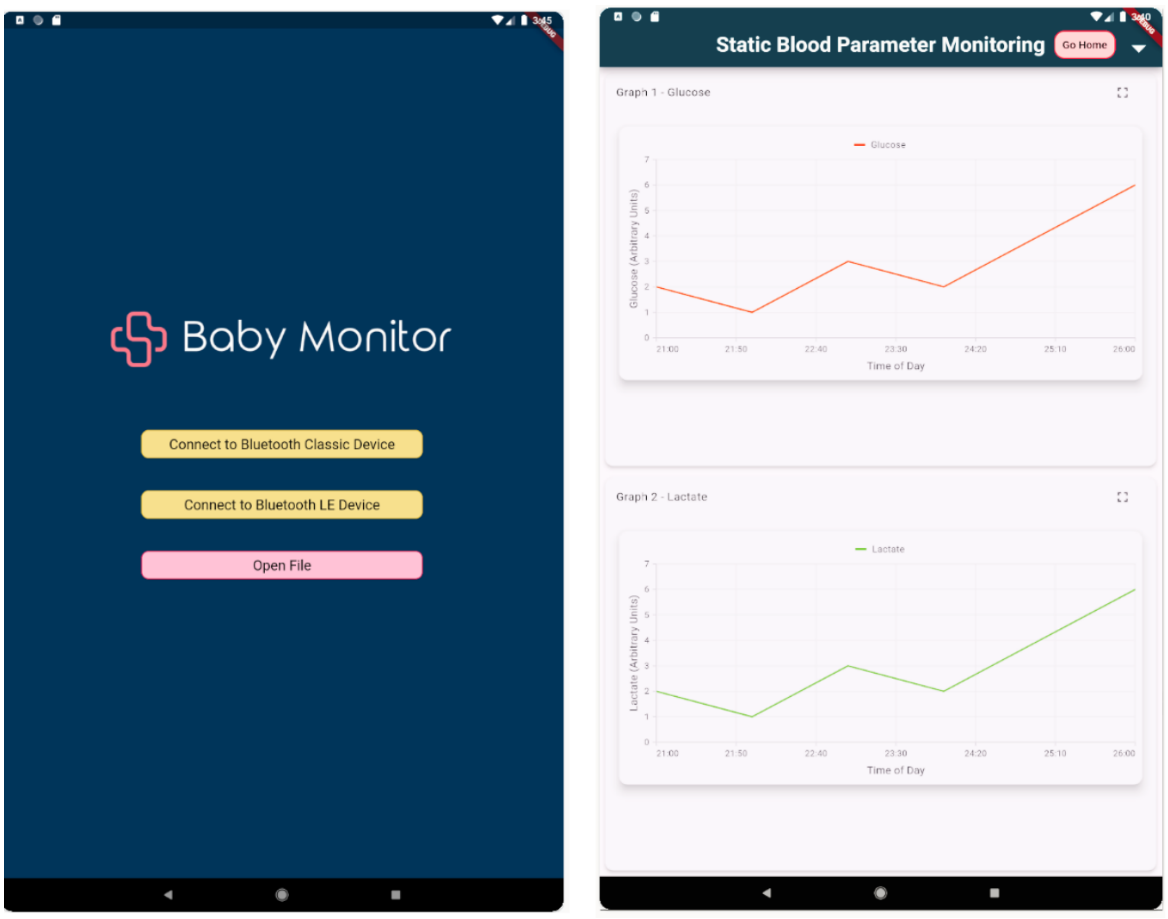
Imperial College London

Premature Baby Monitoring App

Final Report



Baby Monitor
Connect to Bluetooth Classic Device
Connect to Bluetooth LE Device
Open File
Static Blood Parameter Monitoring
Go Home
Graph 1 - Glucose
Glucose
Glucose (Arbitrary Units)
Time of Day
Graph 2 - Lactate
Lactate
Lactate (Arbitrary Units)
Time of Day

Figure 1: Preview of the Baby Monitoring App.

Client: Prof. Martyn Boutelle, Dr. Sally Gowers

Project: BIOE60010 - Software Engineering for Bioengineers

Date: 14th January, 2025

Authors: Edward Goh, Eu Wayne Wong, Joshua DeCoteau, Mael Scott de Martinville, Violaine de Saint-Quentin

GitHub Repository

Department of Biomedical Engineering

Imperial College London

Contents

1 Introduction

Newborns transition from depending on maternal glucose via the placenta to self-regulating glucose production after birth [1]. Hypoglycemia is the most prevalent metabolic issue in neonates, with no universally agreed safe glucose threshold, partly due to uncertainties about its effects on neurodevelopment [2]. Traditionally, glucose levels are tested by blood samples, often via finger pricks, which are challenging and distressing for premature infants and only provide intermittent insights.

Continuous glucose monitoring offers real-time insights into blood glucose levels, highlighting patterns, duration, and severity of hypoglycemia. Additionally, measuring lactate – a byproduct of glucose metabolism – furthers the understanding of the neonates metabolic health [3].

To facilitate continuous monitoring, Professor Martyn Boutelle and his team developed a non-invasive device that monitors blood parameters through skin contact. However, to effectively analyse and utilise the collected data, a specialized application is needed to connect to the devices, displaying data both in real-time and post-measurement. This report outlines the development of such an application.

2 Implementation Strategy

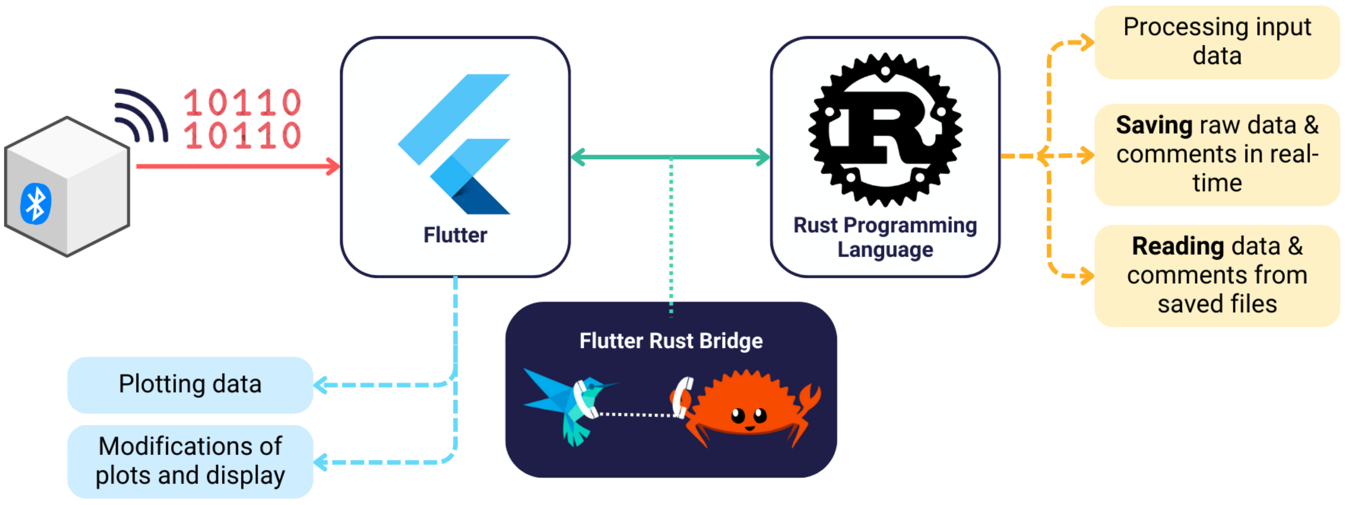

10110
10110
Flutter
Rust Programming Language
Processing input data
Saving raw data & comments in real-time
Reading data & comments from saved files
Flutter Rust Bridge
Plotting data
Modifications of plots and display

Figure 2: Overview of the implementation strategy used for the mobile application.

2.1 Frontend Development: Flutter

The frontend of the application was developed using Flutter, an open-source UI (User Interface) software development kit (SDK) provided by Google. This SDK was chosen for its ability to create scalable and adaptable user interfaces that remain consistent across various screen sizes. Flutter's widget-based design facilitates an object-oriented approach that improves the readability and scalability of the code. Additionally, the extensive community support and availability of packages for real-time plotting and Bluetooth integration enhance the development experience and functionality.

2.2 Backend Development: Rust

For the backend, Rust was selected due to its performance efficiencies, particularly in type safety, memory safety, and concurrency. Rust is particularly suited to handle real-time data processing requirements, such as managing multiple channels concurrently and maintaining high-speed operations. This ensures efficient processing of data transmitted from the frontend. Rust's asynchronous programming capabilities further optimise performance by reducing overhead during data communication between the frontend and backend.

2.3 Integration of Frontend and Backend: Flutter-Rust Bridge

Integration between the frontend and backend is achieved through the Flutter Rust Bridge package, which facilitates the communication of data and operational synchronicity between the Flutter framework and the Rust language. This integration ensures that data handling tasks remain concurrent and efficient, supporting real-time data processing and user interaction.

2.4 Bluetooth Functionality

Bluetooth operations are managed within the Flutter framework, enabling the application to support a variety of Bluetooth devices, including those using both low energy and classic Bluetooth protocols. The application allows the user to select and connect to a target device directly from the interface, after which the Flutter framework manages the data transmission to the Rust-based backend for processing.

2.5 Development Tools and Testing

Visual Studio Code (VS Code) was chosen for the development environment as it supports both Flutter and Rust through various extensions and plugins, simplifying project and code management. An Android device emulator from Android Studio was used to test the application by providing a controlled environment for debugging and performance assessment.

A summary of the implementation strategy is presented in Fig.2.

3 Design Documents

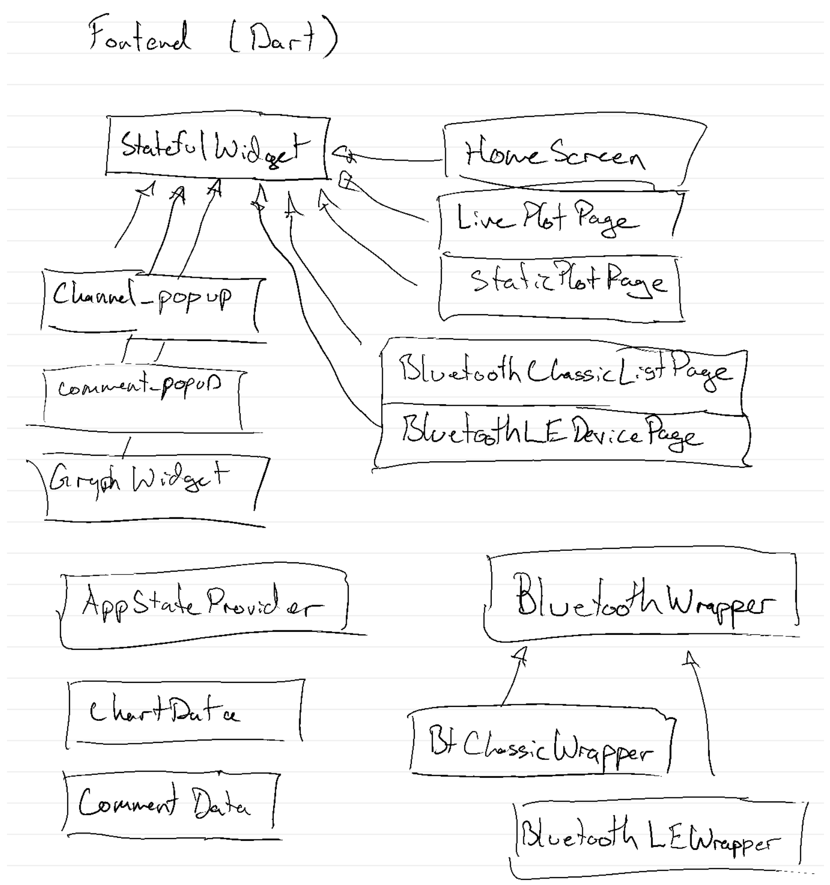


Figure 3: Class diagram frontend.

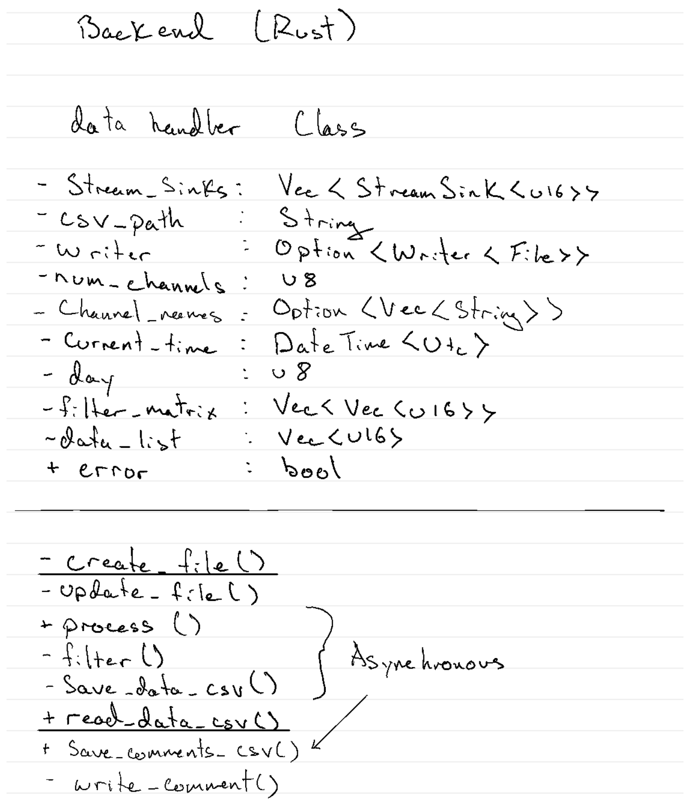
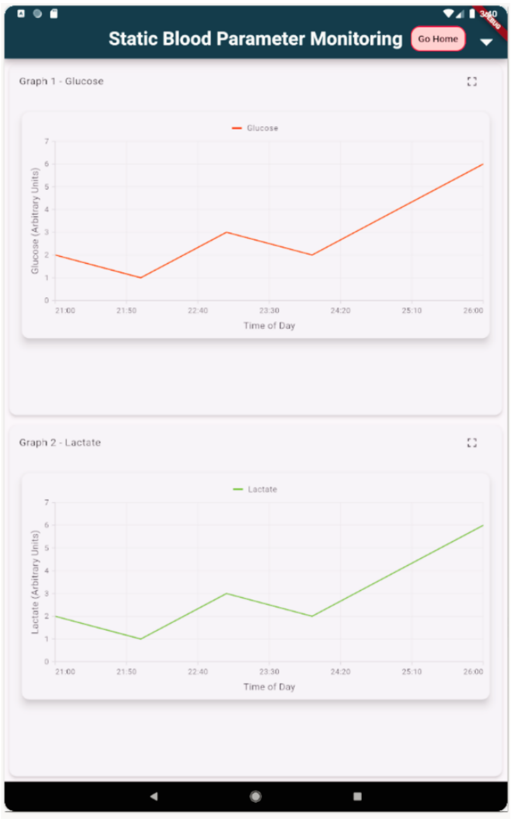


Figure 4: Class diagram backend.

4 User Interface and Layout


Static Blood Parameter Monitoring
Go Home
Graph 1 - Glucose
Glucose
Glucose (Arbitrary Units)
Time of Day
Graph 2 - Lactate
Lactate
Lactate (Arbitrary Units)
Time of Day

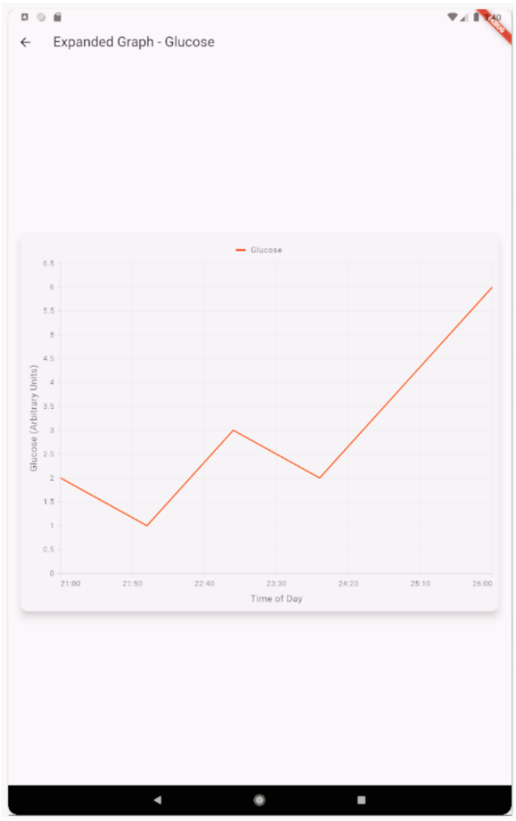
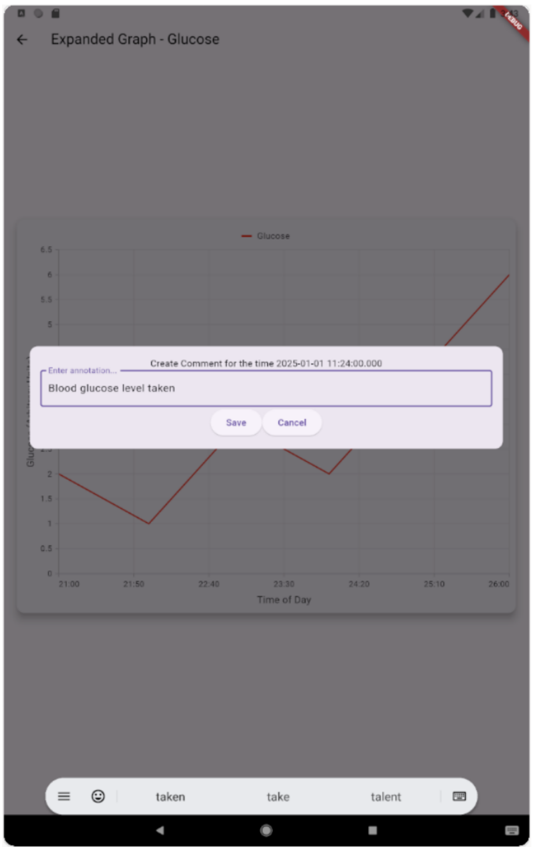


Figure 5: Screen capture of UI interface to the static plots for blood glucose and lactate levels (left) and the expanded view of the glucose plot (right).



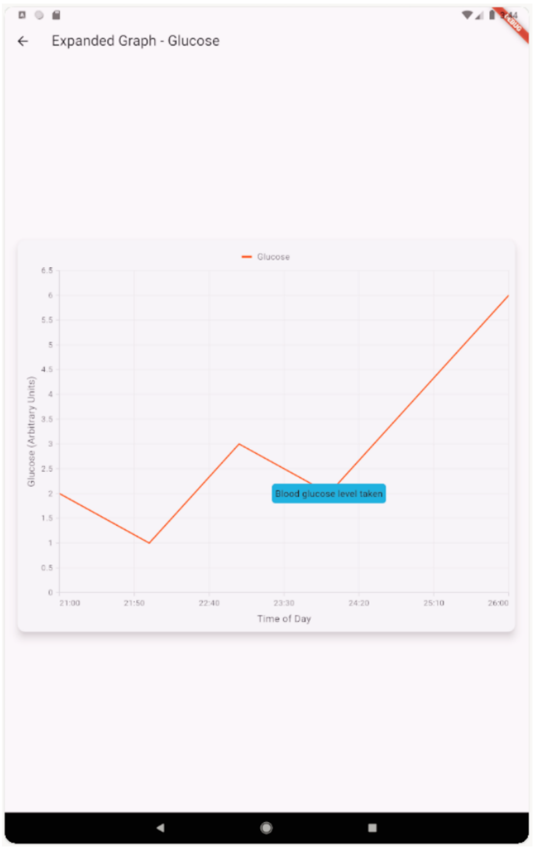
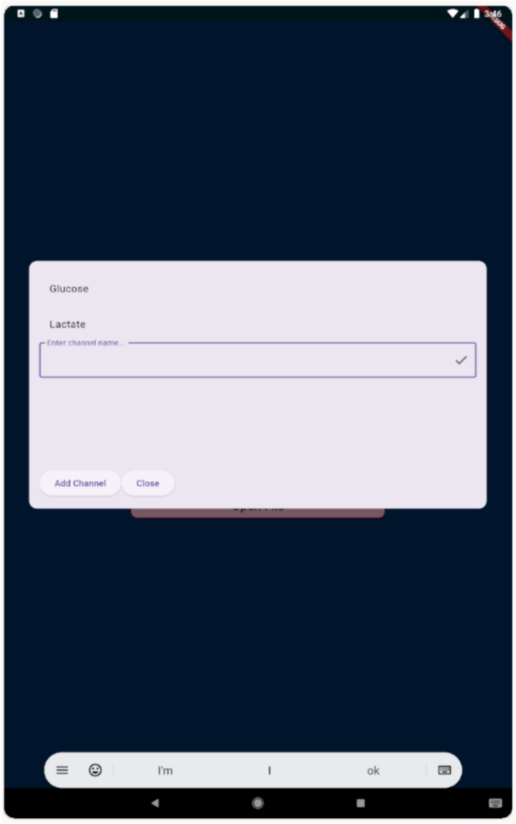


Figure 6: Screen capture of UI interface illustrating the commenting feature.



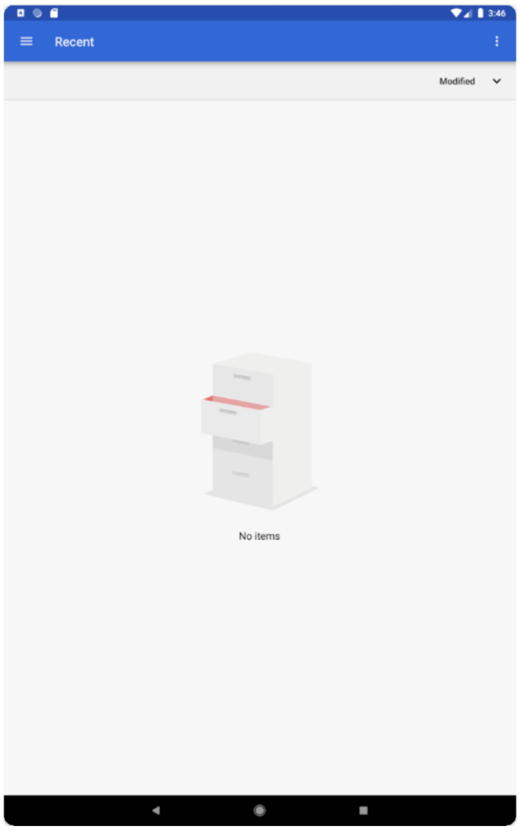


Figure 7: Screen capture of UI interface illustrating the ability to add a channel (left) and access to files from previous recording (right).

5 Evaluation of User Requirements Fulfillment

Table 1: Evaluation of user requirements

	User requirement	Fulfilled
1	Operates on Android devices.	Yes
2	Quick and simple device detection.	Not tested
3	Easy device selection in an environment with multiple active devices.	Not tested
4	Receives and stores data every 0.1s, for every channel.	Yes
5	Plots one data point per second, for every channel (Fig.5).	Yes
6	The software operates continuously for extended periods of time.	Not tested
7	The glucose trace is red (Fig.5).	Yes
8	The lactate trace is green (Fig.5).	Yes
9	The x-axis indicates the time of day (Fig.5).	Yes
10	Plots can be made static.	Yes
11	The user can scroll in time on the plots.	Yes
12	The user can zoom in and out on the plots.	Yes
13	The user can expand a chosen plot in a new window (Fig.5).	Yes
14	The user can add annotations to the plots (Fig.6).	Yes
15	Flexible UI capable of handling multiple plots on one page (Fig.7).	Yes
16	The raw real-time data is stored in a .CSV file, with a new file per day.	Yes
17	The added text comments are saved in the .CSV file at the corresponding times.	Yes
18	Efficient data storage and memory usage.	Yes

6 Team Member Contributions

Edward Goh

- Conducted a thorough review of the previous codebase to understand its intended functionality, identified key areas of failure, and implemented improvements to address these shortcomings in the new version.
- Designed and implemented a robust loop logic system that dynamically handles and presents multiple channels as graphs, enabling the app to scale efficiently with user-defined parameters.
- Assisted in the development of the live plot page by integrating dynamic data handling, ensuring real-time graph updates and maintaining UI consistency.
- Developed the initial Bluetooth connection page to facilitate device pairing and data transfer, later adapting the approach to create separate pages for Bluetooth Classic and Bluetooth Low Energy (BLE) to accommodate evolving project needs.
- Established a structured Git branching workflow, enabling the team to collaborate more effectively, maintaining a clean codebase whilst managing multiple features simultaneously without conflicts.

Eu Wayne Wong

- Reverse-engineered the existing application to determine the structure and data type of the incoming data stream from the Bluetooth device.
- In collaboration with Josh, developed the DataHandler class in the backend (Rust). In the DataHandler class, the following processes were constructed by Wayne:
  - Converts the incoming data stream (bytes) into the channel values (u16 integers) in real-time and asynchronously.
  - Writes the channel measurements into a .CSV file with their respective timestamps in real-time and asynchronously.
  - Automatically creates a new .CSV file every 24 hours to prevent overloading.
  - Creates a joint comments .CSV file and writes any annotations made into the file in real-time.
  - Implemented a real-time moving average filter on the channel values.
  - Utilized StreamSinks to transmit data to Flutter in real-time and asynchronously.
  - Extracts the channel measurements, timestamps, and comments and relays them to Flutter for plotting.
- Created unit tests in the frontend (Dart) for fullscreen, scrolling, zooming, commenting, and the DataHandler Class.

Joshua DeCoteau

- Interfaced the Rust backend and the Flutter frontend.
- Ensured that the Flutter frontend was adaptable to different numbers of data channels.
- Leveraged the functionality of the RustStreamSink class to send data from Rust and display it in Flutter seamlessly.
- Supported the design of graph widgets that are adaptable to both static and live updating data.
- Built the system that manages the app state. This ensures that vital data such as the saved graph data, the data handler for live plotting, the names of each channel and the comments added to each graph can be easily shared among multiple different widgets and different routes of the app.
- Created the pages to discover both Bluetooth classic and Bluetooth low energy devices meaning that the user is not limited to one type of device when using the app.
- Set up the Bluetooth connection so that data is automatically streamed and handled in Rust where it is then sent to the live plots to be updated.

Mael Scott de Martinville

- Interpreted user requirements to contribute to the front end by building the skeleton structure for the data visualization system.
- Implemented real-time plotting of multiple measurements, incorporating interactive features such as zooming and panning to enhance data exploration, using Syncfusion charts.
- Designed and structured the front end with a modular approach, ensuring clear separation between widgets and screens. Navigation was managed using named routes for seamless transitions between device selection, data input, and graph visualization pages.
- Assisted in reverse-engineering the previous application to identify the structure and data type of the input stream, ensuring compatibility with the current system.

- Contributed to the integration of the Flutter front end with the Rust backend, enabling seamless real-time data visualisation.

Violaine de Saint-Quentin

- Initially heavily engaged with the requirements gathering phase and identifying user requirements which were summarised in an early/mid-project presentation.
- Focused on the front end to develop an adaptable graph widget with interactive features, including zooming and scrolling in time, to be used repeatedly for plotting different blood parameters.
- Restructured the visual layout of the pages to improve usability.
- Improved the design and appearance of the pages, playing with colors, shapes, and buttons to enhance the user experience.
- Responsible for the writing, structuring, and curation of the report, with inputs from team members (and with the exception of reflections).

7 Team Member Reflections

Critically reflect on your own understanding and direct independent learning.

Edward Goh
I developed an interest in software engineering and decided to pursue this course to explore the field further. Throughout the project, I came to realise that one of the most challenging yet rewarding aspects of software engineering is collaboration – working effectively as a team and managing multiple Git branches. This experience highlighted to me the importance of excellent communication, and I am truly grateful for having knowledgeable and supportive teammates who took the time to explain complex concepts to me.
At the start of the project, I was very excited as I had never worked on a real-world project from scratch before. The process, from setting up Git branches to observing how my collegue Josh configured the working environment, was an eye-opening experience. Having the flexibility to initially just watch and learn was extremely valuable and led me to reflect on how much in the field is still open to learn.
During the project, I had to learn two new programming languages: Rust and Dart. While my understanding of Rust is still at a foundational level, I worked primarily with Dart through the Flutter framework. I found Flutter's ability to create visually appealing interfaces fascinating. Beginning with a basic Flutter project was a great way to familiarise myself with the language while reinforcing the effectiveness of working on small, focused projects to learn new languages and frameworks.
This project not only improved my coding skills but also deepened my understanding of the entire software development lifecycle. Importantly, it taught me how to work effectively in a team, collaborate efficiently, and contribute to delivering a functional application. Reflecting on our progress, I am proud of the work we accomplished, especially the development of an app that has the potential to improve healthcare for premature babies.

Eu Wayne Wong
Having completed multiple individual programming projects in the past, I was fairly confident in my technical skills. For this project, I had to learn both Rust and Dart. I always aim to understand the fundamentals of programming languages before diving deeper into more specialised areas relevant to my needs. This approach enables me to write code independently with minimal reliance on AI tools, such as ChatGPT.
This project allowed me to develop a solid foundation in using Git and operating GitHub. Beyond technical skills, it introduced me to the complex challenges that arise when working in a collaborative environment. With multiple minds working together, we had to ensure everyone remained aligned and built their code with consideration for others' work. This experience and the challenges encountered taught me the importance of clear communication and collaboration in software engineering projects.

Writing readable code and providing meaningful comments were crucial skills that I had to hone throughout the project. Reflecting on previous experiences, I am conscious that the lack of clear comments or understandable code would likely lead to unnecessary confusion, both for my teammates and my future self. To mitigate this, I made a deliberate effort to structure my code clearly and to comment it meticulously.
Lastly, I was also responsible for writing the unit tests, which was an aspect I had not considered in previous projects but have come to understand their usefulness.
In summary, this project has been an invaluable learning experience. I am confident that the skills I've developed – particularly in collaboration, code readability, and testing – will be extremely useful for future projects. In programming, there are always more features to implement or a more efficient way of coding, and there is potential for many improvements to be made in this project. However, I am proud of what we have developed and I hope this application is capable of assisting Prof. Boutelle and Dr. Sally Gowers with their research project. I am eager to continue refining the app with them while further developing my technical expertise in Rust, Dart, and Git.

Joshua DeCoteau
I have previous experience building biomedical applications, such as those used for cardiac electrogram data analysis. However, this project marked my first time developing an application of this scale and collaborating in a team on a programming project. One of the initial challenges I encountered was using Git as an effective method of collaboration. Having typically worked independently, I was used to managing my work on my computer without the need for coordinating changes with others. Over time, I built a strong Git workflow to ensure my work remained up to date through regular commits. However, I recognise that there was room for improvement, especially in how I structure my commit messages to ensure clearer communication with my teammates about the changes made.
Through this project, I was able to gain a strong understanding of application development. In particular, I learned how to effectively manage the state of a large-scale application. One key challenge was ensuring that widgets and classes always had the correct data, which was both time-consuming and complex. I solved this problem by creating one class that stores all the necessary data for the application to function and is available to every other class in the application. This was a differed from my previous work, where I typically designed applications within a single class. The complexity of working with multiple classes and managing various layers of abstraction and encapsulation was initially difficult for me. However, once I took the time to understand how everything fit together, I found it both rewarding and enjoyable to write my own classes to structure the project.
Ultimately, I came to see object-oriented programming as an extremely efficient and scalable method for application development, especially for larger projects like this one. In the future, I plan to continue improving my skills with Git and object-oriented programming to create more scalable and maintainable applications. Reflecting on this experience, I am proud of the work I accomplished and the skills I developed in Flutter, large-scale application design, and Git.

Mael Scott de Martinville
Having not previously worked on front-end development, I relied on my adaptability and curiosity to contribute effectively to this application's front end. This project required that I learn Flutter, including Dart programming and its widget-based UI structure. I focused on mastering core principles such as widget composition, state management, and UI navigation to ensure I could develop and modify the codebase effectively. While my goal was to write code independently, I occasionally referenced AI tools to accelerate the learning process.
Working within a team emphasised the importance of maintaining a consistent code structure. We had to ensure that the various screens and widgets were seamlessly integrated, particularly when linking the Flutter front end with the Rust back end for real-time data visualisation. This experience and the challenges encountered underscored to me the value of clear communication and collaborative practices. Furthermore, working with Git strengthened my understanding of branch management and merge conflict resolution, while also encouraging better communication practices within the team. Through the progress of the project, the importance of clear documentation and modular code design was constantly reinforced – a lesson which will undoubtedly benefit me in future collaborative projects.

Violaine de Saint-Quentin
Unlike most of my team, I initially lacked confidence in my ability to contribute meaningfully to this project. Their advanced skill sets and extensive experience felt intimidating. However, stepping far outside my comfort zone ultimately became a valuable opportunity to grow and learn from my peers, many of whom may have once been in the same position.
Rather than rushing into the project as I usually would, listening to my team, I recognised the importance of building a solid foundation first. I began by familiarising myself with Flutter, learning Dart programming, and exploring the widgets and functionalities, even if they didn't necessarily fall under the scope of this project. The seemingly straightforward task of setting up the Flutter environment proved to be more complex than expected. Yet, this initial challenge turned into a learning experience, exposing me to unfamiliar parts of my operating system, along with new packages, plug-ins, and extensions that I had never encountered before. Working through this challenge step-by-step gave me insights I am confident will be invaluable for future projects while also enhancing my ability to use community forums such as Stack Overflow.
Once set up, I genuinely enjoyed working with Flutter. The customisable widgets felt intuitive, and as someone who is naturally drawn to visual aesthetics, I found great satisfaction in creating designs that were not only functional but also user-friendly and visually-appealing. This focus on UI/UX was reinforced during conversations with our client, who pointed out small yet impactful design flaws. These discussions deepened my belief that an application's visual appeal and ease-of-use are crucial to its success. Nevertheless, I was cautious about prioritising aesthetics at the expense of functionality and I focused on ensuring that my designs were adaptable and compatible with my teammates' work.
This project was my first experience working on a collaborative software effort, and my prior use of GitHub had been limited to individual work. Learning to manage branches, handle merges, resolve conflicts, and navigate shared workflows was more challenging than I anticipated. I felt a strong sense of responsibility not to disrupt or overwrite my teammates' work, revealing to me the subtleties and intricacies of project management and organising in software enginering.
These challenges highlighted to me the need to be more patient and resilient. I realized that things don't always work perfectly the first time, and unexpected glitches are part of the process. Each difficulty I encountered not only helped me develop new technical skills but also shifted my perspective on collaboration and problem-solving. I came to understand that growth comes from embracing discomfort and learning from every experience, and I truly appreciate the support and learning from my colleagues throughout this process.

References

[1] W. W. Hay and P. J. Rozance, "Continuous Glucose Monitoring for Diagnosis and Treatment of Neonatal Hypoglycemia" The Journal of Pediatrics, vol. 157, (2), pp. 180-182, 2010. Available from: https://www.sciencedirect.com/science/article/pii/S0022347610003070.DOI:10.1016/j.jpeds.2010.04.007.

[2] R. Shah et al, "Neonatal Glycaemia and Neurodevelopmental Outcomes: A Systematic Review and Meta-Analysis," Neonatology, vol. 115, (2), pp. 116-126, 2019. Available from: https://pubmed.ncbi.nlm.nih.gov/30408811/.DOI:10.1159/000492859.

[3] X. Li et al, "Lactate metabolism in human health and disease," Sig Transduct Target Ther, vol. 7, (1), pp. 1-22, 2022. Available from: https://www.nature.com/articles/s41392-022-01151-3.DOI:10.1038/s41392-022-01151-3.

NOTE 1: ChatGPT was used throughout the project as a tool to help understand complex technical information, for guidance regarding software development, and understanding errors. It was also used to help fix any formatting issues encountered when writing this report.

NOTE 2: The Google Developer's Codelab "Your first Flutter app" was used by multiple team members to initially learn Flutter.