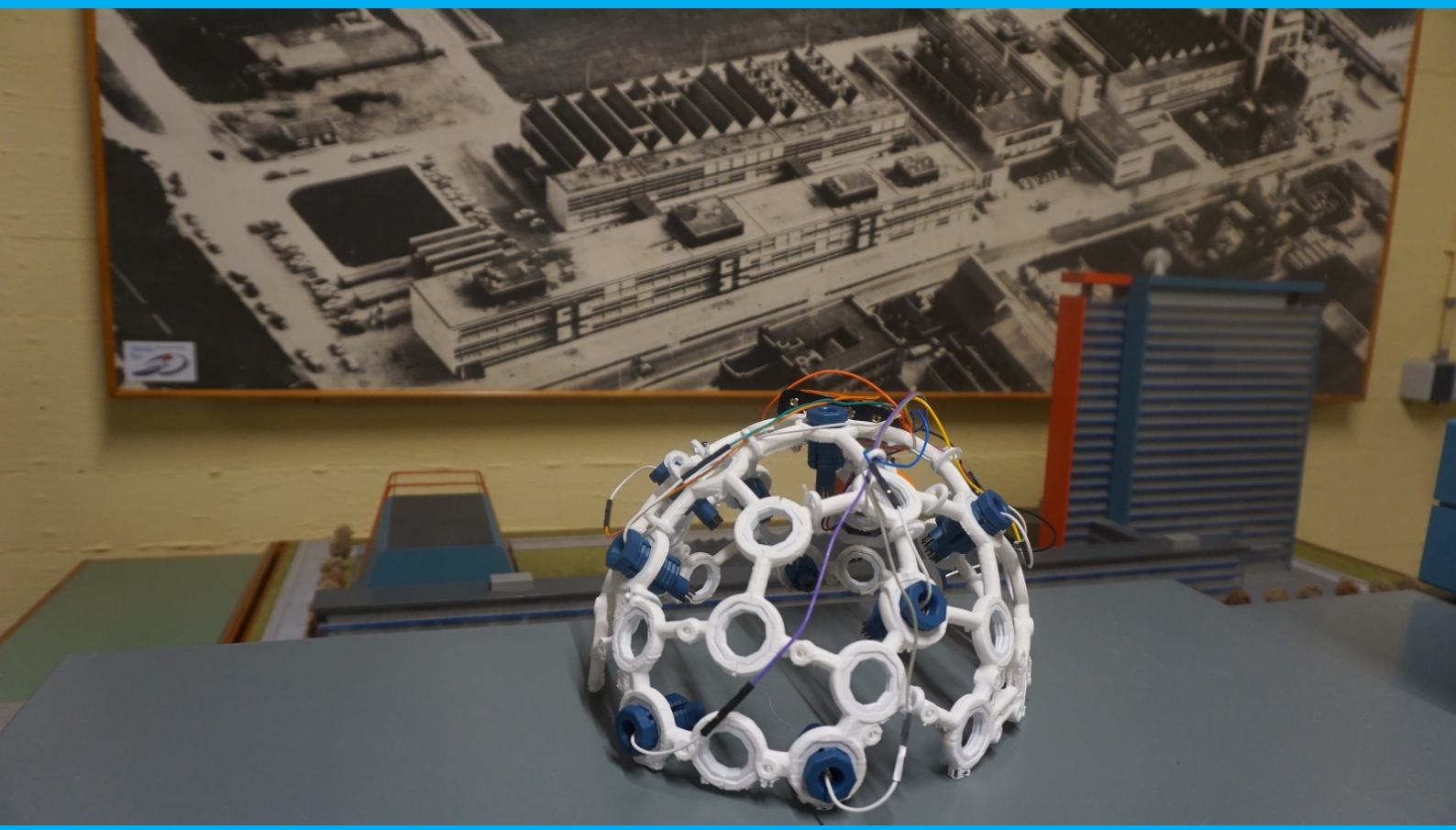
EEG-based Computer Interface

Group: A.03

June 29, 2024

Adnane Acudad 5565499
Oussama Seddouki 4608690



Abstract

The purpose of this thesis is to study the possibility of developing a computer interface that makes use of electroencephalogram (EEG) signals in order to improve the interaction between humans and computers. The major objective is to develop a system that is able to read brain activity in real time and then transform that information into instructions that can be executed on a computer. This study combines electroencephalogram (EEG) technology with sophisticated machine learning algorithms in order to develop an interface that is fluid, responsive, and user-friendly. Among the most important goals are the development of methods for robust signal processing, the design of an interface that is easy to use, and the installation of tools for real-time data presentation and analysis. A comprehensive set of tests was performed on the system, which revealed considerable improvements in terms of accuracy, responsiveness, and the overall user experience. This research makes a significant contribution to the expanding field of neurotechnology by providing an interface that is simple to use and may be used as an inspiration for a broad variety of purposes, including medical diagnosis and treatment, as well as entertainment and other games.

Contents

1 Introduction

1.1. Background

Electroencephalography (EEG) is a non-invasive method for recording the electrical activity of the brain through electrodes placed on the scalp. Since its development, EEG has been fundamental in neuroscience and clinical diagnostics due to its ability to provide real-time monitoring of brain function. Recently, EEG's application in brain-computer interfaces (BCIs) has gained significant attention [1]. BCIs facilitate direct communication between the brain and external devices, enabling various applications such as medical rehabilitation, neuroprosthetics, and gaming.

1.2. Problem Statement

Despite the advances in EEG technology and machine learning, creating user-friendly and accessible interfaces for EEG-based applications remains a challenge. Current systems often lack the integration and real-time capabilities needed for practical use, especially in multi-user environments [2]. This thesis aims to address these challenges by developing a comprehensive EEG interface that facilitates real-time data streaming, integrates with a machine learning model, and supports interactive applications such as gaming.

1.3. Objectives

The primary objective of this thesis is to design and develop an accessible EEG interface that supports multiple users and allows for real-time observation and interaction with EEG data. Specifically, the interface will:

- Stream EEG data in real-time for monitoring and analysis.
- Integrate with a machine learning model to interpret EEG signals for left or right thinking.
- Include a training module to improve the accuracy of the machine learning model for specific users and the baseline model.
- Feature two games that utilize EEG data to demonstrate practical application.

1.4. Thesis Structure

This thesis is structured as follows:

- Chapter 2 : Programme of Requirements: Outlines the objectives and scope of the thesis, detailing the specific requirements and constraints for the EEG interface system.
- Chapter 3: Literature Review: Reviews relevant literature on EEG technology, machine learning applications for EEG, and existing EEG interfaces.
- Chapter 4: System Design: Details the design and architecture of the EEG interface, including requirements analysis and data flow.

- Chapter 5: Interface Development: Discusses the development process, including frontend and backend implementation, and real-time data streaming techniques.
- Chapter 6: Integration with Machine Learning Model: Explains how the interface interacts with the machine learning model and the training module.
- Chapter 7: Game Development: Describes the design and implementation of the EEG-based game.
- Chapter 8: Testing and Evaluation: Presents the testing methodology, results, and analysis of the system's performance with respect to the Programme of Requirements.
- Chapter 9: Conclusion: Summarizes the contributions of the thesis, discusses limitations, and suggests future research directions.

2

Programme of Requirements

This chapter outlines the objectives and scope of the thesis, detailing the specific requirements and constraints for the EEG interface system. These requirements are divided into general specifications and specific requirements for the Interface Group.

2.1. General Specifications

The general market focus of this project is on gaming, with a future focus on medical applications. In a 43-person user study that evaluates the impact of system latencies in gaming it showed that user performance improved for the smallest reductions in latency [3]. It is therefore fundamental for the delay in the general specifications to be as minimal, but also as realistic as possible. After initial testing one requirement is that the Machine Learning model will look at samples of 0.5 seconds (1 epoch = 0.5 seconds) and the classification of these samples should take a maximum of 0.15 seconds. The total delay should therefore be a maximum of 0.75 seconds. This is because the delay to display the decision made by the Machine Learning algorithm on the interface will take a maximum of 0.10 seconds.An extra 0.10 seconds is added to make room for performance errors (This can be caused by the computer if the program just started, because it is not yet in the caches but in memory). Additionally, the classification accuracy must be at least 75%, which is considered acceptable for early-stage technology. Machine learning and streaming must be implemented in Python to ensure easy compatibility across different systems.

2.2. Interface Group Specifications

The interface group's primary objective is to develop a functional and user-friendly interface that integrates with other system components, providing continuous EEG data visualization with a maximum delay of 0.25 seconds. The maximum time of 0.75 seconds mentioned in section 2.1 is independent of these 0.25 seconds as this is for EEG data visualization and has therefore nothing to do with the game interaction and the ML model. The user interface should feature a simple layout with clear navigation, including four main options: Main Menu, Calibration, Training, and Play Game.

2.2.1. Real-time data visualization

For real-time data visualization, the interface must plot the obtained EEG data to users with a maximum latency of 0.25 seconds. As the delay is a maximum of one-fourth of a second it can still be considered real-time. This criterion of EEG visualization is crucial for providing immediate insights to users into the EEG data being recorded. The game control integration should include a basic game where an object moves left or right based on user choices. Also, it should be accounted for that because there is a delay of a maximum of 0.75 seconds for left and right the game should be adjusted such that the game is playable with the mentioned delay. This aspect of the interface highlights the practical application of the EEG data in a gaming context.

2.2.2. Training Module

The training module within the interface should help users train their EEG responses, providing feedback and progress tracking for user training sessions. This feature is essential for improving the accuracy and reliability

of the EEG data interpretation. Within the training module users should have the chance to train the left, right movement or both movements.

2.2.3. Integration

The interface must be compatible with the data collection and machine learning modules, enabling data flow between the EEG data acquisition, preprocessing, and machine learning classification. User-specific customization should allow for the creation of user-specific models, improving classification accuracy for each user and ensuring the interface can adapt to individual user profiles and preferences.

2.2.4. Final Goal

The ultimate goal of this project is to create a motor execution-based BCI in a very simple form that allows the user to control a game with movements of their hands. The game will be made simple to accept only left and right inputs and will be implemented in control software where calibration, user setup, and profiles can be managed. The games will be simplified versions of the popular games 'Space Shooter' and 'Pong'.

3

Literature Review

This chapter provides a review of the existing literature on the use of computer interfaces in training machine learning models, particularly in the field of electroencephalography (EEG). The review highlights the significance of well-structured datasets, real-time data analysis, neurofeedback mechanisms, efficient data stream processing, and robust frameworks and technology stacks in developing and enhancing brain-computer interfaces (BCIs). Each section delves into key studies and findings that contribute to the development of these interfaces, offering insights into the current state of research and identifying areas for future exploration.

3.1. Datasets

In the development of computer interfaces training machine learning models and for the acquisition of brain data well-structured datasets are of utmost importance. Within this context, the work of Nogales and García-Tejedors is particularly relevant. Their review of open EEG datasets and their research in deep learning models marks how important high-quality data is for the development of good neural networks. [4].

3.2. Real-time Analyses

Real-time analytics is very important when machine learning models must perform with high accuracy and low latency. Lim illustrates this through his work on a real-time machine learning model that predicts short-term mortality in critically ill patients. This model's development demonstrates the interface's capability of handling in high-pressure and high-demanding environments with high accuracy and low latency.[5].
Real-time EEG data processing is crucial for numerous applications in neuroscience and brain-computer interfaces. In their paper titled "Real-Time EEG Data Processing Using Independent Component Analysis (ICA)", they propose a real-time processing system based on Independent Component Analysis (ICA) for EEG signals. They initially test and simulate the algorithm offline to verify its functionality and performance. Subsequently, they model a Simulink discrete real-time design based on the processing algorithm. The authors then discuss the system's limitations and various options for translating the designed Simulink model into a real-time processing system. This study provides valuable insights into real-time EEG signal processing and offers a framework for implementing such systems, with potential applications in neuroscience research and clinical settings [6].

3.3. Neurofeedback EEG

In the field of brain-computer interfaces (BCIs), EEG-based neurofeedback has shown promise for both emotion regulation and motor control. Chen introduced a real-time EEG-based BCI system for emotion regulation, demonstrating the effectiveness of machine learning algorithms in analyzing EEG signals to provide real-time feedback. Although their focus was on emotion regulation, the underlying principles of EEG signal analysis and neurofeedback are directly applicable to movement prediction. Studies such as those by Samosir have successfully implemented machine learning models to decode motor intentions from EEG signals, enabling accurate prediction of left and right hand movements [7]. By training these models on labeled EEG data, where the intended movement is known, they can predict movements in real-time, illustrating the potential of this approach for our research.

Building on these methodologies, our approach leverages real-time EEG data analysis to predict left and right movements. Research by Nogales [8] supports the use of specific EEG frequency bands to differentiate motor activities, highlighting spectral features associated with particular movements.

3.4. Frameworks

The integration of Apache Flink, a framework for handling real-time data feeds, is equally crucial for developing scalable and fault-tolerant systems to manage data streams efficiently. Flink's ability to provide high-throughput and low-latency processing makes it an excellent choice for applications that depend on the immediate processing of large volumes of data, such as those analyzing EEG signals in real time. By utilizing Flink in combination with the findings of Samosir , developers can enhance the reliability and performance of brain-computer interfaces [9].

Furthermore, Mohedano-Munoz discusses a streaming data visualization framework designed specifically for intensive care units. This framework supports decision-making by providing real-time visualizations of streaming data, an approach that can be adapted for brain data visualization to help in immediate interpretation and response [10].

While alternative technologies like Apache Flink and the one mentioned in Mohedano-Munoz's paper offer scalability benefits , they introduce higher latency due to their processing overhead [9] . WebSocket protocol, on the other hand, facilitates direct communication between client and server, enabling faster data transmission without compromising on latency [11]. This decision aligns with the need for real-time capabilities, especially in multi-user environments.

4

System Design

This chapter outlines the comprehensive design of our system interface, detailing its key components and how they interact. The system is structured to manage user interactions, EEG signal calibration, command training, and interactive gameplay experiences. Figure 4.1 illustrates a block diagram demonstrating how these components integrate and work together.The main starting point of the system is the Menu. It acts as the central hub, allowing users to move between different sections. The following sections explain the main parts and features of the system. The Menu is the first screen users see when the system is started. From here, users can select, create and delete users. The system can handle up to six users, who can be chosen or added from this section. It also allows users to navigate to other sections. In the beginning, we started by sketching out our dashboard as shown in figure **??**, which helped us plan how everything would look and work. Our goal was to make sure it's easy for users to find what they need and navigate through the system smoothly.

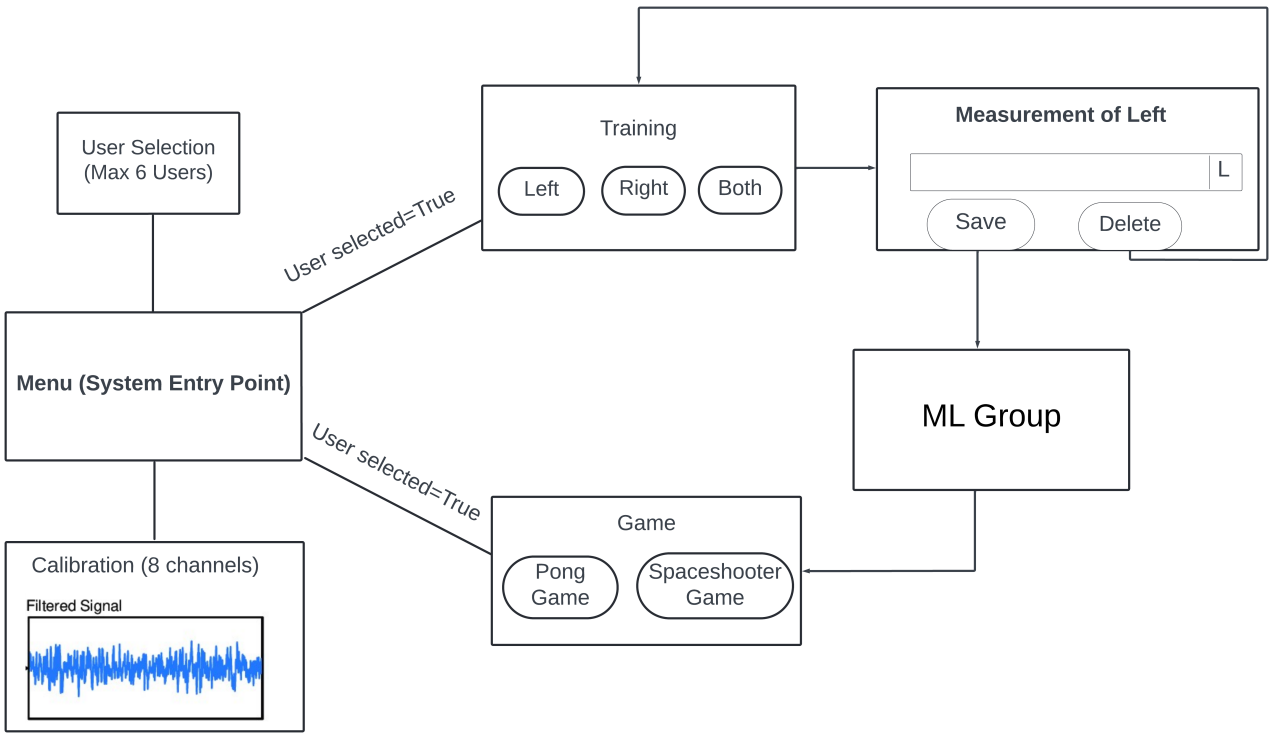


Figure 4.1: System Block Diagram

The Calibration Section is essential for making sure the EEG signals are accurate and reliable. It shows real-time data from the EEG headset's eight channels, helping to confirm that everything is working correctly and the signals are being received accurately. Using WebSocket technology, it connects the Python backend and

the JavaScript frontend, allowing users to see their EEG data in real time. This helps users check the quality of each EEG channel, ensuring the electrodes are connected and working properly. Real-time monitoring is crucial because it directly affects the accuracy of further neural data analysis and interpretation.

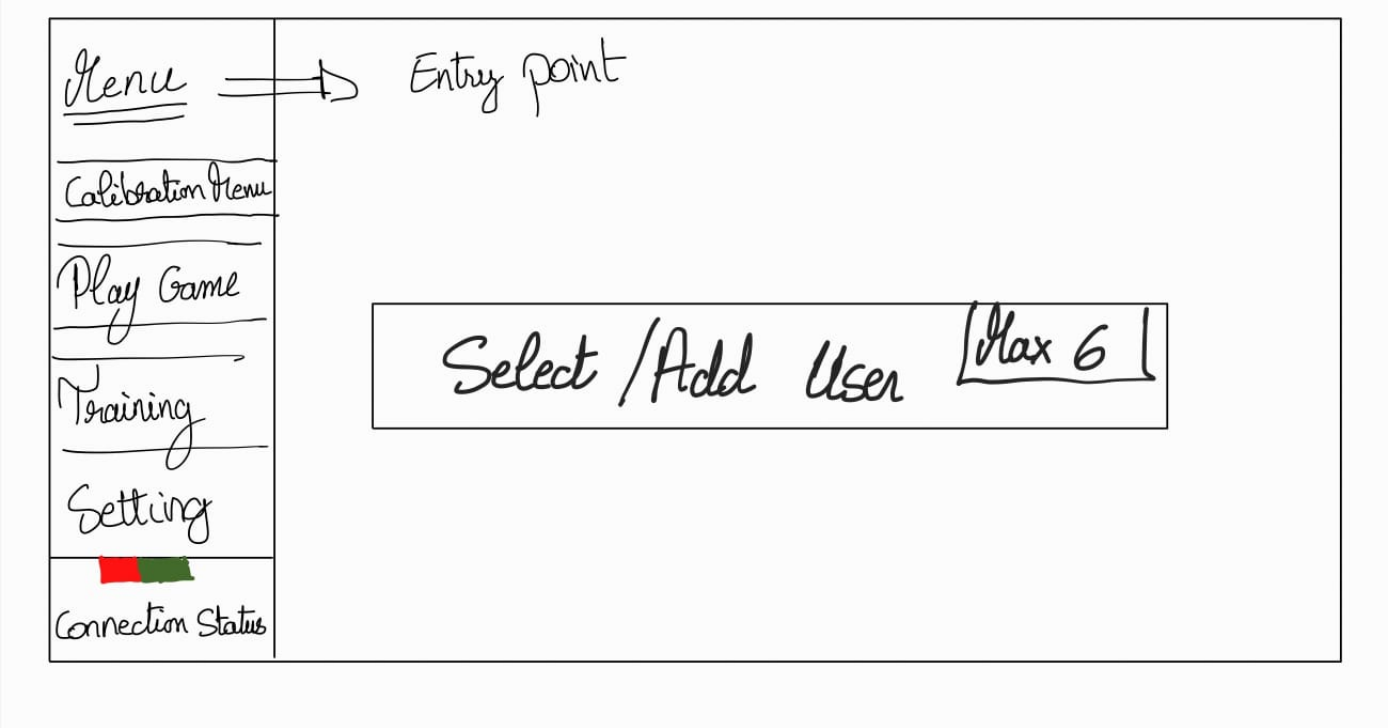


Figure 4.2: User Interface Design

The Training Section allows users to train the system to recognize instructions via three modes: left, right, and both. When left training, users concentrate leftward movements while the system records their EEG signals. Right training follows the same process as righteous action. To help the system in distinguishing between left and right actions within a single session, the training 'Both' process alternates between them. Users have the option to save or remove their training data and retrain it again. The machine learning system then receives the saved data in CSV files for additional processing to enhance accuracy.

The Game Section provides users with the chance to participate in interactive games experiences, using EEG for control. There are two separate games that users may choose from, where the first is the popular and classic game, Pong Game, the paddle moves to left or right using the EEG commands. The second one is the space shooter, a game where users stear a spaceship and shoot targets using EEG commands. Upon selecting a game, the system retrieves the relevant trained model from the ML Group, allowing the user to control the game in real time using their EEG signals.

5

Interface Development

This chapter describes the development of the user interface for the EEG-based computer system, ensuring it meets the outlined objectives and constraints in the Program of Requirements.

5.1. Background

Before going further into detail about the interface development, a little background information is needed about the method of performing actions used within the project to indicate left and right movements to the EEG. The measurement group has decided to use motor execution because these signals are generally easier to distinguish when reading an EEG compared to motor imagery signals. This is due to the fact that imagining movements is often more challenging and produces weaker and less distinct EEG signals compared to actual physical movements [12]. Thus, for left movements on the screen, squeezing in the left hand is needed and for right movements on the screen squeezing in the right hand is needed.

5.2. Choice of Framework

The development of modern user interfaces often relies on JavaScript as the industry-standard programming language due to its cross-platform compatibility and ease of deployment. JavaScript's easy use across different web browsers and computer applications make it an excellent choice for interface development. This allows developers to create user experiences in different platform without constraints like PyQt GUI. Unlike platform-dependent frameworks, JavaScript offers a platform-independent approach that aligns with the universal nature of web technologies, ensuring consistency in user experience regardless of the device or operating system [13].

In this project, Vue.js was selected as the model-view JavaScript framework for its robust architecture and extensive support for libraries and tools. Vue.js provides developers with a comprehensive model-view approach that facilitates the creation of sophisticated user interfaces with ease. Its modular and component-based structure enables developers to break down complex UIs into manageable pieces, promoting code reusability and maintainability. Moreover, Vue.js's extensive ecosystem of libraries and tools enhances the development process, offering pre-existing components and utilities that streamline UI development and optimize performance. By leveraging Vue.js and its ecosystem, the EEG-based computer interface can deliver a visually appealing, interactive, and user-friendly experience, meeting the demands of modern interface design while ensuring cross-platform compatibility and ease of deployment [14]. This choice directly addresses the requirement for an intuitive and user-friendly interface, as outlined in the general and interface group specifications.

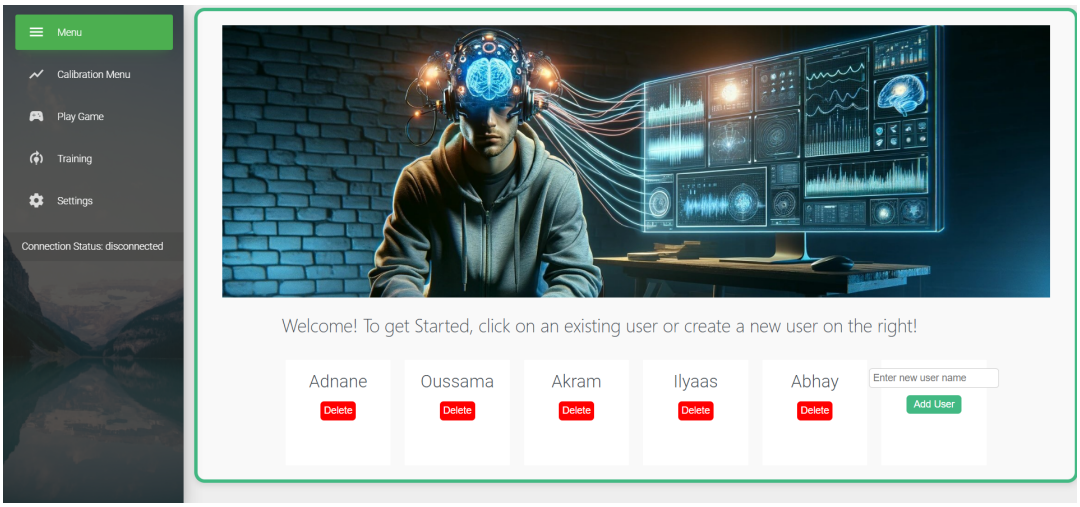
5.3. Choice of Data-streaming protocol

The choice of the Data-streaming protocol and frameworks has depended on several factors such as reliability, consistency and speed. After having tested frameworks such as Apache Spark and databases such as Redis, the latency was between 4 and 5 seconds. The programme of requirements states that the total latency

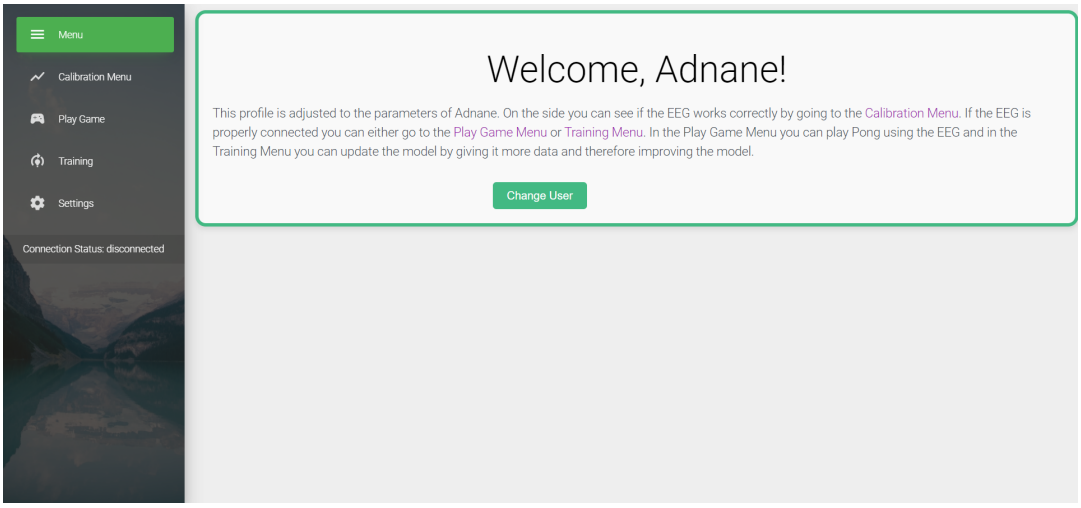
should be a maximum of 0.75 seconds. Therefore, Apache Spark and Redis have not been used and instead the websocket protocol has been used to communicate between javascript and python. This protocol has latencies of less than 0.05 seconds when running locally. Websocket is also supported as a library in both python and javascript and is therefore an excellent choice for our requirements.

5.4. Menu Components

The EEG-based computer interface features several key menu components: Menu, Calibration Menu, Play Game and Training. In the Menu the user can choose, add and delete different users, while in the Calibration Menu one can see if the EEG is properly connected by looking at the real-time graph. In the Play Game Menu the user can play 2 different games using the EEG brain data or the control keys of the keyboard.



(a)



(b)

Figure 5.1: An overview of the Main Menu

5.4.1. User Management

User management is a fundamental aspect of the EEG-based computer interface development to adapt the system to individual needs, as has been stated in the Programme of Requirements. With a limit of six users, this feature enables the addition and deletion of user profiles, ensuring optimal resource allocation and system performance. This can be seen in figure 5.1a. Each user profile has personalized parameters stored in the interface's database. Selecting a user profile configures the interface settings, including loading specific machine learning models. The menu where the specific files are loaded can be seen in figure 5.1b. Simultaneously, it enhances the baseline model to accommodate new users who have not undergone prior training. This is done by sending the user name over websocket to the python file where an if-statement checks if the new user is already within the database. If the user is not within the database, the baseline model trained by the ML group is used. This baseline model is a model that is trained on all users by the ML group and then generalized. If the user is within the database, the most recent version of the user-centric ML model is selected. The user-centric ML model is a model that is adjusted to the training data of the user and has more weights set on the training data.

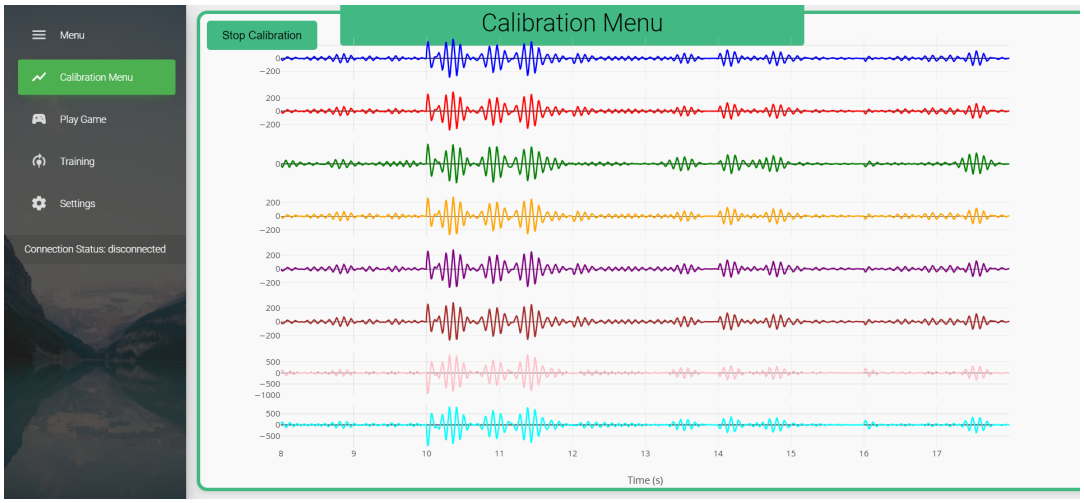


Figure 5.2: An overview of the Calibration Menu

5.4.2. Calibration Menu

The Calibration Menu within the EEG-based computer interface stands as a fundamental component for a user-friendly experience, designed to ensure precision and reliability in EEG waveform connections. This menu is crucial for determining if the EEG is properly connected. When the start button is initiated a message '1' is sent to a python file over a websocket connection. Whenever this '1' is sent the python script streams data from the serial USB that is wirelessly connected to the EEG headset. This data dat is streamed is then sent back over the websocket connection to the javascript file and is then plotted on the web app. An overview of the Calibration Menu can be seen in figure 5.2. This functionality of the Calibration Menu is essential, particularly in scenarios where the quality of EEG data directly impacts the accuracy of the machine learning model.

Moreover, to ease the visualization process and extract meaningful plots from the EEG signals, a frequency filter spanning the 8 to 30 Hz range is applied. This filter isolates the alpha and beta waves, which are known to correlate with various cognitive states and neural activities [15]. By displaying these specific frequency ranges, users can observe patterns and fluctuations in the EEG signals. In essence, the Calibration Menu serves as a diagnostic menu, giving users the chance to fine-tune their EEG equipment for optimal performance and ensuring the precision of neural data acquisition.

5.4.3. Game Menu

The Game Menu within the EEG-based computer interface offers users a dynamic experience, introducing them to two games: Space Shooter and Pong. By integrating these games into the interface, users can experience EEG technology with interactive gaming. This fulfills the requirement for an interactive and game that accurately responds to user intentions, as specified in the Programme of Requirements.

5.4.4. Training Menu

The Training Menu within the EEG-based computer interface is an important feature designed to give users a way to increase the accuracy of the ML model on that specific user and the baseline model. The menu offers three distinct training options: Left, Right, and Both.

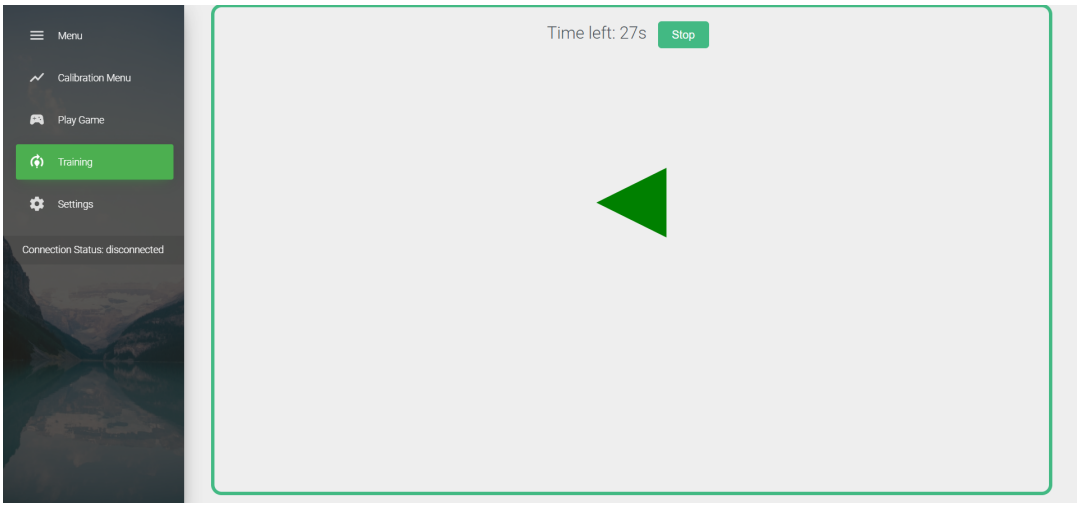


Figure 5.3: An overview of the Training Menu

Training Options

In the Left training option, users focus on training the interface to recognize EEG patterns associated with leftward cognitive directions or intentions. During this training session, users are presented with an arrow that points to the left or to engage in leftward-directed actions, such as twisting the left hand. The Training Menu with the arrow pointing to the left can be seen in figure 5.3. As users perform these actions, the interface saves this data and the ML model gets retrained based on this user data . For the 'right' training option the functionality is the same, but the arrow points to the right and the label is R. The full functionality will be described in the next chapter. The 'Both' training option offers a more dynamic approach, changing between leftward and rightward training arrows within the same session (so in the first 5 seconds Left, then Right for 5 seconds and so forth until 30 seconds have passed by).

6

Integration with Machine Learning Model

This chapter discusses the integration of the user interface with the machine learning model, ensuring the seamless operation of the EEG-based system. The interaction between the JavaScript frontend and the Python backend through WebSockets is crucial for maintaining real-time performance and user-specific model updates, as outlined in the Programme of Requirements. The integration ensures that the system can dynamically adapt to user-specific needs while providing a responsive and efficient user experience, which is critical for both gaming and potential future medical applications.

6.1. User Management and Data Handling

The user management functionality in the interface ensures optimal resource utilization. When a user is deleted, a message is sent via WebSocket to the Python backend to delete the corresponding model and measurement data. This process is facilitated by a sharedState JavaScript file, which tracks users and associates a unique number with each user. The available numbers for new users are maintained in a list, allowing for efficient management of user data. The deletion of user data helps create space for new users, ensuring the system remains flexible.

Index	Channels	Label
0	8 channels	L or R
1	8 channels	L or R

Table 6.1: Table for given data

6.2. Training Data Collection

During training sessions for left and right movements, the EEG data is collected via a serial USB connection to the computer. A Python script saves this measured data in a csv file. An example of a csv file with data can be seen in table 6.1. The first column is an index, then the subsequent 8 columns represent the channels 1 to 8 and the last column represents the labels: L or R. Also, 30 seconds of data is measured each time, meaning that because we have epochs of 0.5 seconds, the total number of epochs is 60 within one training (except if the training is stopped beforehand). The csv file is saved with the username and the corresponding number of the user. In this way the python backend can check whether the user already exists or not and based on that train an existing user-centric model or start training the baseline model with the new data. The use of user-centric models shows superior performance compared to generic models [16] [17]. So, by focusing on user-specific data, the model can significantly improve its accuracy and reliability.

6.3. Real-Time Data Streaming and Classification

For real-time gameplay, EEG data is streamed to the Python model through the serial USB, which outputs 'left' or 'right' decisions. The decisions ('left' and 'right') are then transmitted back to the JavaScript module, where they control the movements in the games 'Pong' and 'Space Shooter', as will be detailed in the game development section.

7

Game Development

The integration of BCI with game development offers a unique platform for testing and enhancing this technology. Games provide a controlled environment where real-time feedback and adaptive challenges can effectively measure the responsiveness and accuracy of BCI systems. Additionally, the interactive nature of games ensures high engagement levels, making them ideal for extended testing periods necessary for iterative development and optimization.

This project explores the integration of EEG-based control methods with traditional game control mechanisms. The primary objective is to create a series of games that can be controlled by EEG signals, thereby validating the feasibility and performance of BCI in real-world applications. The development process involves using WebSocket technology to facilitate real-time communication between the EEG device and the game, ensuring a seamless and responsive user experience.

The first game implemented was a straightforward block movement game. In this game, a block could be moved to the right or left either by using keyboard arrow keys or EEG signals. This dual-control setup was essential for ensuring the functionality of the game mechanics before integrating the EEG control method. The EEG method used WebSocket technology to facilitate real-time communication between the EEG device and the game. Initially, a Python backend script sent "right" or "left" signals every five seconds to simulate EEG inputs. This setup was crucial for validating the WebSocket communication and the game's response to EEG commands

When lunching the game interface, users are presented with a game selection screen as shown in figure 7.1, where they can choose their desired game. Each game has its own welcome message as shown in 7.2, allowing users to select their preferred control method (keyboard or EEG signals) before gameplay begins.

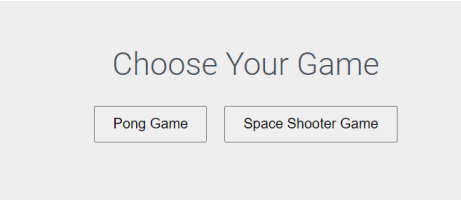


Figure 7.1: Game selection menu

7.1. Pong Game with Bricks

Building on the success of the initial game, the next development is a Pong game that includes bricks that need to be destroyed. This game also supported dual control methods, allowing the paddle to be moved either by keyboard inputs or EEG signals.

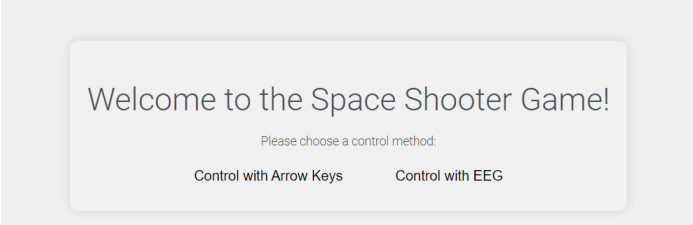


Figure 7.2: Game Welcome message with control methods Options

The game started with a control method selection screen. Players could choose between using arrow keys or EEG. If EEG was selected, the game set up a WebSocket connection to receive control signals. The paddle movement was smoothened to ensure a responsive gameplay experience. The ball's collision with the paddle and bricks was precisely implemented to enhance the game's challenge and engagement.
The bricks were randomly generated and varied in color, indicating different levels of difficulty. Green bricks required two hits to be destroyed, whereas green-yellow bricks needed only one hit. The score increased with each brick destroyed, providing a quantitative measure of the player's success.

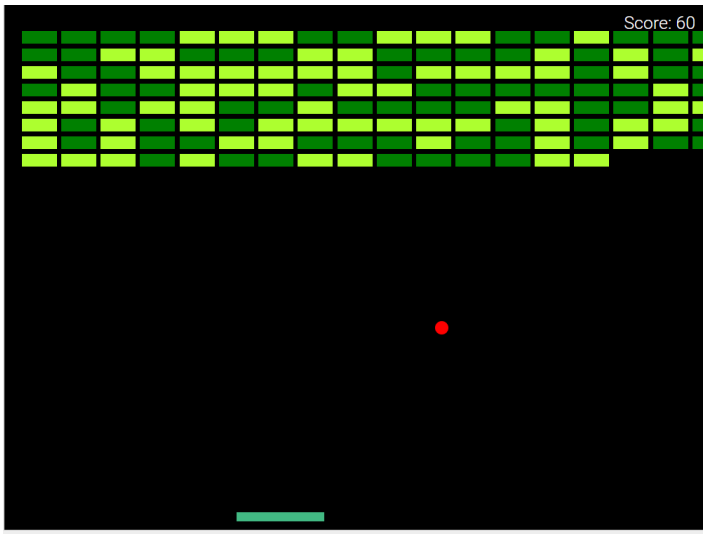


Figure 7.3: A screenshot of the PONG game

7.2. Space Shooter Game

The final and most complex game developed was a Space Shooter game. This game featured a spaceship that could move left or right and shoot bullets to destroy incoming obstacles. Upon selecting the control method, the game initiated a countdown before starting. The spaceship could be moved using EEG signals, with a WebSocket connection set up to receive these signals. The game's core mechanics included shooting bullets, generating obstacles, and detecting collisions.
The game increased the difficulty by progressively speeding up the obstacles as the player's score increased. This dynamic adjustment ensured that the game remained challenging and engaging over time. A pause functionality was also implemented, allowing players to temporarily stop the game and resume it later.

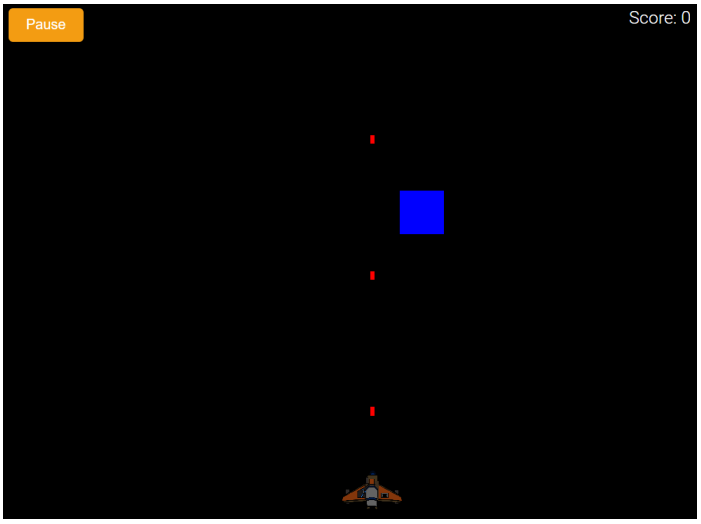


Figure 7.4: A screenshot of the Space Shooter game

Furthermore, in each game's main section, players can pause and then decide to resume, restart, or go back to the main menu. This design gives users more control and creates a smooth, enjoyable way to move between different gaming experiences. Being able to switch games easily enhances the interaction, making it more personalized and engaging based on what users prefer. Furthermore, the score was also important in each game, giving players feedback on how well they were doing. In the Pong game, players earned points by hitting the ball with the paddle and breaking bricks. The bricks came in different colors, showing their difficulty. In the Space Shooter game, players scored points by shooting and destroying obstacles. This scoring system kept players motivated to do better and showed them how successful they were.

8

Testing and Evaluation

8.1. Testing the streaming data

During the development of the calibration menu and training module we ran into a few problems. First of all, the latency was too high from streaming the data directly from the serial USB. When using the database technology Redis the latency was between 5 and 6 seconds. After switching to websocket protocol the delay between USB serial data and displaying it on the web app has gone down to 0.25 seconds. This is 20-24 times faster than using Redis. Also, by optimally using if-statements in the python backend delay has gone down. At first, all data was streamed directly to the JavaScript frontend, but this caused a queue that was too long and caused trouble for filtering the data. Therefore, filtering has been done in batches of 125 data points, corresponding to 0.5 seconds, which is also enough for one epoch for the ML classification. After having done this the delay was at maximum 0.15 seconds. This shows that carefully testing and evaluating the performance can significantly improve the performance of the overall system.

8.2. Game Testing and evaluation

We initially checked the games functionality using just keyboard controllers throughout the testing and assessment stage. Before introducing controls based on EEG, this helped us lay a solid and dependable basis. Every game was thoroughly tested by moving components like blocks or paddles using keyboard keys. To facilitate real-time interaction between the game and the EEG equipment, we next included a WebSocket connection. To mimic EEG inputs, we wrote a Python script that broadcast left or right signals every five seconds. This stage was essential to determining how effectively the game reacted to the directions.
As we developed, we ran into a number of challenges and mistakes. But we were able to resolve these problems and produce games that functioned properly by meticulous debugging and plenty of testing. Both control strategies underwent fine-tuning to guarantee smooth integration and prompt reaction.

To allow using EEG headsets to control the games, we also connected our interface with the machine learning department. Enhancing the whole gaming experience and enabling efficient assessment of EEG-based game control, this integration offered the required capabilities to interpret EEG data and convert them into game commands.

9

Conclusion and discussion

The conclusion of this thesis is that it illustrates that an EEG-based computer interface is both feasible and practical. The combination of complex machine learning models with real-time EEG data processing has resulted in a considerable improvement in the interaction that takes place between humans and computers. The results of our research shown that the system is capable of correctly understanding brain signals and translating them into computer instructions, so delivering a user experience that is more intuitive and simple. The research found a number of limitations, including the need for additional testing across a larger range of user groups and the need for further developing of signal processing algorithms in order to handle a wider array of brain patterns. Despite these encouraging findings, the study found numerous limitations. Future study should try to solve these limitations and also investigate the potential of this technology in several fields, such as healthcare, where it might be used for monitoring and rehabilitation, and education, where it could provide new learning tools. Future research should also examine the potential of this technology in healthcare. In addition, applications in the field of entertainment, especially those in the gaming industry, have the potential to change user interaction and involvement. The overall result of this study is that it creates a solid basis for the future growth of brain-computer interface technology. It also opens the way for creative applications and advancements in human-computer interaction.

References

[1] X. Gu, Z. Cao, A. Jolfaei, et al., "EEG-Based Brain-Computer Interfaces (BCIS): A survey of recent studies on signal sensing technologies and computational intelligence approaches and their applications," IEEE/ACM transactions on computational biology and bioinformatics, vol. 18, no. 5, pp. 1645–1666, Sep. 2021. DOI: 10.1109/tcbb.2021.3052811. [Online]. Available: https://doi.org/10.1109/tcbb.2021.3052811.

[2] A. Craik, J. J. González-España, A. Alamir, et al., "Design and Validation of a Low-Cost Mobile EEG-Based Brain–Computer Interface," Sensors, vol. 23, no. 13, p. 5930, DOI: 10.3390/s23135930. [Online]. Available: https://doi.org/10.3390/s23135930.

[3] S. Liu, M. Claypool, A. Kuwahara, J. Sherman, and J. J. Scovell, "Lower is Better? The Effects of Local Latencies on Competitive First-Person Shooter Game Players," Association for Computing Marchinery, May 2021. DOI: 10.1145/3411764.3445245. [Online]. Available: https://doi.org/10.1145/3411764.3445245.

[4] A. Nogales and J. García-Tejedor, "A systematic review of electroencephalography open datasets and their usage with deep learning models," IEEE access, vol. 11, pp. 72 384–72 399, Jan. 2023. DOI: 10.1109/access.2023.3293421. [Online]. Available: https://doi.org/10.1109/access.2023.3293421.

[5] L. Lim, U. Gim, K.-J. Cho, D. Yoo, H. G. Ryu, and H. C. Lee, "Real-time machine learning model to predict short-term mortality in critically ill patients: development and international validation," Critical care, vol. 28, no. 1, Mar. 2024. DOI: 10.1186/s13054-024-04866-7. [Online]. Available: https://doi.org/10.1186/s13054-024-04866-7.

[6] L. Tuţă, G. Roşu, C. Popovici, and I. Nicolaescu, "Real- time eeg data processing using independent component analysis (ica)," in 2022 14th International Conference on Communications (COMM), 2022, pp. 1–4. DOI: 10.1109/COMM54429.2022.9817209.

[7] A. Samosir and A. Author2, "Machine learning models for decoding motor intentions from eeg signals," Journal Name, vol. 122, no. 3, pp. 456–467, 2023.

[8] A. Nogales and A. Author2, "Eeg frequency bands differentiation for motor activities," Journal Name, vol. 121, no. 2, pp. 345–356, 2022.

[9] A. Katsifodimos and S. Schelter, "Apache flink: Stream analytics at scale," in 2016 IEEE International Conference on Cloud Engineering Workshop (IC2EW), 2016, pp. 193–193. DOI: 10.1109/IC2EW.2016.56.

[10] M. A. Mohedano-Munoz, C. Soguero-Ruíz, I. Jiménez, M. Rubio-Sánchez, J. Álvarez Rodríguez, and A. Sánchez, "A streaming data visualization framework for supporting decision-making in the Intensive Care Unit," Expert systems with applications, vol. 227, p. 120 252, Oct. 2023. DOI: 10.1016/j.eswa.2023.120252. [Online]. Available: https://doi.org/10.1016/j.eswa.2023.120252.

[11] L. Qigang and X. Sun, "Research of web real-time communication based on web socket," International Journal of Communications, Network and System Sciences, vol. 05, pp. 797–801, Jan. 2012. DOI: 10.4236/ijcns.2012.512083.

[12] C. Neuper, R. Scherer, M. Reiner, and G. Pfurtscheller, "Imagery of motor actions: Differential effects of kinesthetic and visual–motor mode of imagery in single-trial eeg," Cognitive Brain Research, vol. 25, no. 3, pp. 668–677, 2005.

[13] JavaScript vs PyQt | What are the differences? [Online]. Available: https://stackshare.io/stackups/javascript-vs-pyqt.

[14] E. Bojanowska and E. Bojanowska, Pros and cons of the Vue.js framework. [Online]. Available: https://naturaily.com/blog/pros-cons-vue-js.

[15] Z. Fodor, A. Horváth, Z. Hidasi, A. A. Gouw, C. J. Stam, and G. Csukly, "EEG Alpha and Beta Band functional connectivity and network structure mark Hub overload in mild cognitive impairment during memory maintenance," Frontiers in aging neuroscience, vol. 13, Oct. 2021. DOI: 10.3389/fnagi.2021.680200. [Online]. Available: https://doi.org/10.3389/fnagi.2021.680200.

[16] J. Smith, "User-centric models for eeg-based brain-computer interfaces," Journal of Neural Engineering, vol. 17, no. 2, p. 045 001, 2020.

[17] A. Brown, "Improving bci performance with personalized machine learning," IEEE Transactions on Biomedical Engineering, vol. 66, no. 9, pp. 2530–2541, 2019.

Appendix

Source Code (click here to go to the sourcecode page)

Statement of use of AI

Various AI tools have been utilized throughout the creation of this report for several purposes. These tools were employed to correct grammar, summarize and clarify certain papers that were ambiguous or written in complex language. Additionally, they were used to enhance the readability of some sections that we had written ourselves.