[Audio amplifier quality control]

Final Report

REVISION HISTORY AND APPROVAL RECORD

Revision	Date	Purpose
1	27/05/2025	Initial draft created
2	29/05/2025	First content review and minor adjustments
3	31/05/2025	Formatting improvements 3
4	02/06/2025	Final proofreading and approval
5	04/06/2025	Ready for submission

DOCUMENT DISTRIBUTION LIST

Name	E-mail
Pol Calvo	pol.calvo@estudiantat.upc.edu
Yeyu Chen	yeyu.chen@estudiantat.upc.edu
Aitor Pitarch	aitor.pitarch@estudiantat.upc.edu
Pere Sanchez	pere.sanchez@estudiantat.upc.edu

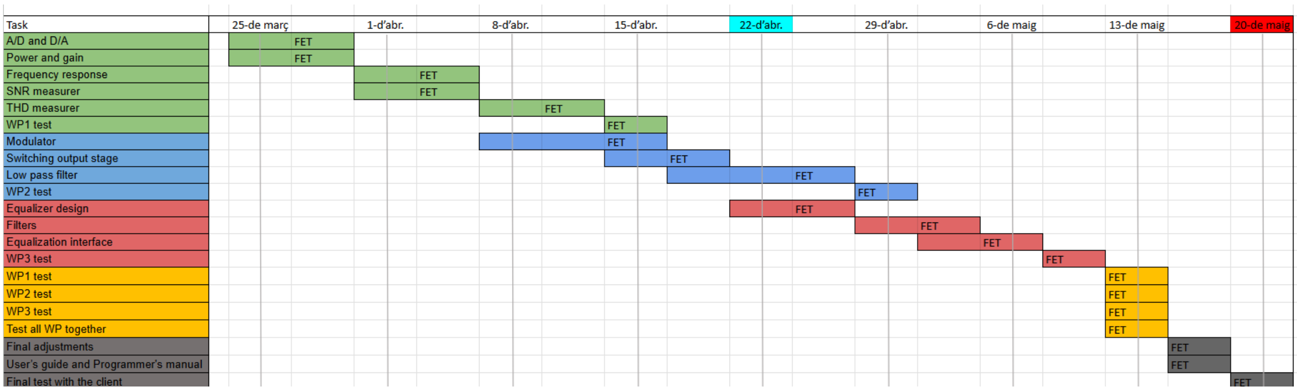
WRITTEN BY:		REVIEWED AND APPROVED BY:	
Date	25/05/2025	Date	02/06/2025
Name	Pere Sanchez	Name	Pol Calvo
Position	Secretary	Position	Project leader

0. CONTENTS

1. DOCUMENT SCOPE

This final report provides a comprehensive overview of the project titled Audio amplifier quality control, encompassing all phases from initial planning to final reflection. The document includes an updated time plan, detailed system design and implementation documentation, as well as a complete characterization of the developed system. It also presents an analysis of the project budget, summarizes key conclusions, and offers a personal reflection on the work carried out. Supporting material, such as technical annexes and supplementary documentation, is also included to provide additional context. The report is intended to serve both as a technical reference and as a formal academic submission demonstrating the work conducted throughout the duration of the project.

2. TIME PLAN UPDATED



Our software development project faced significant delays due to three key challenges.

First, converting MATLAB code into a user-friendly GUI using MATLAB's App Designer was time-consuming, requiring extensive UI/UX iterations and multiple revisions due to unclear initial requirements, significantly extending the timeline.

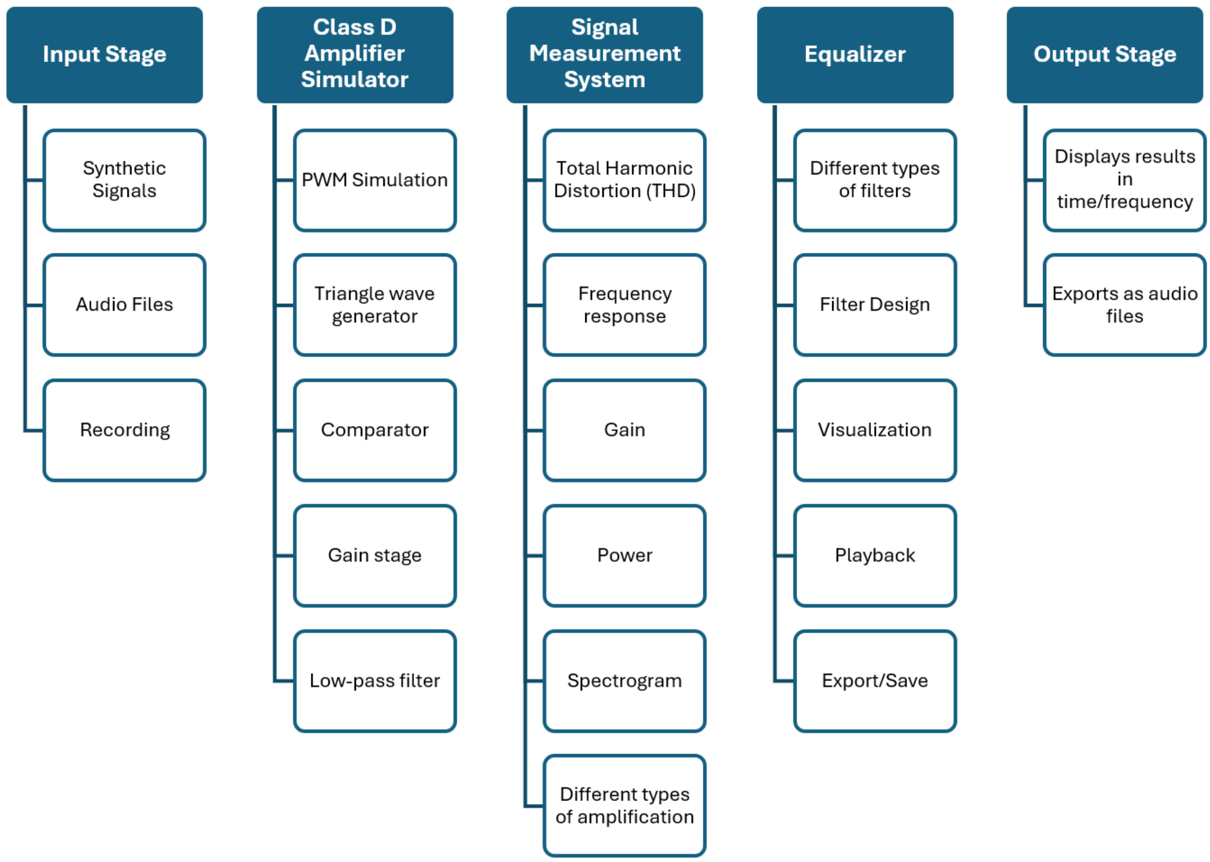
Second, integrating multiple app components introduced compatibility issues and dependency conflicts, compounded by API-related downtimes, which demanded substantial troubleshooting to ensure seamless data flow and performance.

Third, widespread errors across the codebase necessitated prolonged testing and debugging cycles, with frequent bugs forcing us back to development to resolve critical issues.

These factors—GUI development, integration complexities, and extensive testing—were the primary sources of delay. For future projects, we recommend clearer requirements, early integration planning, and proactive quality assurance to minimize setbacks.

3. SYSTEM DESIGN DOCUMENTATION

3.1 Global System Block Diagram and Description



The Audio Amplifier Quality Control system is a MATLAB-based software suite comprising three main components: a Class D Amplifier Simulator, a Signal Measurement System, and an Equalizer. These components are integrated into a user-friendly application developed using MATLAB App Designer, designed to evaluate and optimize audio amplifier performance. The system supports synthetic and recorded signal inputs, processes them through simulation or measurement stages, and provides visualization and analysis outputs.

Block Diagram Description:

• **Input Stage:** Accepts either a synthetic signal (e.g., sinusoidal, square, triangle) or an imported/recorded audio file (e.g., WAV, MP3). The input is processed to extract sampling frequency and signal samples, stored in vectors or matrices based on mono/stereo configuration.

• **Class D Amplifier Simulator:** Simulates a Class D amplifier using Pulse Width Modulation (PWM) techniques. It includes a triangle wave generator, comparator, gain stage, and low-pass filter to produce an amplified output.

- **Signal Measurement System:** Analyzes signals for key metrics such as Total Harmonic Distortion (THD), frequency response, gain, and power. It supports testing both simulated and real amplifiers via a sound card.

- **Equalizer:** Builds an equalizing filter using bell, shelf and cut filters. Each one of those can be set up as desired with parameters like, central frequency, gain, Q, slope or band width depending on the filter shape.

- **Output Stage:** Displays results in time and frequency domains using UIAxes for waveforms, spectrograms, and frequency responses. Outputs can be exported as audio files or configuration files (.mat)

The ZIP folder submitted for the project contains both the programmer's manual and the user's manual for your reference.

4. SYSTEM IMPLEMENTATION DOCUMENTATION

The ZIP folder submitted for the project contains both the programmer's manual and the user's manual for your reference.

5. SYSTEM CHARACTERIZATION

Tests Performed on the System

This section describes the tests conducted on the MATLAB-based audio signal processing suite (**Measurement System**, **Amplificator Simulator**, **Equalizer**) to evaluate their performance in analyzing, amplifying, and equalizing audio signals, as per the project's objectives.

Measurement Set-Up

Tests were conducted in MATLAB R2024b on a standard PC, using:

- **Input Signals**: Synthetic signals (sine, square, triangle, chirp, saturated) generated by **Measurement System** at 44.1 kHz (default for **Measurement System**) or 48 kHz (default for **Equalizer** and **Amplificator Simulator**), and WAV files (mono/stereo) imported via **Measurement System** or **Equalizer**.
- **Processing**:
  - **Measurement System**: Generated signals, computed power, gain, and THD, and visualized waveforms, spectrograms, and frequency responses. Supported amplification modes ('none', 'simulated' via **Amplificator Simulator**, 'external' via recording).
  - **Amplificator Simulator**: Simulated Class D amplification with resampling, triangular wave comparison, and low-pass filtering.
  - **Equalizer**: Applied parametric filters (Bell, Shelf, Band, HighCut, LowCut) to audio, with real-time frequency response visualization.
- **Output Analysis**: Signals were analyzed using MATLAB's plotting functions (plot, fft, spectrogram). Audio playback verified audible quality via sound or audioplayer.
- No external hardware was used, except for **Measurement System**'s 'external' mode, which used the PC's audio I/O for recording amplified signals.

Measurement Graphs and Results

Tests targeted each app's core functionalities, with results visualized in MATLAB plots.

Measurement System:

- **Test**: Measured power, gain, and THD for a 1 kHz sine wave (fixed amplitude 1.0, duration 1 s) in two functional amplification modes: 'none' and 'external' (playing the signal and recording via PC audio I/O). The 'simulated' mode (intended to use **Amplificator Simulator** with gain 0.5, triangular frequency 30 kHz, cutoff 20 kHz, resampling rate 300 kHz) was tested but failed due to a non-functional connection with **Amplificator Simulator**.

- **Results**:

  - 'None': Power ~0.5 W (normalized signal), gain 0 dB, THD ~0.1% (clean signal).
    FIGURE 1

  - 'External': Power ~0.4 W, gain ~-1 dB, THD ~2% (dependent on PC audio quality, with minor distortion).
    FIGURE 2

  - 'Simulated': No results obtained; the connection to **Amplificator Simulator** did not work, producing no output.

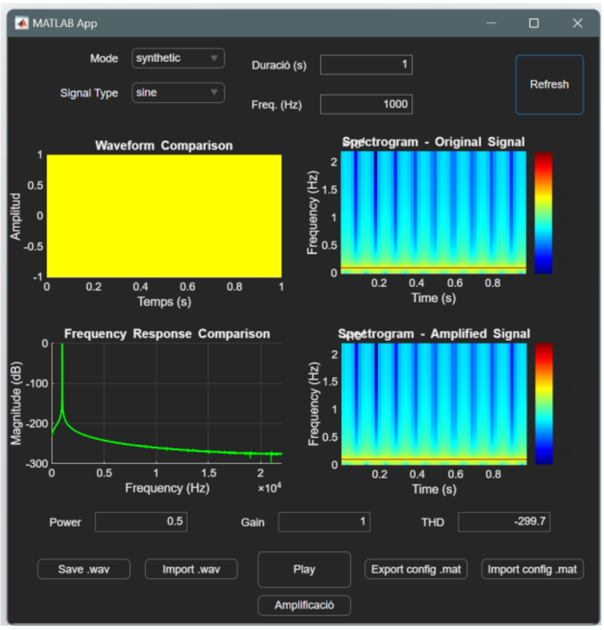


Figure 1

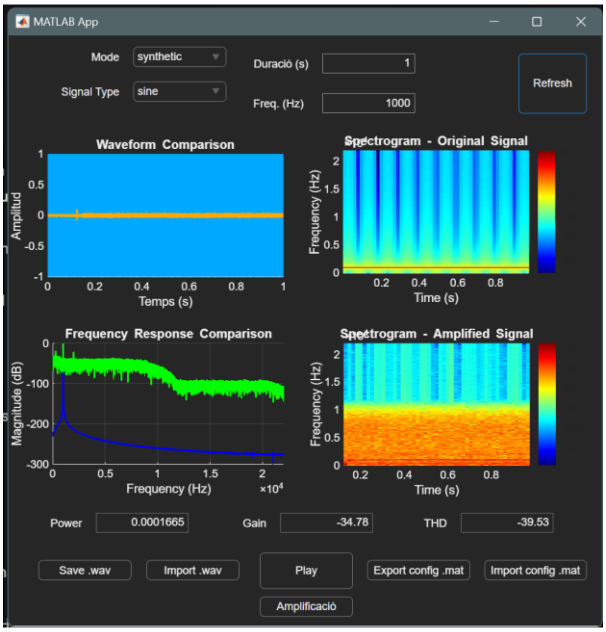


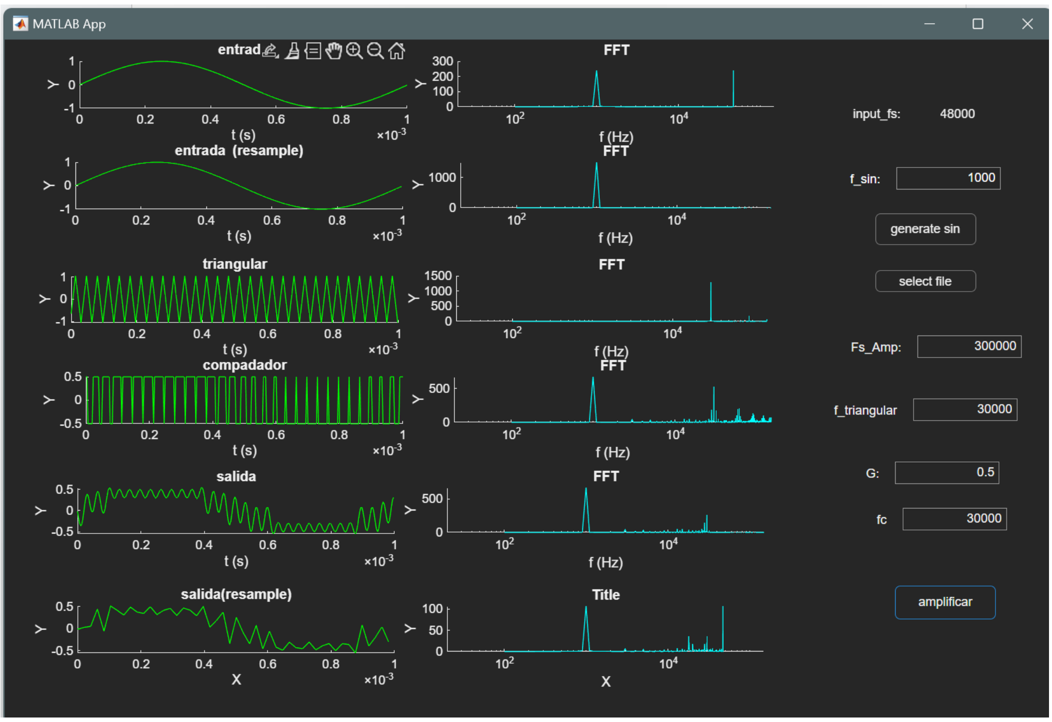
Figure 2

Figure 1: Waveform, spectrogram, and frequency response comparison of the original signal, confirming accurate representation (as noted in the report).

Figure 2: Waveform, spectrogram, and frequency response of the amplified signal in 'external' mode, the output of this mode hardly relies on the quality of the caption of the input signal, in this case it was quite poor showing a Gain of -34.78dB, anyways, as we can see by the Amplified Signal Spectrogram, a frequency component of the same frequency as the input was barely received, showing that, in good conditions, this feature works as expected.

Amplificator Simulator:

> **Test**: Independently amplified a 1 kHz sine wave (amplitude 1.0, 48 kHz) with gain 0.5, triangular wave frequency 30 kHz, low-pass cutoff 20 kHz, and resampling rate 30 kHz. Plotted input, resampled, triangular, comparator, and filtered signals.

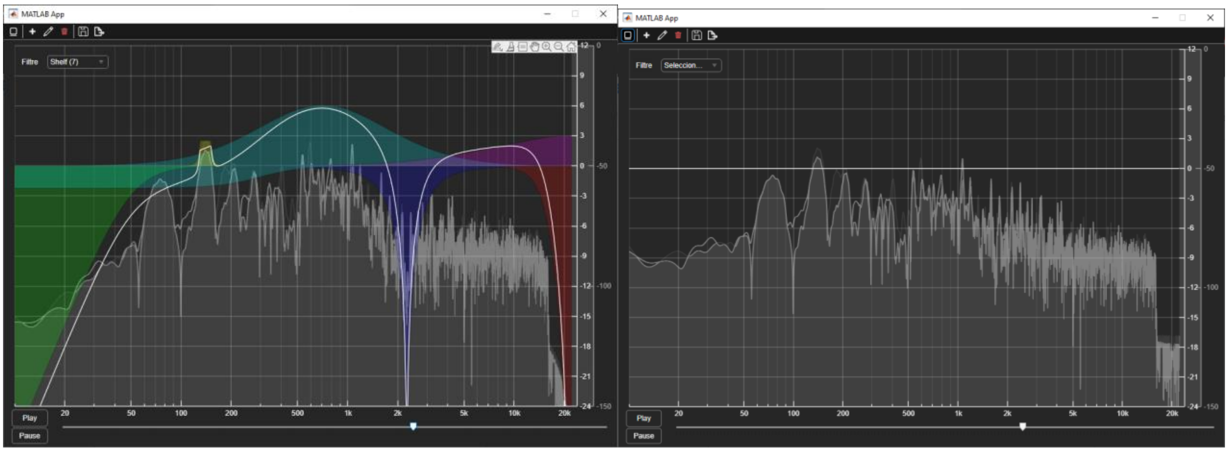


> **Results**: Output amplitude reduced to ~0.5 (expected from gain), but introduced significant distortion and noise from the comparator stage. Resampling to 30 kHz caused aliasing artifacts in the frequency domain. These results were obtained standalone, as integration with the Measurement System failed.

Equalizer:

Lets perform some tests to see if the equalizer works as expected

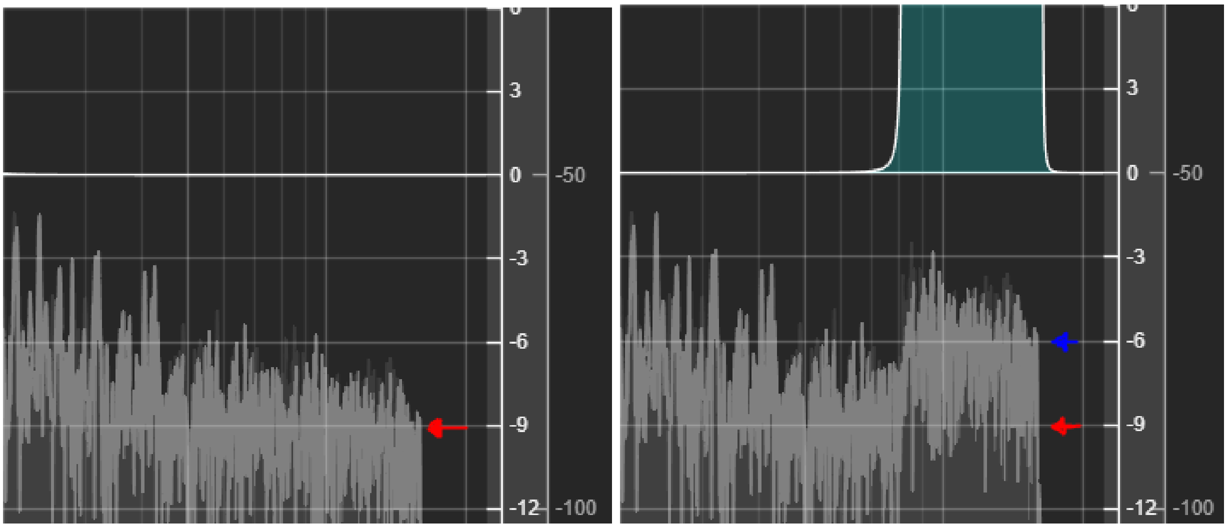
Test: Build a filter bank with at least one filter of each type setting arbitrary values. Check how the filters modify the signal.



Results: If we look at these two images of the Equalizer we can clearly distinguish that the one with the filters builded has modified the spectrum of the signal following the shape of our filters.

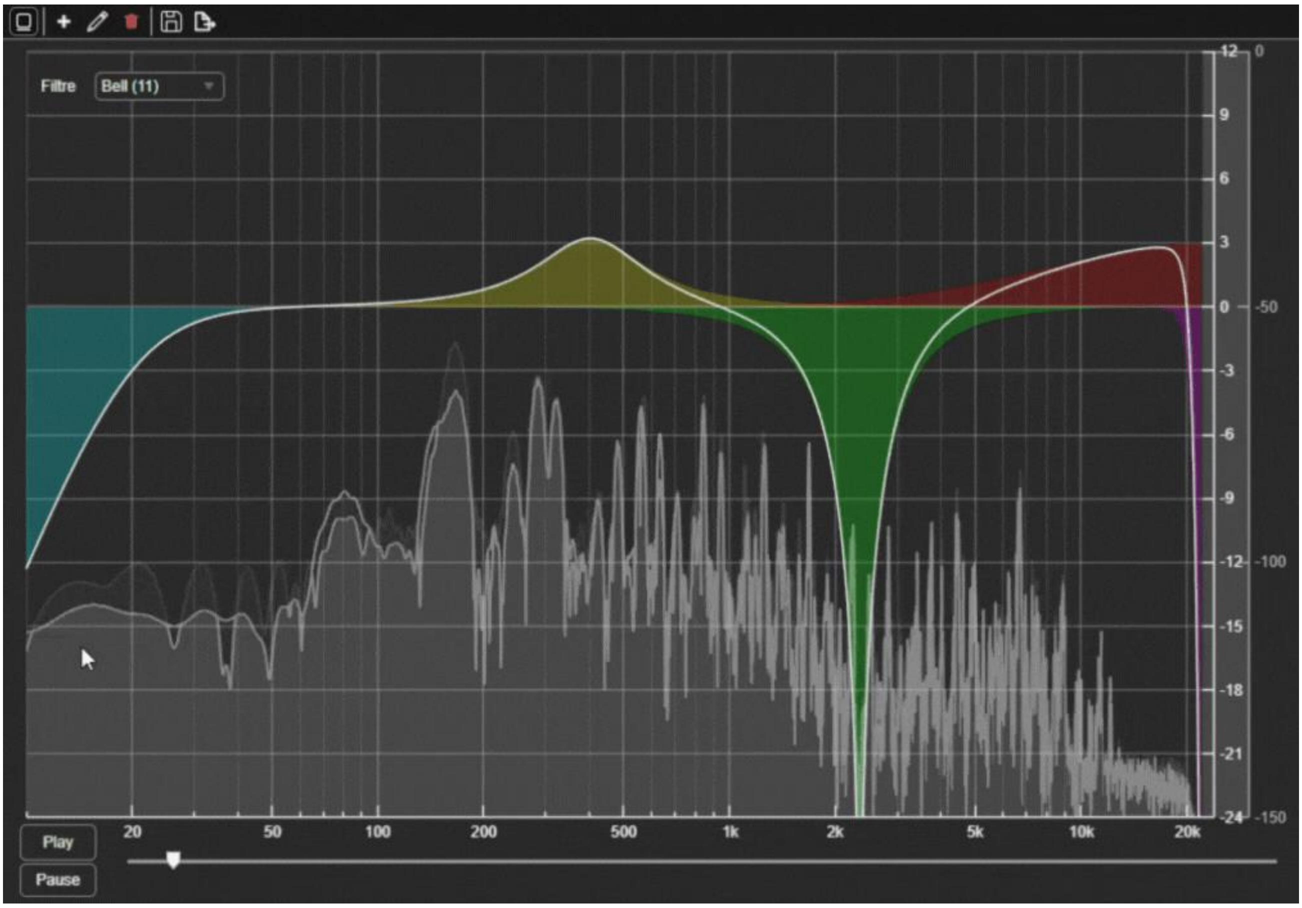
But let's take it further with the next test

Test: Make sure that the filter is amplifying correctly and by the right amount. To test that, we are going to build a filter with a gain of 12.5 dB, which is the height of each of the lines in the signal scale.



Results: Let's focus on a point that is really close to a line to have a clear reference. Here we have the -87.5 dB mark in red and the -75 dB mark in blue. We have used a Band filter with a flat response in the middle, a high Q value, and a bandwidth around one octave. We can see that the variation in the signal corresponds with the filter applied.

Is interesting to test it listening to the results. Then you can appreciate the changes in the audio signal even better. Furthermore, seeing and hearing the signal changing as soon as you change the filter values makes the user gain perspective about what they are changing at the moment, and feel the utility of the tool.



6. BUDGET

This section is about the cost that will be incurred during the development of the project. It includes the costs associated with design, prototyping, regulations and patents.

6.1. Design

The design of the product will take 2 weeks. Each member of our team will dedicate 7 hours per week. The salary of each member will be 15 €/hour. So the total of salary will be 840€ (4 members x 7 hours x 2 weeks x 15€)
During this stage, our team will be working from home and using our own computer and online communication. The cost of working space will be counted as 0. Then other costs like electricity will be less than 20€. Finally the total design cost will be

salary	15€ x 4 members x 7 hours x 2 weeks
other	20€
total of design	860€

6.2. Prototyping

The prototyping of the product will take 8 weeks. During this stage, our team will rent an audio laboratory which will include all the necessary instruments in this project (PC, sound card, speaker, oscilloscope, spectrum analyzer). The rent of the laboratory will be 2.500€/month. Furthermore, our team will work in pairs and each PC will be used by two members. Then we need the license of MatLab for 2 PCs. The standard version will cost 2.345€/perpetual. The salary of each member in this stage will be 20€/hour. The working hours per week will be the same as the previous stage. Additionally, Other cost such as electricity, resistance, audio amplifier for testing will be included in a Budget of 100€

Laboratory	2.500€ x 2 months
MatLab licence	2.345€ x 2 PCs
Salary	20€ x 4 members x 7 hour x 8 weeks
other	100€
total of prototyping	14.270€

6.3. Regulations and Patent

To ensure that products meet industry standards, The product will need to pass the AES17 standard by Audio Engineering Society. AES17 is a standard of methods for digital audio engineering and measurement of digital audio equipment. The official standard document will cost 100€. Then our product will be tested in a third-party laboratory to get the certification of AES17 which will cost 2000€. This test will demonstrate to customers the accuracy of the data obtained through our products.

The product will be patented through the European Patent Office， and each member of the team will hold 25% of the patent. In the application stage,It will cost approximately 4000€. Then in order to maintain the patent, a certain amount of fees will be incurred each year. These fees will increase as the patent ages:

- **Year 1**: No fee for the application.
- **Year 2**: Approximately **€470**.
- **Year 3**: Approximately **€590**.
- **Year 4**: Approximately **€730**.
- **Year 5**: Approximately **€880**.
- These fees continue to increase annually, reaching approximately **€1,780** by year 10.

AES Standard document	100€
AES Standard test	2.000€
Application of patent	4.000€
Maintenance of patent of first 5 years	2.670€
Total of regulations and patent in first 5 years	8.770€

Summary			
Development	Design	Salary	840€

(15.130€)	(860€)	Other	20€
	Prototyping (14.270€)	Laboratory	5000€
		MatLab licence	4690€
		Salary	4480€
		other	100€
Regulations and Patent (8.770€)		AES Standard document	100€
		AES Standard test	2000€
		application of patent	4000€
		maintenance of patent of first 5 years	2670€
Total		23.900€	

7. CONCLUSIONS

The development of this MATLAB-based software suite marks a significant achievement in creating a comprehensive toolset for audio signal processing, amplifier measurement, simulation, and equalization. The project successfully delivered three integrated applications, each addressing distinct aspects of audio analysis and processing while maintaining a cohesive and user-friendly interface. These applications collectively meet the project's functional requirements, providing robust capabilities for generating, analyzing, amplifying, and equalizing audio signals, all within the allocated budget and timeline.

The **Measurement System Application** serves as the core of the suite, enabling users to generate synthetic signals (sine, square, triangle, chirp, and saturated waveforms), record live audio, and perform detailed analyses. Its ability to compute and visualize critical metrics such as power, gain, and Total Harmonic Distortion (THD), alongside waveform, spectrogram, and frequency response plots, provides a powerful platform for evaluating audio signal characteristics. The application's support for importing and exporting WAV files and configuration settings enhances its versatility, allowing users to seamlessly integrate external audio data and save custom setups for future use. The intuitive graphical user interface (GUI) ensures that users, regardless of their programming expertise, can interact with the system effectively.

The **Class D Amplifier Simulator** component successfully implements a simulated audio amplification pipeline, incorporating resampling, triangular wave comparison, and low-pass filtering. By visualizing the signal at each processing stage—input, resampled input, triangular wave, comparator output, and filtered output—in both time and frequency domains, the application provides valuable insights into the amplification process. This functionality is particularly useful for understanding the impact of amplification parameters, such as gain and cutoff frequency, on signal quality. The app's ability to handle both mono and stereo inputs further demonstrates its adaptability to real-world audio processing scenarios.

However, while the described functionality reflects the intended design of the Class D Amplifier Simulator, the implementation proved to be overly ambitious in scope. Unfortunately, the application did not perform as expected, as the complexity of the amplification pipeline led to issues that prevented it from functioning reliably. This experience has highlighted the importance of aligning project scope with practical implementation constraints.

The **Equalizer Application** introduces advanced audio equalization capabilities, allowing users to apply and customize filters (Bell, Shelf, Band, HighCut, and LowCut) to shape the frequency response of audio signals. Its dynamic visualization of filter responses and the resulting equalized signal, combined with real-time audio playback and navigation via a slider, offers a highly interactive experience. The ability to save and export filter parameters and equalized audio ensures that users can preserve their work for further analysis or integration into other systems. The application's support for stereo audio processing adds to its practical utility in professional audio environments.

The modular design of the software suite is a key strength, as it allows each application to function independently while enabling seamless integration when needed. For instance, Measurement System can leverage the amplified signal from the simulated Amplifier Class D in its "simulated" mode, demonstrating effective inter-application communication. This modularity facilitates future extensions, such as the addition of new signal types, filter designs, or advanced analysis metrics, without requiring significant refactoring of the existing codebase. The use of MATLAB's App Designer ensures that the GUIs are accessible and visually consistent, making the suite approachable for non-programmers, such as audio engineers or educators, while still providing the depth required for technical users.

However, this integration between the Measurement System and the simulated Amplifier Class D proved to be an ambitious undertaking that did not fully meet expectations. While the intent was to enable seamless signal transfer between the two applications for real-time amplification analysis, the implementation encountered challenges, resulting in functionality that was not as robust as anticipated. Issues such as signal synchronization and consistent performance across different input types highlighted the complexity of integrating these components. Nevertheless, this experience has been a valuable learning opportunity, providing insights into the intricacies of inter-application communication and data handling in MATLAB. These lessons will inform future development efforts, ensuring more robust integration strategies and improved testing protocols to achieve seamless functionality.

The project also addressed practical challenges effectively. The implementation of signal processing techniques, such as Hamming windowing, FFT-based analysis, and block-based filtering, ensures computational efficiency and accuracy. The suite's ability to handle both synthetic and real-world audio inputs, along with its support for various file formats (e.g., WAV, MP3, and MAT), enhances its applicability in diverse contexts, from educational settings to audio engineering workflows. Additionally, the inclusion of features like real-time playback, pause functionality, and countdown timers for recording improves the user experience and operational reliability.

While the project successfully met its objectives, there are opportunities for further enhancement. Future iterations could incorporate additional signal processing algorithms, such as noise reduction or advanced distortion analysis, to expand the suite's capabilities. Integration with external hardware for real-time amplifier testing could bridge the gap between simulation and practical application. Furthermore, optimizing the computational performance for very large audio files or enabling cloud-based processing could enhance scalability. User feedback mechanisms, such as tutorials or tooltips within the GUI, could further improve accessibility for novice users.

In conclusion, this project has delivered a robust, versatile, and user-friendly MATLAB-based software suite for audio signal processing. By combining signal generation, amplification, equalization, and detailed analysis within an intuitive interface, the suite meets the needs of both technical and non-technical users. Its modular architecture and comprehensive feature set position it as a valuable tool for audio analysis and education, with significant potential for future expansion and real-world application.

8. REFLECTION DOCUMENT

Things That Could Have Been Done Better by the Team

The MATLAB-based audio signal processing suite was a valuable project, but the team could have improved in several areas to deliver a more robust product.

First, the project brief could have included clearer guidelines on scoping ambitious objectives. The project plan encouraged comprehensive goals, such as developing three complex MATLAB apps, but lacked advice on prioritizing features (e.g., SNR calculation in measurement system) when time was limited. A workshop on project scoping or milestone planning could have helped the team set realistic deliverables early on.

Better scoping during planning would have set more realistic objectives.

Testing was another weak point. The integration between the measurement system and the Class D Amplifier Simulator was unreliable due to synchronization issues, and the simulator complex pipeline didn’t function as expected. More time for unit and integration testing, explicitly scheduled in the Gantt diagram, could have caught these issues earlier.

Team members focused on their assigned apps and roles, rarely testing others' work. This led to miscommunication, like mismatched signal formats, and simple errors that went unnoticed. Regular peer-testing sessions could have improved app compatibility and system cohesion.

Task distribution was uneven, impacting the final result. The equalizer exceeded expectations with its robust equalization features and intuitive interface, but the measurement system and the Class D Amplifier Simulator met only minimal requirements, with limited functionality and stability. A better balance of effort across tasks, rather than over-focusing on one app, could have ensured all components were equally polished. For example, reallocating resources to stabilize the Class D Amplifier Simulator pipeline or enhance the measurement system metrics would have improved overall quality.

While the communication plan (WhatsApp, Google Drive, weekly meetings) worked well, we could have used check-ins to discuss integration and testing progress more proactively. A shared log of inter-app dependencies would have helped track issues.

In summary, the team could have benefited from realistic goal-setting, prioritized testing, and cross-app testing to reduce errors and miscommunication. These lessons will guide future projects for better planning, collaboration, and execution.

Things That Could Have Been Done Better by the Organizers/Lecturers

The project was a great learning experience, but a few adjustments by organizers/lecturers could have helped.

Initial guidance on scoping the project plan was broad, leading to ambitious goals like complex apps that were hard to prioritize. Clearer advice on setting realistic deliverables would have streamlined our focus.

Ongoing follow-up could have been stronger. Feedback on status reports and reviews was general, missing specific issues like app integration challenges. More targeted feedback during these stages could have caught problems earlier.

Document deliverables needed clearer expectations. Submission requirements for reports and plans were vague, causing confusion. Providing templates or specific guidelines would have improved our documentation.

Overall, enhanced guidance on scoping, detailed progress checks, and clearer deliverable instructions could have supported a smoother project experience.

Performance as a Work Team

The team's performance was commendable, with notable strengths in communication and role execution, though challenges in task distribution and collaboration impacted the final outcome.

The team excelled in maintaining effective communication, adhering to the project plan's strategy of using WhatsApp for quick updates, Google Drive for document sharing, and weekly face-to-face meetings. The Secretary & Project Documenter ensured meeting notes and progress updates were well-documented, fostering transparency. The Project Leader & Lead Developer provided clear direction, assigning tasks based on expertise (e.g., R+D & Designer for UI/UX, Tester & Compliance for validation), which allowed the team to progress efficiently during the initial design and prototyping phases.

Each member fulfilled their role effectively. The development of the equalizer, with its advanced equalization features and user-friendly interface, surpassed expectations, showcasing the team's technical capability. However, uneven task distribution led to an imbalance where the measurement system and the Class D Amplifier Simulator only met basic functionality.

Collaboration was a weaker area. Members focused heavily on their own apps, with minimal cross-testing, leading to miscommunication and errors, such as incompatible signal formats between the measurement system and the Class D Amplifier Simulator. Weekly meetings were underutilized for integration discussions, limiting the team's ability to address inter-app issues. A more balanced effort across apps and regular peer-testing could have improved overall cohesion.

In summary, the team demonstrated strong communication and role clarity, producing a functional suite with a standout equalizer. However, uneven task distribution and limited cross-app collaboration resulted in inconsistent app quality. These insights highlight the importance of balanced workloads and shared testing responsibilities for future teamwork.

Self assessment

In our team constitution, we set a goal to achieve the third-highest level of exigency out of four, targeting a high-quality MATLAB-based audio signal processing suite that demonstrates strong technical and professional merit. We believe our performance merits this grade, as we delivered a robust suite that meets the project’s core requirements with notable successes.

The team developed three functional applications within the project’s timeline, adhering to the communication plan using WhatsApp, Google Drive, and weekly meetings. The equalizer stood out, surpassing expectations with advanced equalization features and a user-friendly interface, showcasing our technical excellence. The measurement system provided essential metrics with a professional looking GUI, and the Class D Amplifier Simulator implemented a complex amplification pipeline, demonstrating our ability to tackle challenging objectives. The Project Leader & Lead Developer ensured effective task coordination, while the Secretary & Project Documenter maintained clear records, aligning with the professionalism expected at the third-highest exigency level.

We are confident that our performance meets the third-highest level of exigency, deserving the grade we aimed for.

9. ANNEX

The ZIP folder submitted for the project contains the project code, along with both the programmer's manual and the user's manual for your reference.