Brain Wave Analysis: Software Development Project Proposal

Prepared for:
Yossi Buskila

Prepared by:
Khalil Ibrahim
Tri Vinh Phan
Juliet Jin
Nathan Gatto

Table of Contents

Executive Summary

The Neurophysiological Data Analysis Software project aims to develop a desktop application for analysing neurophysiological recordings, specifically focusing on extracellular signals, resonance frequency, and Potassium (K+) clearance to assist neuroscience research. The software will be designed to streamline data processing, provide accurate signal analysis and enhance visualisation for researchers studying brain wave activity.

The core objectives of the project are:

- To develop an intuitive Graphical User Interface (GUI) that allows users to easily load and analyse ABF files.

- To implement advanced signal processing algorithms such as FFT analysis, noise filtering, and exponential decay fitting for neurophysiological data interpretation.

- To ensure cross-platform compatibility (Windows & Mac) for accessibility and usability.

- To provide comprehensive data visualisation tools (heat maps, power spectra and trace graphs) to help researchers better interpret their findings.

- To integrate robust error handling, debugging support, and data privacy measures to protect sensitive data

By successfully delivering this project, the team aims to create an intuitive and accessible tool that enables students and researchers to analyse neurophysiological signals without requiring prior coding experience.

Introduction

As project plan provides a general strategy of the project by explaining the Work Breakdown Structure to understand the the essential tasks for software development, and thus, allocating the tasks for each member, the project proposal will focus more on the progression of the project, which means that the project should be planned on several solutions to decide the final method of developing the software. Moreover, the requirements of the software should be outlined in the project proposal so that the development stage can be engaged accordingly, which increases the efficiency of the software.

In this project proposal, there are several sections that are essentially needed to demonstrate. The high level business function explains the tasks in different levels of priority to complete since there are numerous tasks that are the core or the supporting objects of the software, and the evaluation plays an important role in this section so that team members can prioritise on core tasks.

In terms of functional and non-functional requirements, they are the core of the project proposal for several reasons. First of all, functional requirements provide specific functions of the software to demonstrate how the software will be operated and what the expected results that the users desire. Therefore, to achieve these expected results, the second reason is the user stories which can perform the process of each function in the software. Finally, non-functional requirements concentrate on other functions that are not as prior as the core functions. This means that they can perform other tasks to highlight the diversity of the software, even though it is less important for the client to expect.

Alternative solutions are also one of the important sections that contribute to this project because it provides opportunities to suggest various ideas, including the description, both advantages and disadvantages, and therefore, make a team agreement to the solution that they believe the most and provide reasons why they select the solution.

Release schedule demonstrates the time consumption to complete each task of the project. This schedule will support team members to observe how long they can be developed to implement in the software, including different stages such as design, development and testing.

On the other hand, cost benefit analysis handles the cost of the project, which means that each task should be considered to estimate the cost so that the client can understand if the expenditure is suitable to his finance. This section also provides the benefits of each cost for the development to support the decision of the client because it is difficult to justify the project when there are several negative impacts that affect the client.

High Level Business Functions

ID	High-Level Business Function	Ranking	Description
BF1	Neurophysiological Data Processing	Essential	The system must be able to load, parse and process ABF files to extract extracellular, resonance frequency, and K+ Clearance data.
BF2	Signal Analysis & Computation	Essential	The software must implement FFT analysis, noise filtering and exponential decay fitting to analyse brain wave signals.
BF3	Graphical User Interface (GUI)	Essential	The system must provide a user friendly interface with options to load files, run analyses, and visualise results.
BF4	Data Visualisation & Reporting	Essential	The software must generate plots, graphs and heat maps of processed signals for interpretation.
BF5	System Integration & Performance	High Priority	The software must efficiently integrate front-end GUI and back-end signal processing.
BF6	Cross-Platform Compatibility	High Priority	The application should be compatible with both Windows and Mac.

BF7	User Experience & Usability	High Priority	The system should be intuitive allowing users with minimal programming experience to operate it efficiently.
BF8	Security & Data Privacy	Medium Priority	The system must handle sensitive data securely preventing unauthorised access.
BF9	Error Handling & Debugging Support	Medium Priority	The system should provide error messages and debugging logs to assist users in troubleshooting issues.
BF10	FTP Server Storage & Remote Access	Low Priority	A future version of the system could allow users to store processed data in the FTP server and access it from multiple devices

Requirements

Functional Requirements

ID: F001	Feature: ABF File Loader	Priority: Essential
BF1: Neurophysiological Data Processing		Estimate: 12 hrs
Story: As a researcher, I want to be able to load ABF files into the software so as to analyse neurophysiological recordings.		

ID: F002	Feature: Episode Navigation	Priority: Essential
BF1: Neurophysiological Data Processing		Estimate: 10 hrs
Story: As a researcher, I want to navigate through multiple episodes within a recording to analyse them individually.		

ID: F003	Feature: K^+ Clearance (Tau and Rate Constant)	Priority: Essential
BF2: Signal Analysis & Computation		Estimate: 16 hrs
Story: As a researcher, I want to calculate Tau and 1/Tau from K^+ signals so I can measure clearance rates.		

ID: F004	Feature: FFT Analysis	Priority: Essential
BF2: Signal Analysis & Computation		Estimate: 16 hrs
Story: As a researcher, I want to run a FFT on signal data so I'll be able to examine its frequency components.		

ID: F005	Feature: ZAP Profile and Resonance	Priority: Essential
BF2: Signal Analysis & Computation		Estimate: 16 hrs
Story: As a researcher, I want to compute a ZAP profile and find resonance frequency to analyse the neuron’s frequency-dependent impedance		

ID: F006	Feature: Power Spectrum Heat Map	Priority: Essential
BF4: Data Visualisation & Reporting		Estimate: 14 hrs
Story: As a researcher, I want to generate a heatmap of the power spectrum over a period of time to visualise its patterns.		

ID: F007	Feature: Graph Output and Visualisation	Priority: Essential
BF4: Data Visualisation & Reporting		Estimate: 12 hrs
Story: As a researcher, I want to visualise the signals and results with clear graphs to be able to interpret the data easily.		

ID: F008	Feature: Integrated Frontend-Backend Architecture	Priority: High
BF5: System Integration & Performance		Estimate: 10 hrs
Story: As a researcher, I want the software to work seamlessly so that I can process and visualise data without any errors.		

ID: F009	Feature: Guided Workflow with Prompts	Priority: High
BF7: User Experience and Usability		Estimate: 11 hrs
Story: As a user, I want guidance while using the software so it can be easier to understand what each feature does without having to need training.		

ID: F010	Feature: Cross-Platform Installer	Priority: High
BF6: Cross-Platform Compatibility		Estimate: 11 hrs
Story: As a researcher, I want to be able to install the software on both Windows and Mac systems.		

ID: F011	Feature: Security File Handling	Priority: Medium

BF8: Security & data Privacy	Estimate: 12 hrs
Story: As a researcher, I want the software to avoid any modifications to the data files to maintain data integrity.	

ID: F012	Feature: Comprehensive Experiment Export	Priority: Medium
BF10: FTP Server Storage & Remote Access		Estimate: 8 hrs
Story: As a researcher, I want to be able to export my entire experiment session so that I can then access it later or from another device.		

ID: F013	Feature: File save integration	Priority: Low
BF10: FTP Server Storage & Remote Access		Estimate: 10 hrs
Story: As a researcher, I want to save my results on the internet so they can be easily accessible on different computers or sharing purposes.		

ID: F014	Feature: User Account and Experiment Saving	Priority: Low
BF10: FTP Server Storage & Remote Access		Estimate: 14 hrs
Story: As a researcher, I want to have a user account so that I can save and revisit my experiments across sessions and/or devices.		

Non-functional Requirements

ID: NF001	BF1: Neurophysiological Data Processing	Priority: Essential	Estimate: 7 hrs
Story: As a researcher, I want the system to load larger ABF files, around up to 100MB, quickly without freezing, and be able to view the first episode within 5 seconds of loading. The files are also to never be modified during analysis to maintain data integrity.			

ID: NF002	BF2: Signal Analysis & Computation	Priority: Essential	Estimate: 8 hrs
Story: As a researcher, I want FFT and fitting computations to be completed within 3 seconds for a 10 second signal and all numerical calculations to be accurate to at least 4 decimals.			

ID: NF003	BF3: Graphical User Interface (GUI)	Priority: Essential	Estimate: 6 hrs
Story: As a user, I want the UI to display correctly on screen resolutions and UI buttons to react within 2 seconds for a responsive interface. All UI components need to be accessible and follow usability guidelines.			

ID: NF004	BF4: Data Visualisation & Reporting	Priority: High	Estimate: 6 hrs
Story: As a researcher, I want all plots to render within 2 seconds after analysis with axis labels, units and a legend, and to be able to export files with a timestamp in their file names.			

ID: NF005	BF5: System Integration & Performance	Priority: High	Estimate: 6 hrs
Story: As a developer, I want backend and frontend modules to be separate for easier maintenance and its architecture to support future modular analysis integration.			

ID: NF006	BF6: Cross-Platform Compatibility	Priority: High	Estimate: 6 hrs
Story: As a user, I want the software to run identically on Windows and MacOS as well as the installer to work offline while including all dependencies.			

ID: NF007	BF7: User Experience & Usability	Priority: High	Estimate: 7 hrs
Story: As a user, I want to complete basic tasks in under 10 minutes without reading a manual, where every analysis option includes a brief help description and access key functions with no more than 3 clicks on the main menu.			

ID: NF008	BF8: Security & Data Privacy	Priority: Medium	Estimate: 4 hrs
Story: As a user, I want all exported data to be saved in a separate folder to avoid overwriting any originals and no data to leave my computer unless chosen to.			

ID: NF009	BF9: Error Handling & Debugging Support	Priority: Medium	Estimate: 5 hrs
Story: As a user, I want all system errors logged with a timestamp and in a user-friendly message so that it can be troubleshooted.			

ID: NF010	BF10: FTP server Storage & Remote Access	Priority: Low	Estimate: 8 hrs
Story: As a user, I want to log in with Open Authorisation to save my Google Drive securely and upload files to be encrypted during transfer for data protection.			

Alternative Solutions

Alternative Solution 1

Solution Overview

Solution 1 focuses on using matlab as the backend logic. Using python to access the benefits of Matlab allows us to build a better GUI than capable using matlab alone. This solution would expand on the current solution used by the client. The user-friendly GUI requested would be integrated with pyQT as the frontend. This solution will be capable of accurately and swiftly analysing three different types of neurophysiological recordings with a well designed, user-friendly interface.

Languages + Services + Frameworks and APIs

Matlab is known for its matrix-based language making it ideal for mathematical tasks. Matlab is mostly used for numerical computing, data analysis, data visualisation and more. For this project, using this programming language will assist in analysing data. Matlab has high interoperability, allowing for integration with applications using other languages such as python, C++/C and Java. Matlab has amazing visualisation tools.

PyQT is an extension of the existing QT(C++) framework. It utilises python bindings to build cross-platform applications. This framework focuses on building amazing graphical user interfaces (GUI) in python by leveraging the QT toolset.

PyQT key features include:

- Desktop cross-platform GUI Framework (Windows, MacOS, Linux)
- Many existing rich GUI elements
- Supports CSS-like style and theming
- Has an existing QT designer application to use instead of hard coding
- Has multithreading capabilities
- Has openGL support allowing for 3d rendering

Pros	Constraints	Assumptions
- User-friendly built in functions for complex math operations - Matlab has great built-in debugging tools allowing for a better performing functions - Matlab is known for having excellent documentation and a large community	- Matlab requires a paid license - Matlab is considered slower then compiled languages such as C++ for - Matlab is proprietary making it less flexible for open-source projects - PyQT may have some potential licensing issues, due	- Hardware of the students will be in a mid-range bracket - We can utilise the existing Matlab code to build on it and expand it.

- Cross-platform for PyQT - PyQT allows for widget creation - Multithreading support using Qthread -	to the GPL (General public license) or paid license - PyQT and Matlab has a steep learning curve - PyQT can have some Performance Overhead	

Reasoning for solution
The Matlab solution focuses on building on the existing analysis software used. The current solution’s problem is due to the reliance on multiple external applications used in unison. The need for a user-friendly, easy to use GUI is solved by using PyQT instead of clampfit. Using Matlab can achieve great performance with amazing visualisation but it requires tweaks to the code to analyse different recordings of either the same type. This issue is eliminated through the use of python instead of relying on the Matlab application. This integration may slow performance at the expense of allowing for the use of python libraries. Other reasons to suggest this solution include the choice of GUI that allows for modern user-friendly interfaces to be made. This solution takes into account that students may not have strong devices available on hand. We have ensured that users are able to get the best of both worlds in terms of greatly performing analysis with some scalability for more features later. Cobbled into an application with potential to have widgets and a modern UI.

Alternative Solution 2

Solution Overview
For this solution we focus on playing to our strengths as a team, with C++ proficiency. Using R as the backend to analyse the abf files. This solution focuses on the efficiency of C++ through R to push for performance computation with a C++ frontend as well. Using Rinside with R allows for C++ to assist in easing the performance limitations of large datasets.

Languages + Services + Frameworks and APIs
R is a statistical programming language derived off of C++. It is primarily used for data analysis and statistical computing focusing on providing data scientists with rich statistical and mathematical functions. R uses cran to allow developers to access an extensive package ecosystem, extending the capabilities. R is also known for having amazing data visualisation. Similar to Matlab, R has high interoperability, allowing for smooth integration with C/C++, python and java.

As opposed to PyQT, QT is the original frontend framework using C++. QT has cross-platform capabilities extending from desktop to mobile. As a framework, Qt offers a modular architecture that has components ready based on the applications needs. This framework does offer GUI development with both widgets and QML (QT modelling language). Qt also offers extensive 2d and 3d graphical rendering, pushing boundaries of interactive UIs.

Pros	Constraints	Assumptions
- R is open-source and free - Great for statistical analysis - Extensive package ecosystem - Advanced data visualisation - QT allows for modern, user-friendly GUIs to be made. - QT has better graphical performance - Extensive documentation for Qt	- R can struggle with performance limitations - Steep learning curve - R is not versatile as C++ - R has limited builtin GUI support - Qt can be complex to use with a steep learning curve. - Qt has high ram usage - Has licencing costs for commercial products - Qt is considered heavy weight for simple projects.	- Mid-range to strong hardware - We need a heavier frontend framework and backend - Datasets are large and analysis is resource intensive.

Reasoning for solution

Our complete C++ solution comes with a steep learning curve at the expense of a highly powerful backend. C++ allows for programming to be intentional and efficient. Unlike Matlab which is an interpreted language, C++ is a compiled language. This detail alters the performance of the language during runtime by executing differently. Native QT is one of the best performing GUI frameworks available. In comparison to PyQT, the native version is written in complete C++ giving it a steep learning curve added to the R and C++ backend. QT has amazing performance at the expense of having a larger memory footprint. The great benefit of this solution is how efficient and powerful the analysis software can be without any bottleneck. All this performance comes at a cost; difficulty in development, powerful hardware requirements and a steep learning curve.

Alternative Solution 3

Solution Overview

For this solution, we are utilising a full python stack that focuses on high performance and a fantastic GUI. Our scope focuses on creating a better performing analyses software for neurophysiology recording. The other main requirement is to have a user-friendly GUI for students to be able to easily analyse their experiments. The full stack python solution covers both requirements with ease. Python libraries ensure that all backend analysis can be done swiftly and coded with ease as opposed to the current MatLab solution that's being used. The PyQT frontend will be the icing on the cake, allowing for a better user experience as opposed to the difficult to set up and use matlab solution.

Languages + Services + Frameworks and APIs

This solution uses Python as the main language. Our application is localised for individuals to install on their own devices. A big part of the software is allowing for cross-platform desktop capabilities, which both pyQT and standard python backend are. As stated in the previous sentence, pyQT will be used to create a simple user-friendly GUI. Within the backend, we will use pyABF to dissect the ABF files more accurately without the potential loss of metadata by converting it to CSV. We will use neuroDSP to do analysis coupled with other libraries such as; Pandas, NumPy, SciPy, TFTB, PyWavelets.

Pros	Constraints	Assumptions
- Python comes with a high degree of ease of development - Tons of libraries to assist - Free and open source - Full cross-platform capabilities - Very smooth integration between languages	- Memory dense language - Needs to be optimised - GUI performance overhead - Limited parallel execution with heavy CPU tasks - PyQT licensing issues	- Mid-range to strong hardware - We need a heavier frontend framework and backend - Datasets are large and analysis is resource intensive.

Reasoning for solution

The full stack python solution is extremely flexible. Solution 3 is a focus on balance and scalability. The base solution without optimisation will still perform greatly on mid-range systems which most users will have access to. Our choice of Frontend with pyQt will help build amazing modern UIs (User Interface) with a focus on the UX (user experience). Despite the possible performance overhead, optimisation and the use of C++ integrated can alleviate these issues. With this solution, the end product will be able to run better and better as we tinker more. Python provides a high degree of ease of development, this benefit helps push a viable product within the feasible time we have. There is a learning curve for PyQT, but that's the only hurdle before development. The other big issue and main concern is the multithreading limitations, and performance optimisation. Python does have methods to ensure these issues can be minimised at the end.

Additional Features
Potential idea for adding Internet and database. Our application will allow users to make an account and save their experiment logs under their account. This will be an added feature allowing students to be able to access their recordings from any device they need too and anywhere with internet access. This feature will be low priority and added later with the remaining time. (Planned from now, with possible changes). Our stack will rely on filezilla to set up the sql server. To access and create the database we will use Duckdb to query and read the large datasets that will be saved for accounts to access. Finally, we will use parquet to store the experiment results in column-oriented data file format

Alternative Solution Overview

Feature	Solution 1	Solution 2	Solution 3
Ease of development	Medium	Hard	Easy
Performance	High for numerical computations but slower than c++	Very High – R's statistical power paired with	Medium – Python is slower but can be optimised
GUI flexibility	Good – PyQT allows better UI then Matlab	Very good – QT offers modern and high performing UI	Good – PyQT provides clean UI but lacks native QT performance
Licensing	Matlab requires paid licencing	QT has costs for commercial use	Free and open-source
Statistical analysis	Strong – Great signal processing and numerical analysis	Very strong – Built for statistics and C++ integration	Strong – Has powerful libraries
Scalability	Medium – Limited scalability but nice libraries	High – Optimised processing for larger datasets	Medium – Scalable but requires optimisation for larger datasets
Hardware Requirements	Moderate - optimized for mid-range systems	High – Requires high ram and Cpu Power	Moderate – libraries can be memory intensive but are manageable
Integration with existing code	High	Medium	High

Cross-platform Support	Good – supports multiple OS but requires extra steps	Excellent – QT is cross-platform	Excellent – Supports all major OS
Visualisation	Excellent – Matlab has great built in visualisation	Excellent – R has strong data visualisation	Very Good – many visualisation tools
Learning curve	Medium	Steep	Easy

Recommended Solution

After evaluating all of the proposed solutions, utilising a full Python stack with Solution 3 has proven to be the most balanced and effective choice for the project. It prioritises the creation of a user-friendly, intuitive and modern user interface through the use of PyQt, making the software highly accessible to students without prior coding experience. Unlike the MATLAB based or C++/R alternatives, Solution 3 minimises development complexity and enables faster iteration while retaining all core functionality required by the project. Python's extensive library ecosystem, including tools like pyABF, neuroDSP and SciPy supports robust and accurate neurophysiological analysis without the licensing and performance constraints of other solutions.

Lightweight and modern querying will be adopted using DuckDB. It is great for handling and fetching scientific and large datasets. We'll also use Parquet files to store and transfer experiment results in a fast and efficient file format. Finally, tying the two database solutions we will use of FileZilla Server will be to host the database securely, allowing users to access their data from other devices. This setup keeps things flexible and easy to manage, while still being powerful enough for our needs.

Development Release schedule

	Sprint 1		Sprint 2		Sprint 3		Sprint 4		Final week
	31/03/25	07/04/25	14/04/25	21/04/25	28/04/25	05/05/25	12/05/25	19/05/25	26/05/25
Designing stage									
Design Front-end UI	🟩								
Establish UX design and ideology	🟩								
Design Backend Logic and dataflow	🟩								
Development Stage									
Frontend Development	🟩								
Initiate Frontend Development	🟩	🟩							
Create GUI Components		🟩							
Add minimal GUI Functionality for MVP			🟩						
Further Develop Minimal GUI for MVP				🟩					
Conduct internal testing			🟩						
Advance GUI components						🟩			
Finalise and Polish GUI Functionality						🟩			
Conduct internal testing							🟩		
Final GUI created							🟩		
Backend Development	🟩								
Initiate Backend Development	🟩								
Create ABF file loader		🟩							
Plot power spectrum heat map of trace		🟩							
Calculate the Spectrum power density over frequency		🟩							
Adding Database				🟩					
Adding Account system				🟩					
Export results						🟩			
Conduct internal testing				🟩					

Calculate the FFT of Current and Voltage Signals			🟩						
Plot Zap Profile			🟩						
Calculate the resonance Frequency			🟩						
Calculate the HWA (Half-Width at Amplitude)			🟩						
Conduct internal testing						🟩			
Filter for 50/60 Hz noise					🟩				
Segment K+ clearance into 4 episodes					🟩				
Calculate the Exponential Decay (Tau)					🟩				
Calculate the Rate constant (1/tau)					🟩				
Testing Stage									
Perform unit testing							🟩	🟩	🟩
Conduct integration testing							🟩	🟩	🟩
Validate against business requirements							🟩	🟩	🟩
Acceptance testing							🟩	🟩	🟩
Deployment and Closure									
Deploy and handover system									🟩
User Guide								🟩	🟩

COST BENEFIT ANALYSIS

Since the complexity of neurophysiological data has been increasing steadily, the accuracy of calculation is negatively affected and medical professionals confront receiving various false results. As a result, the purpose of this project is to develop the software that can analyse and calculate the data in three methods which are extracellular, resonance frequency and K plus clearance based on the neurophysiological recordings. Moreover, the project plan succeeds in explaining specific tasks of the development, which achieves client’s expectations. Nevertheless, to engage this project, the cost benefit analysis should be reviewed thoroughly because finance plays an important role in this project, which means that the cost of the implementation should be estimated based on current finance. Therefore, it is recommended to explore and acknowledge the benefits of various features such as frontend or backend development in the software, and thus, determine whether they are valuable to be implemented. To estimate the financial cost and evaluate the operational impact of the software development, there are six basic aspects to explain the core of the cost benefit analysis which are software design, software development, testing, maintenance, training, and hardware investments.

PROJECT COSTS:

Software Design Costs

As the client requires the software to be implemented with a user-friendly design, it is important to brainstorm and establish User Interface designs and User Experience designs by creating a number of samples. Team members will subsequently apply User Interface designs and User Experience designs that are the best quality in the project so that users can interact with the software without causing problems. The table below is the cost estimation of each item in software design:

Item Description	**Estimated Cost**
Frontend UI Design	$400
UX Design and Ideology	$400
Backend Logic and Dataflow	$500
Total	$1,300

Software Development Cost

Software Development is the most expensive cost for the project since it contains the core resource of the software to activate its functionalities, including analysis and calculation. This stage includes the cost of frontend development and backend development. In particular, the cost of frontend development involves the implementation of graphical user interface (GUI) components and minimal viable product (MVP) with internal testing to check if false result has occurred. In terms of backend development, the cost includes the implementation of ABF files loader and CSV data conversion, and the calculation of three different methods which are extracellular, resonance frequency and K plus clearance with internal testing as well. In addition, the main platform for the software development focuses on Python since it is known as a programming language that is essentially used for different developments, including software development. The table below will display the cost of frontend and backend development:

Item Description	**Estimated Cost**
Frontend Development	$1,000
Backend Development	$1,000

Python Platform and Libraries	$100
Total	$2,100

Software Testing Cost

This cost represents the time spending of unit testing, system testing, and internal testing. Unit testing focuses on checking individual components or units of code if their functions engage accordingly. On the other hand, system testing evaluates the quality of code in the system, which means that the code structures of calculation and analysis should be operated according to the expectations to deliver the appropriate results. Furthermore, internal testing occurs in the development stage to check functionalities of frontend and backend development before entering the system testing. Moreover, it is advisable to utilise the debug and optimization to detect bugging in the system so that solutions can be suggested punctually to eliminate them before the official release. This table lists the specific costs in testing stage:

Item Description	**Estimated Cost**
Unit Testing	$200
System Testing	$200
Internal Testing	$200
Debug and Optimization	$100
Total	$700

Software Training Cost

To interact with this neurophysiological software, training is planned to help medical professionals to learn how the software is operated in an effective way. The cost of training contains two primary features. The user guides provide instructions and documentations of each functionality in the software to interact with the software. On the other hand, it should provide several practical sessions, including tutorial sessions, to comprehensive the main purpose of each functionality in the software, and thus, get familiar with the software. Moreover, each practical session should be recorded thoroughly and deliver a final report as a hard copy to the client to observe. This clarifies the cost of training by the table below:

Item Description	**Estimated Cost**
User Guide	$100
Practical Sessions (Including Tutorials)	$150

Report and Documentation	$50
Total	$300

Summary of Costs

Project Items	**Estimated Cost**
Software Design	$1,300
Software Development	$2,100
Software Testing	$700
Software Training	$300
Total	$4,400

EXPECTED BENEFITS

Time Savings and Efficiency:

Since a significant amount of time is required to manually calculate brain waves based on neurophysiological recordings, this software will automatically calculate for medical professionals to save time spending, approximately 5 hours per week, which shows the workflow improvement.

High Data Analysis:

The automatic calculation of this software can be performed by approximately 60% of time reduction, which is significantly faster than manual calculation of medical professionals.

High Accuracy:

In addition to its high speed and automatic calculation, this software also succeeds in delivering results with high accuracy, which increases the efficiency of the software.

Usability and Reliability:

Based on the highly accurate result, medical professionals can rely on its calculation for their future projects. Also, a user-friendly software helps users have a positive experience by inputting data and receiving results without complex problems.

The table below demonstrates the value of expected benefits of this software approach to neurophysiological data.

Expected Benefits	**Estimated Value ($AUD per year)**
Time savings and Efficiency	$2,000 per year
High Data Analysis	$1,500 per year
High Accuracy	$1,500 per year
Usability and Reliability	$2,000 per year
Total	$7,000 per year

Conclusion

The project proposal provides a detailed description of the process of the software development. It succeeds in explaining the level of tasks’ priority so that they can be concentrated to complete in a short time. In addition, it can be seen that the explanation of alternative solutions is the large contribution of the project since it shows not only the teamwork and collaboration between each member but also the high productivity of the project, and therefore, enhances the efficiency of the software. Moreover, functional and non-functional requirements assist in describing the process of the software by user stories to understand their functionalities clearly. While the release schedule handles the working process of the software development, the cost benefit analysis focuses on the expenditure of the project and explores the benefits to decide whether it is worthy to implement the software or not. In conclusion, the project proposal is a fundamental instrument not only to outline the requirements for the project but also provide various options for the client so that he can consider thoroughly.