Activity and Heart Rate Variability signals with research grade precision. This system utilizes a PC-based architecture that connects to a custom wearable device via Bluetooth Low Energy to ensure low latency performance.

Our technical approach leverages the processing power of a standard laptop to implement advanced signal decomposition techniques that are too computationally heavy for standard microcontrollers. We utilize Convex Optimization algorithms to separate skin conductance signals into distinct tonic and phasic components to allow for accurate stress detection. The software stack is built on open-source technologies including Python and Visual Studio Code to ensure the system remains accessible and cost effective. We selected TimescaleDB for our database to handle high velocity sensor data and implemented PyQtGraph to render biological signals without visual lag.

This design prioritizes ethical data management by keeping all sensitive information on the local machine rather than transmitting it to the cloud. This strategy mitigates privacy risks and allows us to secure patient data using AES 256 encryption standards. The project operates on a lean budget of two hundred dollars by maximizing the use of free development tools and academic resources. We adhere to an Agile workflow broken into three distinct phases to ensure the delivery of a validated and documented software package by the spring term.

Table of Contents

Sections

1. Problem Description

1.1.1 Background

Electrodermal activity (EDA) and heart-rate variability (HRV) are widely used to study sympathetic nervous system activity, cognitive workload, and stress. [9] Although these signals contain physiological information, most existing tools for analyzing them rely on proprietary, black-box software. These systems hide how filtering, decomposition, and artifact suppression are performed, preventing researchers from validating results.

As described in Posada-Quintero & Chon's review of recent techniques, modern EDA research requires transparent access to raw data, customizable processing, and reliable tonic/phasic decomposition. [9] However, widely used platforms such as commercial wearables offer only partial solutions. Academic tools such as Ledalab provide advanced analysis but lack real-time capability, while commercial devices like Fitbit or Garmin hide raw data entirely and with no reproducibility. [4]

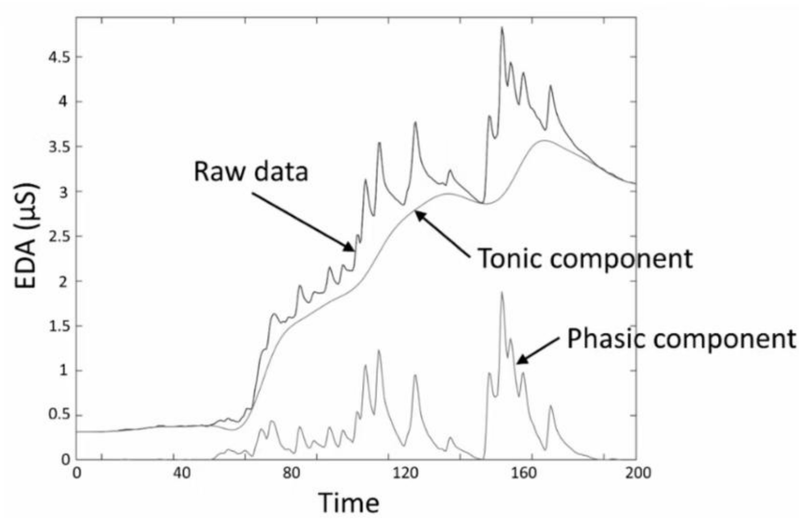


Figure 1. EDA data decomposition into tonic and phasic components. Reproduced from Posada-Quintero & Chon (2020)

The figure above illustrates EDA signal and its components, slow-varying tonic components and rapid phasic responses. Each reflects different physiological processes, and accurate interpretation depends on transparent processing. Without access to both raw and processed components, researchers cannot trust or replicate captured arousal metrics.

Given the growing interest in cognitive modeling, emotional regulation studies, and mental-health analytics, there is a clear need for a research-grade, transparent, and accessible software solution for processing EDA and HRV signals.

1.1.2 Stakeholders

The development of EDA and HRV analytics software requires careful consideration of multiple stakeholder groups. Through a stakeholder interview with Dr. Evangelia Chrysikou, who studies cognitive and affective regulation using multimodal signals, we identified the following needs:

Stakeholder	Need	Standards
Research Laboratories	• Raw and clean data access • Adjustable filter settings from environmental noise • Long, continuous recording • Quick hardware & software setup	• Export both raw and filtered data, validated against cvxEDA benchmarks (>90% similarity) • Full documentation of API • Real-time processing capability for EDA and HRV sampling • Setup time less than 10 minutes
Clinical Therapists	• Minimal false readings • Readable visualizations of data	• Artifact suppression for EDA activity vs noise, checked against a validated dataset (>85% accuracy) • Visualizations follow human-system interaction guidelines ISO 9241-12
Study Participants	• Data privacy • Clear consent procedure • Minimal stress from setup	• Data Encryption Standard AES-256 • Follow IRB standards and explain risks/benefits • Human-system interaction ISO 9241-210
Society	• Accessibility • Ethical algorithms • Progress of mental-health research	• Ability for software to run on low-power hardware (Drexel PC requirements) • No hidden steps, algorithm transparency report, validation on diverse datasets • Use of standardized methods (CSV or EDF) for replicability

Figure 2. Stakeholder Needs and Standards Table

Together, these needs define the guiding principles of the proposed software: transparency, accuracy, accessibility, and ethical responsibility.

1.2 State of the Art

Many software tools and commercial products exist for processing EDA and HRV signals; however, each provides only a partial solution relative to the needs of research-grade analysis. Methods such as Ledalab are widely used in academic studies for tonic and phasic decomposition and spectral analysis. [4] These tools are validated and offer strong analytic capabilities but are designed primarily for offline processing and are not optimized for real-time or wearable applications. Their complexity and lack of integrated HRV analysis limit their usability for continuous monitoring.

Model-based tools such as SCRalyze offer advanced decomposition techniques suited for carefully controlled experimental designs but introduce steep learning curves and are not intended for wearable systems. Kubios HRV provides industry-standard HRV metrics and visualizations but includes minimal EDA functionality and is fully proprietary.

On the commercial side, devices such as Fitbit and Garmin provide user-friendly hardware and consistent heart-rate tracking. However, their signal processing pipelines are entirely under-the-hood, offering no access to raw EDA data and no transparency regarding filtering, decomposition, or noise. As discussed in Posada-Quintero & Chon's systematic review, the lack of standardized processing and the prevalence of hidden algorithms create major barriers for reproducibility and scientific reliability in EDA research. [9]

Across all existing solutions, there is no unified tool that provides:

- Open, transparent processing for both EDA and HRV
- Real-time capability suitable for wearable systems
- Integrated signal quality assessment
- Reproducible signal decomposition
- Research-grade outputs accessible to non-experts

This gap highlights the need for a platform that combines the consistency of academic tools with the usability and accessibility of modern wearables, plus full transparency in how the biometric signals are processed.

1.3 Problem Statement

There is currently no accessible, open, and transparent software pipeline that enables users to reliably process EDA and HRV signals from wearable hardware while providing full visibility into:

- Signal preprocessing including filter design and artifact handling
- Tonic and phasic component decomposition
- Frequency-domain sympathetic metrics
- Data quality assessment and reproducibility

Researchers, clinicians, and students require a system that provides interpretable outputs, avoids black-box algorithms, and supports reproducible experiments.

Problem Statement:
Current wearable devices lack transparent and reproducible software tools for processing EDA and HRV data. This prevents users from validating signal decomposition, assessing sympathetic activity, and reliably interpreting biometric responses.

2. Context and Impact

2.1 Economic Analysis

From an economic standpoint, the system's use of open-source tools significantly reduces financial barriers to the research. Many existing platforms require costly software licenses, dedicated hardware, or subscription-based cloud services. Our design reverses this trend by offering a freely accessible, modular processing pipeline that researchers can adopt, extend, and repurpose across future projects.

Economic benefits include:

- Long-term cost reduction through open-source tools and transparent methods
- Reusability of components through modular architecture
- Low barrier to entry, excludes specialized infrastructure
- Scalability, supports lightweight setups or advanced desktop analysis

This economic model supports equitable access to biometric research technologies and encourages collaborative scientific development.

2.2 Environmental Analysis

The proposed system emphasizes local processing of EDA and HRV signals, resulting in lower energy consumption compared to cloud-dependent architectures. By minimizing data transmission and eliminating external servers, the software reduces power use and carbon emissions. [7] This aligns with sustainable computing practices.

Key environmental advantages include:

- **Local signal processing** that reduces energy requirements during active use
- **Efficient algorithms** that eliminate reliance on cloud servers
- **Overall minimal carbon footprint**, when paired with low-resource microcontrollers

These benefits make the system suitable for long-term studies, classroom use, and global deployment in settings where energy resources are limited.

2.3 Social and Ethical Analysis

The social and ethical impact of cognitive and biometric analytics are priorities in the design of this system. Physiological signals such as EDA and HRV are deeply personal, and misinterpretation can carry serious consequences. In discussions with stakeholders like Dr. Chrysikou, the need for ethical transparency, participant protection, and contextual interpretation was strongly emphasized.

Our system addresses these concerns through:

- Strong data protection, using AES-256 encryption
- Adherence to IRB-informed consent practices
- Transparent processing, ensuring stakeholders understand how data are transformed and interpreted
- Open-source access, excluding hidden algorithms or undisclosed processing steps

By promoting interpretability and proper utilization of data, the system helps prevent misuse of biometric information.

3. Conceptual Design

The software collects data from the hardware which sends the measured EDA and HRV data from the hardware to the computer running the software by utilizing a high-speed Bluetooth signal. Then the software on the computer processes the signal by utilizing the Tonic and Phasic Decomposition approach and possibly machine learning tools. After processing the signal, the software creates a csv file and writes the processed data on to the csv file and displays the results to the user via a live graph or a summary of results with a complete graph.

If the software detects an error such as hardware disconnects, signal issues due to interference, low storage on the computer with the software, or error within the software, the software displays an error prompt on the screen and goes through the necessary error handling procedures. Users will be able to export the csv data collected before and during the error occurs so users will be able to analyze when and where the error occurred and use the "usable data" collected (i.e. data collected before the wearer got too excited near the end of data collection and accidentally disconnects a sensor.

3.1 Relationship with Hardware

The hardware collects and transmits EDA and HRV data by using its built-in sensors. Data is transmitted wirelessly from the wearable device to the computer running the software via Bluetooth Low Energy 5.0. Wireless was chosen over wires since wires are prone to accidental disconnection during data transmission since the wearable device is used to transmit data is wearable that will be worn by a patient. All signals processing and analysis are handled on the computer running the software by utilizing PC-Based Architecture.

3.2 Technical Specifications and Constraints

The main constraint of the software design is the processing power required to run a PC-Based Architecture with Convex Optimization algorithms that utilize a heavy matrix math. Since the data also utilizes patient's vitals, processing data over the cloud risks patient's privacy. Since live processing is a priority, latency is also a constraint that needs to be addressed. These restrictions require the system to have a powerful component that will be able to run the Convex Optimization Algorithm efficiently and fast as possible so that the researchers can gather more data within the 6–8-hour battery life of the wearable device.

Given that the runtime of the heavy matrix calculations of the Convex Optimization Algorithm depends on the CPU, the software benefits computers with higher clock speeds and cores used to run the Convex Optimization Algorithm, providing the following feasible CPU specifications assuming signal is processed on the computer running the software.

Need	Intel core i5-11400 (4.4GHZ/6 cores)	Intel core i5-12400 (4.4GHZ/6 cores)	Intel core i7-11400 (4.8GHZ/8 cores)	Intel core i7-12400 (4.9GHZ/8 cores)
Maximum run time of less than 10 minutes	Meets requirement	Meets requirement	Meets requirement	Meets requirement
Speed of running Convex Optimization Algorithm	.	...	....	....

Figure 3. Minimum requirements based on the Convex Optimization Requirements

3.3 Design Concepts

3.4.1 Concept 1

The first concept of our software utilizes convex optimization to deconvolve signals transmitted to the computer by running the software from the wearable device via Bluetooth. This differs from peak detection utilized by average smartwatches and other wearable devices on the market. However, this approach requires a system that can handle heavy matrix calculations.

3.4.2 Concept 2

The second concept focuses on processing the signals on the wearable device and transmitting the processed data to the software used to display results. This simplifies the software since the data processing occurs on the wearable device, making the software implementation simpler and more intuitive.

3.4.3 Concept 3

The third concept focuses on cloud computing, by utilizing the sensor data sent from the wearable device to a server that contains powerful hardware components that can run the Convex Optimization Algorithm efficiently without needing to invest in high-end components for the research computer running the software.

3.4 Design Matrices

Parameter	Design 1	Design 2	Design 3
Time to process data	5	1	5
User data security	5	5	1
Compatibility	3	2	5
Total Score	13	8	11

Figure 4. Table Showing Output of Design Matrix

3.4.1Design Matrix Evaluation Explanation

For the design matrix, each concept was evaluated against three required parameters to determine the most efficient, secure, and compatible concept to utilize for the software. Each design was tested for each parameter and a score of 1 to 5 was given to each design for each parameter, determining the best design by the highest total score. For the first parameter, time to process data, design 1 and design 3 had a faster data processing time compared to design 2. Sometimes, design 2 fails to process the data due to weaker components compared to components utilized by design 1 and design 2.

For the second parameter, user data security, design 1 and design 2 scored the highest, since the patient's data is processed on board, either on the computer running the software or the wearable device, without uploading any patient data outside of the "ecosystem". Design 3 requires the data to be uploaded to the cloud over the internet, compromising the security of the patient's data.

The third parameter, compatibility, determines how independently the components of the computer run the software and how long it takes to process data. Design 3 scored the highest since design 3 relies on cloud computing for signal processing, which causes the time to process data to be independent from the components used by the computer running the software. Design 1 had a less score than design 3 since the time to process data can vary for design 1 based on the components used on the computer. For example, a computer with a CPU of a base clock speed of 4.8 GHZ will be able to process data faster than a computer with a CPU of a base clock speed of 3.3GHZ. Design 2 scored the lowest since the microcontrollers used for the wearable device wasn't designed to perform heavy calculations required for convex optimization algorithms. However, it didn't score the lowest score, 1, since the time to process data does not depend on the computer running the software.

4. Approach

4.1 Detailed Design

The detailed design demonstrates the relationship between the connected hardware device and the software utilized to gather the data transmitted by the device. The device transmits the EDA and HRV data via Bluetooth Low Energy 5.0. The software utilizes the Tonic and Phasic decomposition and analysis models.

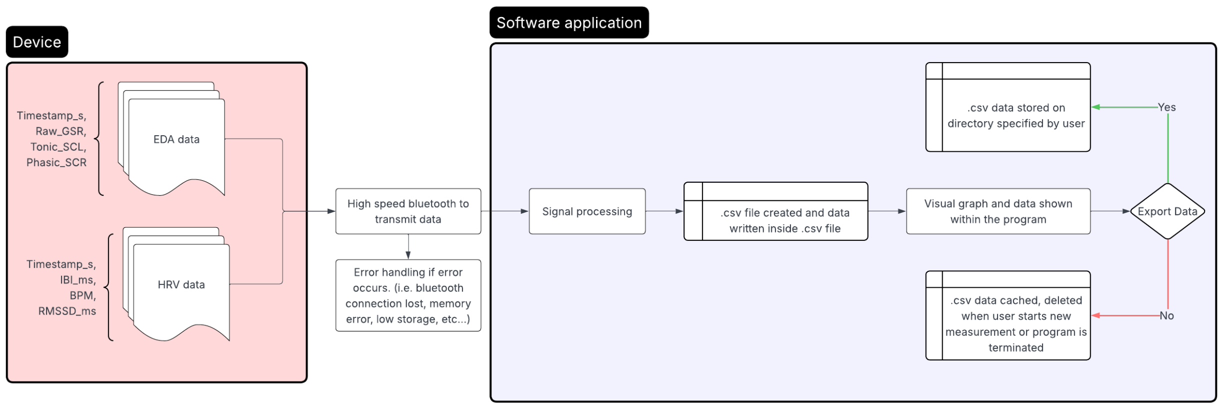

Device
Timestamp_s, Raw_GSR, Tonic_SCL, Phasic_SCR
EDA data
Timestamp_s, IBI_ms, BPM, RMSSD_ms
HRV data
High speed bluetooth to transmit data
Error handling if error occurs. (i.e. bluetooth connection lost, memory error, low storage, etc...)
Software application
Signal processing
.csv file created and data written inside .csv file
Visual graph and data shown within the program
Export Data
Yes
.csv data stored on directory specified by user
No
.csv data cached, deleted when user starts new measurement or program is terminated

Figure 5. Flow Chart Demonstrating Detailed Design

4.1.2 Signals Processing

A signal processing approach that can be utilized is the Tonic and Phasic Decomposition via AC recording. This approach allows the user to distinguish sweat from wet skin by utilizing Susceptance, the relationship between Admittance and Conductance, as shown in figure below. Susceptance measures the capacitive properties of the skin, which is primarily determined by the stratum corneum, the outermost layer of the epidermis. Assuming the hardware supports AC signals, the software distinguishes sweat gland activity from hydration since wet skin has a higher Susceptance value, B, than sweaty skin. The sweat gland activity also generates a spike in Conductance value, G.

Admittance vs. Conductance: In an AC system, the software receives a complex impedance signal (Z), composed of Resistance (R) and Reactance (X). The inverse is Admittance (Y), composed of Conductance (G) and Susceptance (B).

$$Y = G + jB$$

Using this approach, several models can be applied to process out signals. The following models and how each models process Admittance to obtain the measurements of stress as mentioned from “Innovations in Electrodermal Activity Data Collection and Signal Processing: A Systematic Review.":

- Linear Convolution Models (SCRalyze)
  - This model treats each Skin Conductance Response (SCR) as the output of a linear, time-invariant filter which allows us to use full-waveform.
- Non-Negative Deconvolution (Ledalab)
  - This model separates each signal into tonic and phasic components and extracts the discrete and compact SCR events, which improves the measurement precision
- Convex Optimization Models (cvxEDA)
  - This model displays admittance as a sum of tonic, phasic, and noise terms, which constraints phasic activity to be sparse and non-negative.

These processing models would allow the users and the software to detect indicators of arousal by transforming and converting the raw admittance data to meaningful data. These processing models also prevent inaccurate readings from skin hydration since the tonic-phasic decomposition can differentiate between hydrated skin and skin covered in sweat from sweat gland activation, enabling real-time stress detection by the software.

4.2 Software Development Environment

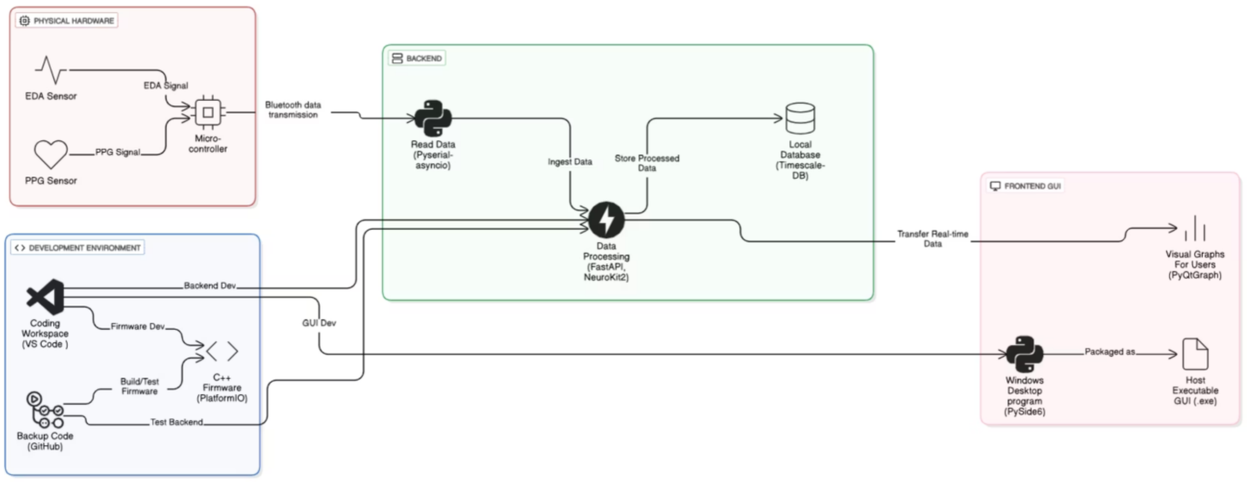

PHYSICAL HARDWARE
EDA Sensor
EDA Signal
PPG Sensor
PPG Signal
Micro-controller
Bluetooth data transmission
BACKEND
Read Data (Pyserial-asyncio)
Ingest Data
Store Processed Data
Local Database (Timescale-DB)
Data Processing (FastAPI, NeuroKit2)
Transfer Real-time Data
FRONTEND GUI
Visual Graphs For Users (PyQtGraph)
DEVELOPMENT ENVIRONMENT
Backend Dev
Coding Workspace (VS Code)
Firmware Dev
GUI Dev
Build/Test Firmware
C++ Firmware (PlatformIO)
Backup Code (GitHub)
Test Backend
Windows Desktop program (PySide6)
Packaged as
Host Executable GUI (.exe)

Figure 6. Overall Flow of Software Development Environment

Our primary development environment is Visual Studio Code. We have configured it as a dual language workspace to support both low level embedded C++ and high-level

Python analysis. This setup is used for bridging the gap between firmware development and data processing.

For the firmware layer, we utilize PlatformIO within Visual Studio Code. This transforms our microcontroller code into a managed project with specific build environments and dependency trees. We declare sensor drivers in a configuration file rather than manually linking libraries. This creates a reproducible firmware blueprint that ensures consistent builds across all developer machines.

On the graphical user interface side, we adopt a "design first" workflow using Qt Designer and PySide6. Instead of manually coding the placement of widgets, we visually construct the interface files. This allows us to inspect ergonomics and data hierarchy before writing logic. This separation of visual design from business logic serves as our architectural cornerstone. The interface file acts as a strict contract. The backend is built specifically to satisfy the data requirements defined by the GUI. For example, it must provide a list of float values for the tonic component to ensure seamless integration. With the design environment established we turn our attention to how the data is presented to the user.

4.3 Embodiment

Our core value proposition is transforming biological processes into visible insights in real-time. To achieve this, we avoided standard plotting libraries like Matplotlib. Matplotlib lacks support for GPU acceleration. The libraries used are often too heavy for weaker systems pertaining to several high frequency sensor streams. Instead, we implement PyQtGraph. This library is optimized for high performance engineering visualizations and allows us to render physiological changes without lag.

The goal for the visualization is to get multiple graphs going with various data that would prove important to a researcher. The first is the Raw Signal View. This provides a direct oscilloscope style feed of what the sensors are reading. Running in parallel is the Decomposed View. This view overlays the separated signal components to show both the Tonic baseline and the Phasic reactions concurrently. This allows the user to distinguish between lingering mood states and acute shock responses. Finally, we implement the Spectral Waterfall. This is a heatmap animation visualizing the power density of the signal over time. This requires processing Fast Fourier Transforms in near real time to paint a scrolling texture. This texture represents the energy intensity of the sympathetic nervous system.

4.3.2 Simulations and Testing

Before deploying sensors on human subjects, we run simulations to validate our software. We utilize the WESAD (Wearable Stress and Affect Detection) dataset. This is an industry standard collection of skin conductance recordings with annotated stress events that serve as our basis for tuning.

One proposed strategy involves a Virtual Sensor module within the Python backend. This module ingests the WESAD CSV benchmark files but mimics the behavior of the physical Bluetooth device. It pushes data packets at the exact 50Hz sampling rate. This allows us to validate our algorithms against known physiological events. We can check if the benchmark indicates a stress peak at a specific timestamp. If our algorithm fails to detect it, we can isolate the software flaw independent of hardware noise. This strategy ensures that the software is already a proven quantity before we move to human trials.

4.3.3 Numerical Modeling

Beyond simple data logging, our software models the underlying physiology of the human sudomotor system. Our approach is grounded in a specific mathematical concept. We treat skin conductance as a convolution of a sparse neural driver, the sweat gland's impulse response function, an red a tonic baseline.

To isolate these factors, we implement Convex Optimization algorithms via the NeuroKit2 library. This model imposes constraints. Specifically, it assumes that neural drivers are sparse, or bursty, and that tonic components are smooth. This allows us to numerically strip away drift caused by sweat accumulation or electrode polarization. Furthermore, we employ Spectral Density Modeling. This quantifies Sympathetic Tone by integrating power within the 0.045 to 0.25 Hz frequency band. This is what separates the device from a consumer wrist monitor. In tandem with the software, It mathematically reverses the physical process of sweating to estimate the neural activity in the brainstem.

5. Materials and Resources

5.1 Hardware

To support these modeling capabilities, we require a specific set of physical and digital resources. Although this is a software proposal, our stack is strictly defined by its physical inputs. Our system relies on a custom wearable unit equipped with sensors for EDA and PPG. Data aggregation is handled by an ESP32 or similar architecture microcontroller. This chip is responsible for sensor packetization. Connectivity is established via Bluetooth Low Energy (BLE) using specific characteristics defined by the hardware team. Finally, the host station is a standard Windows based laptop with Bluetooth 5.0 capability. With the hardware interface defined we have selected a software stack that maximizes performance on these physical constraints.

5.2 Software

Our stack is a carefully selected assembly of open-source tools designed for scientific rigor and performance. Visual Studio Code serves as the primary hub. It manages Python code, C++ compilation, and database connections. GitHub Actions handles Continuous Integration. It automatically runs unit tests on database changes or firmware logic updates to prevent errors. For firmware management, PlatformIO is used for its library management. This allows us to lock specific sensor library versions to ensure builds are cohesive across development computers.

The backend logic is driven by Python. We utilize FastAPI for a high performance, asynchronous data ingestion engine. Pyserial-asyncio-fast manages non-blocking Bluetooth reads. NeuroKit2 provides the pre validated implementations of cvxEDA and signal cleaning algorithms. For data persistence, TimescaleDB, an extension of PostgreSQL, was selected over SQLite. It was chosen for its ability to handle high velocity time series writes. This extended table structure would allow us to sort and retrieve information at a speed required to achieve our “real-time” goals. SQLite is not as capable of these goals. It is a good choice for simple and serverless applications but for sensor data it is not as capable. Its hyper table structure supports local storage of high resolution sensor readings and fast analytical queries. The front end is built upon PySide6. This provides the native Windows application shell that hosts the PyQtGraph visualizations described in the Approach section.

6. Project Management

6.1 Team Organization

Our team operates as a cohesive engineering unit where every member owns a primary domain but maintains knowledge of the full system. The Data Architect serves as the primary bridge between hardware outputs and software inputs. This role focuses heavily on the TimescaleDB implementation and manages integration points. Their goal is to ensure that firmware transmission formats align perfectly with the database schema. Working closer to the hardware side is the Firmware Specialist. This groupmate manages the PlatformIO environment and the Data Ingestion layer. This ensures the ESP32 captures clean data and that the Python ingestion script captures every packet without loss. Sitting at the top of the stack, the Integration Specialist applies intelligence to the stored data. This involves implementing NeuroKit2 algorithms for decomposition and managing the experimental integration of a Local Large Language Model for natural language data context.

We utilize an Agile workflow adapted for academic constraints. We assign "Task Leaders" for specific modules. To avoid problems where people run off and build their own incompatible pieces, we adhere to a strict "Interface First" rule. This means data structures must be defined and agreed upon by the team before any implementation code is written. With our roles defined we have mapped out a three-term schedule to guide our development from prototype to final release.

6.2 Schedule and Milestones

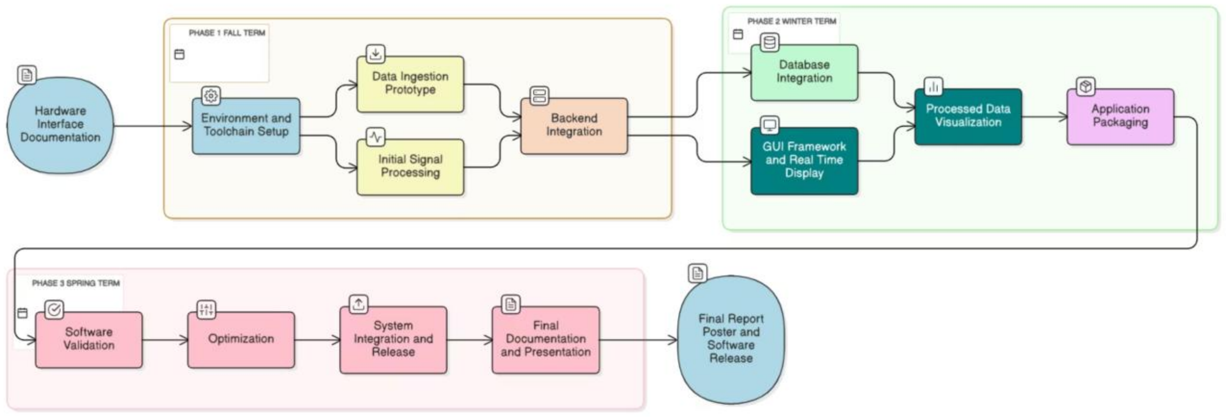
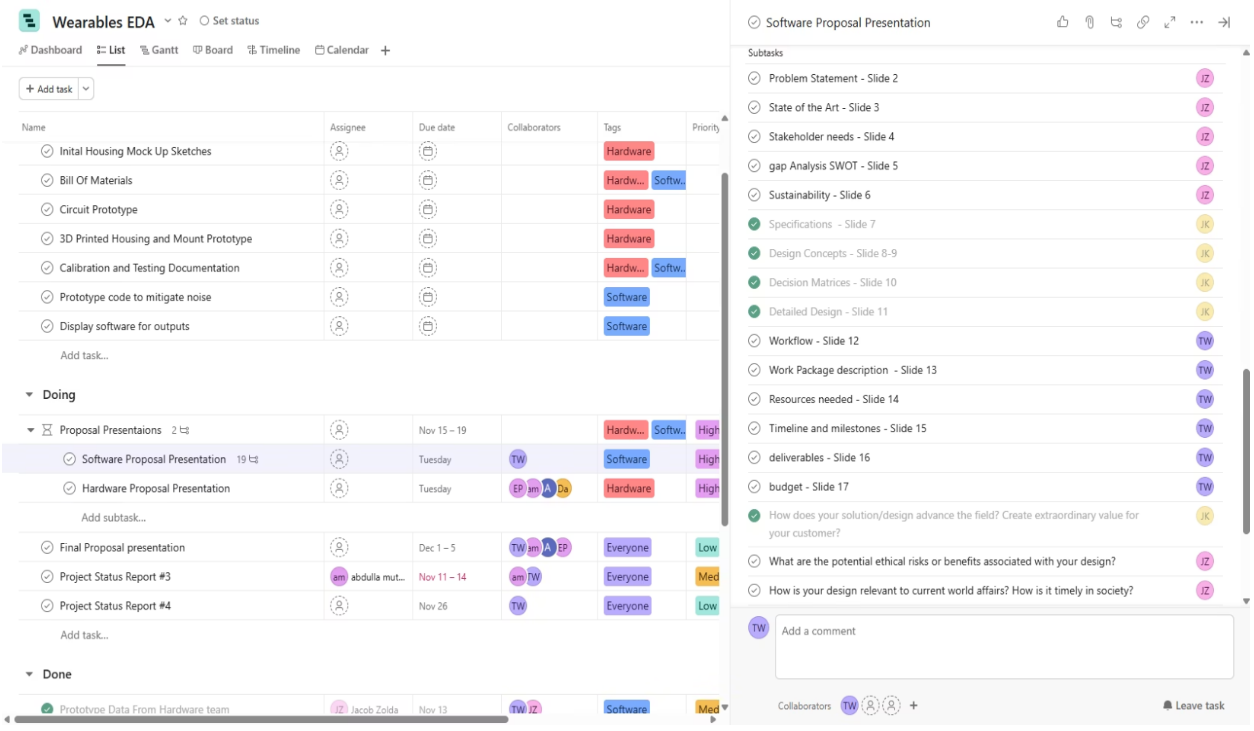

Hardware Interface Documentation
PHASE 1 FALL TERM
Environment and Toolchain Setup
Data Ingestion Prototype
Initial Signal Processing
Backend Integration
PHASE 2 WINTER TERM
Database Integration
GUI Framework and Real Time Display
Processed Data Visualization
Application Packaging
PHASE 3 SPRING TERM
Software Validation
Optimization
System Integration and Release
Final Documentation and Presentation
Final Report Poster and Software Release

Figure 6. Work Packages and Milestones

Our timeline is broken into three distinct phases to ensure steady progress. Phase 1 (Fall Foundation) focuses on the groundworks to hopefully get running in the Winter ter,. It begins with environment and toolchain setup. The major milestone for this phase is the Data Ingestion Prototype. This proves the sensor to screen pipeline. The phase concludes with the Backend Architecture being worked on. Phase 2 (Winter Implementation) shifts to heavy coding. During this time, we will finalize Database Integration and construct the GUI Framework using PySide6. The critical milestone here is Real time Plotting. The phase concludes with packaging the software into a standalone executable. Phase 3 (Spring Validation) shifts from building to testing. We focus on validation using WESAD simulations to verify mathematical accuracy. We also execute optimization for performance and build export tools for researchers. The final milestone is the Final Documentation and Presentation of a research grade software package. To stay on track with these long-term milestones we adhere to a strict weekly operational routine.

6.2.2 Sprint description



Name	Assignee	Due date	Collaborators	Tags	Priority
Inital Housing Mock Up Sketches				Hardware	
Bill Of Materials				Hardw... Softw..	
Circuit Prototype				Hardware	
3D Printed Housing and Mount Prototype				Hardware	
Calibration and Testing Documentation				Hardw... Softw..	
Prototype code to mitigate noise				Software	
Display software for outputs				Software	
Doing					
Proposal Presentaions 2		Nov 15 – 19		Hardw... Softw..	High
Software Proposal Presentation 19		Tuesday	TW	Software	High
Hardware Proposal Presentation		Tuesday	EP am A Da	Hardware	High
Final Proposal presentation		Dec 1 – 5	TW am A EP	Everyone	Low
Project Status Report #3	am abdulla mut...	Nov 11 – 14	am TW	Everyone	Med
Project Status Report #4		Nov 26	TW	Everyone	Low
Done					
Prototype Data From Hardware team	JZ Jacob Zolda	Nov 13	TW JZ	Software	Med

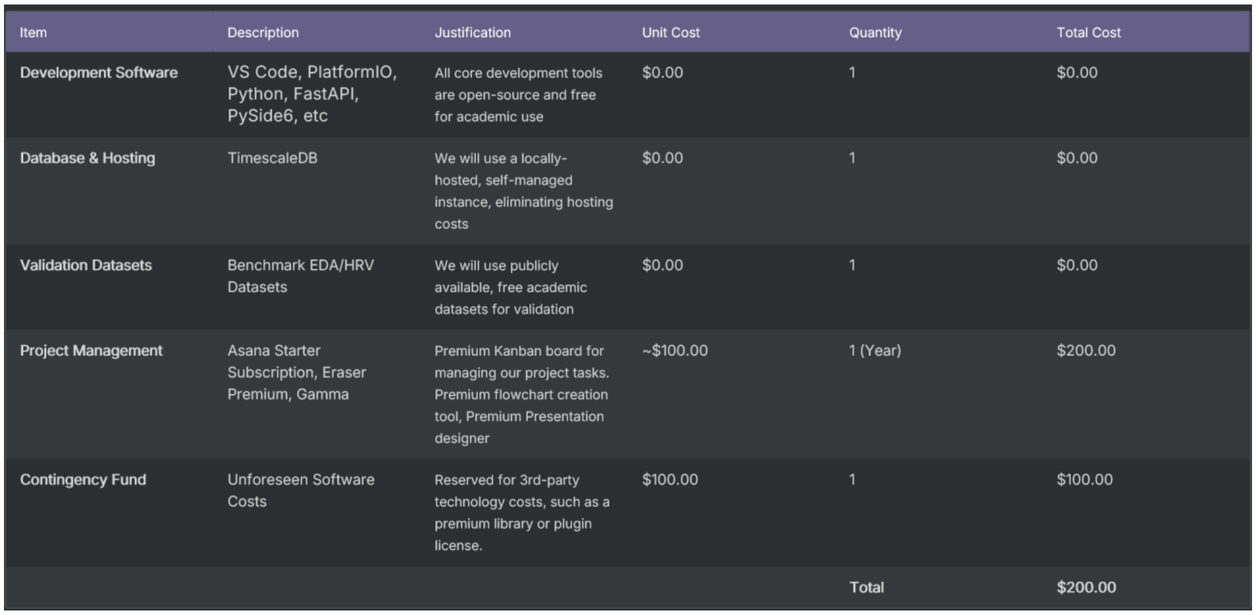
Software Proposal Presentation

Subtasks	
Problem Statement - Slide 2	JZ
State of the Art - Slide 3	JZ
Stakeholder needs - Slide 4	JZ
gap Analysis SWOT - Slide 5	JZ
Sustainability - Slide 6	JZ
✓ Specifications - Slide 7	JK
✓ Design Concepts - Slide 8-9	JK
✓ Decision Matrices - Slide 10	JK
✓ Detailed Design - Slide 11	JK
Workflow - Slide 12	TW
Work Package description - Slide 13	TW
Resources needed - Slide 14	TW
Timeline and milestones - Slide 15	TW
deliverables - Slide 16	TW
budget - Slide 17	TW
✓ How does your solution/design advance the field? Create extraordinary value for your customer?	JK
What are the potential ethical risks or benefits associated with your design?	JZ
How is your design relevant to current world affairs? How is it timely in society?	JZ

Figure 7. Asana Project Management Board

A project management software called Asana is used. The benefit of using Asana is that is allows both hardware and software team to collaborate in an environment where tasks are assigned. There are also features like GANNT charts and auto-updating dependency timelines we use. The software team primarily uses it to assign coding tasks. To maintain a good pace, we do a rigid weekly rhythm. Tuesdays are dedicated to sprint planning. We review the Asana board and move cards from Backlog to Doing. Thursdays are reserved for code review and integration. We use this time to verify that assigned work functions correctly on the main branch. Every feature, bug, or document exists as a card to ensure accountability.

6.3 Budget



Item	Description	Justification	Unit Cost	Quantity	Total Cost
Development Software	VS Code, PlatformIO, Python, FastAPI, PySide6, etc	All core development tools are open-source and free for academic use	$0.00	1	$0.00
Database & Hosting	TimescaleDB	We will use a locally-hosted, self-managed instance, eliminating hosting costs	$0.00	1	$0.00
Validation Datasets	Benchmark EDA/HRV Datasets	We will use publicly available, free academic datasets for validation	$0.00	1	$0.00
Project Management	Asana Starter Subscription, Eraser Premium, Gamma	Premium Kanban board for managing our project tasks. Premium flowchart creation tool, Premium Presentation designer	~$100.00	1 (Year)	$200.00
Contingency Fund	Unforeseen Software Costs	Reserved for 3rd-party technology costs, such as a premium library or plugin license.	$100.00	1	$100.00
				Total	**$200.00**

Figure 8. Chart Showing Overall Budget

Our budget is lean. We leverage open-source technology for development while investing in management efficiency. We have allocated a Total Budget of $200. We spend $0 on Development Tools and Databases by utilizing free, permissive license tools. This also allows for local hosting to align with data privacy goals. We allocate $100 for Project Management. This funds premium tools including Asana Premium, Eraser, and Gamma to prioritize clear communication. The remaining $100 is set aside as a Contingency buffer. This is reserved for unexpected costs, such as paid library licenses, or potential backup hosting fees. The final measure of our financial and technical planning will be our ability to meet specific performance targets.

6.4 Success Benchmarks

Success is defined by specific, measurable metrics across four categories. Validation requires high accuracy. The processed output for Phasic/Tonic separation must achieve greater than 90% similarity to the ground truth provided in the WESAD benchmark dataset. Performance dictates responsiveness. The GUI must render scrolling graphs at a consistent frame rate of over 30 FPS without lag. Additionally, database ingestion must consume less than 80% CPU under load on a standard specification laptop. Usability standards require the interface to adhere to ISO 9241-12. This ensures color contrast and layout reduce cognitive load for clinicians. Finally, Signal Decomposition success is measured by the system's ability to autonomously decompose raw EDA signals. It must separate distinct tonic and phasic components while effectively filtering motion artifacts to produce an output that aligns with our stakeholders needs.

Codes Referenced

- ISO 9241-12 Ergonomic requirements for office work with visual display terminals (VDTs) — Part 12: Presentation of information https://cdn.standards.iteh.ai/samples/16884/6e77a9f4a7ac485bb8beaab86eb99923/ISO-9241-12-1998.pdf
- ISO 9241-210 Ergonomics of human-system interaction — Part 210: Human-centered design for interactive systems https://cdn.standards.iteh.ai/samples/77520/33cc3cf17e5e4f1894ee872ed4f54fb6/ISO-9241-210-2019.pdf
- AES-256 Advanced Encryption Standard (AES) Category: Computer Security, Subcategory: Cryptography https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197-upd1.pdf

Sources

1. Gamboa, Patricia, et al. *"Electrodermal Activity Analysis at Different Body Locations." Sensors*, vol. 25, no. 6, 2025, pp. 1–12. (novaresearch.unl.pt)
2. Bach, Dominik R. "A Head-to-Head Comparison of SCRalyze and Ledalab, Two Model-Based Methods for Skin Conductance Analysis." *Biological Psychology*, vol. 103, Dec. 2014, pp. 63–68. *PubMed Central*, https://doi.org/10.1016/j.biopsycho.2014.08.006. Accessed 16 Nov. 2025.
3. Bari, Dindar S., Haval Y. Y. Aldosky, Christian Tronstad, and Ørjan G. Martinsen. "Disturbances in Electrodermal Activity Recordings Due to Different Noises in the Environment." *Sensors*, vol. 24, no. 16, 2024, article 5434, https://doi.org/10.3390/s24165434. (mdpi.com)
4. Greco, Alberto, et al. "cvxEDA: A Convex Optimization Approach to Electrodermal Activity Processing." IEEE Transactions on Biomedical Engineering, vol. 63, no. 4, 2016, pp. 797-804.
5. International Organization for Standardization. Ergonomic requirements for office work with visual display terminals (VDTs) — Part 12: Presentation of information. ISO 9241-12:1998, 1998

6. Makowski, Dominique, et al. "NeuroKit2: A Python toolbox for neurophysiological signal processing." Behavior Research Methods, vol. 53, no. 4, 2021, pp. 1689-1696.
7. Namboodiri, Vinod, and Toolika Ghose. "To Cloud or Not to Cloud: A Mobile Device Perspective on Energy Consumption of Applications." 2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), IEEE, 2012, pp. 1-9.
8. PlatformIO: An open source ecosystem for IoT development. PlatformIO, https://platformio.org/. Accessed 5 Dec. 2025.
9. Posada-Quintero, Hugo F., and Ki H. Chon. "Innovations in Electrodermal Activity Data Collection and Signal Processing: A Systematic Review." Sensors, vol. 20, no. 2, 2020, p. 479.
10. PyQtGraph: Scientific Graphics and GUI Library for Python. PyQtGraph, https://www.pyqtgraph.org/. Accessed 5 Dec. 2025.
11. Ramírez, Sebastián. FastAPI. https://fastapi.tiangolo.com/. Accessed 5 Dec. 2025.
12. Schmidt, Philip, et al. "WESAD: A Multimodal Dataset for Wearable Stress and Affect Detection." Proceedings of the 20th ACM International Conference on Multimodal Interaction, 2018, pp. 400-410.
13. TimescaleDB. Timescale, Inc., https://www.timescale.com/Accessed 5 Dec. 2025.