Live Computer-Generated Visuals Based on the Sound Waves of Music

Nadia Abouelleil
18340116
B.A. (Mod.) Computer Science
Final Year Project, August 2023
Supervisor: Dr. Mads Haahr

Table of Contents

1. Abstract

In today's entertainment scene, the blend of sound and visuals is changing the way we experience music. This study delves into a unique web tool that crafts on-the-spot visuals by tapping into the very pulse of music: its sound waves. By doing so, it's not just about enhancing our music-listening journey; it's also about giving artists and music creators a fresh lens to see and shape their compositions. Through smart techniques and design choices, this tool aims to merge the auditory and visual, carving out a new avenue for music enthusiasts to explore and enjoy.

2. Acknowledgements

I wish to extend my profound appreciation to all who have been instrumental in the realisation of this Final Year Project.

At the forefront, my gratitude is directed towards my supervisor, Mads Haahr, whose guidance and unwavering support have been indispensable. My family and friends have consistently provided encouragement and backing, and for that, I also remain thankful.

Acknowledgement is also due to the architects behind HTML5, JavaScript technologies, and the P5, Meyda, and Chroma libraries, for their pivotal role in facilitating the technical aspects of this project. The vast array of tutorials and blogs available online have been a reservoir of knowledge, and I extend my gratitude to their contributors. I'd also like to express my appreciation to the open-source community for their collaborative spirit and invaluable contributions.

Lastly, the financial aid extended by the college and SUSI has been a cornerstone in my academic journey, enabling the completion of this Final Year Project and my broader academic pursuits.

In sum, this endeavour would not have reached its culmination without the collective contributions and support of all mentioned. I remain deeply thankful.

3. Introduction

In the contemporary realm of entertainment, the interplay between audio and visuals has become a prominent focal point. Despite its appeal, the real-time synchronisation of visuals with music poses significant challenges. This stems from the intricate technical skills required, coupled with the often-prolonged production process. Addressing this challenge, this Final Year Project unveils a web-based tool designed to automatically generate dynamic visuals in sync with musical sound waves. The primary aim is to provide users with the means to effortlessly produce and obtain real-time visual representations of their music.

This tool is founded on a web interface adept at producing live visuals that resonate with the subtle intricacies of musical sound waves. With a keen focus on user engagement, the application is geared towards intuitive design, enabling visuals that perfectly align with the music's rhythm and tempo. Moreover, users are granted the flexibility to customise their visuals, coupled with the functionality to download and share their creations. At its core, the application leverages technologies such as `meyda.js` for audio processing and `p5.sound.js` for visualisation, all meticulously integrated using JavaScript and HTML5. To guarantee a smooth user experience, extensive compatibility tests have been carried out across various web browsers.

The subsequent sections of this report delve into the realm of music visualisation, shedding light on the technological foundation that facilitates the creation of sound-driven visuals. This is succeeded by a thorough analysis of the web application, encapsulating its features, the architectural choices made, and the trajectory of its development. The application's efficiency, responsiveness, and user-centricity are later scrutinised through systematic user evaluations and feedback. The concluding sections envision enhancements to the tool's functionalities and suggest potential avenues where this innovation could be applied.

Through this initiative, a novel pathway emerges for music enthusiasts, enabling real-time visual interpretations. Such a tool bears profound implications for professionals like DJs, sound engineers, and music producers, simplifying the task of producing visuals that harmonise with music. On a larger scale, it paves the way for music aficionados to engage with and perceive music from a renewed, imaginative perspective.

4. Motivation

The inception of this final year venture emanates from my deep-seated intrigue in the cohesive integration of visual art and music. Throughout the annals of history, music has transcended mere auditory pleasure; it embodies a complex mosaic of feelings, recollections, and tales. Art, with its vast expressions, has portrayed these narratives. My family background, abundant with creative figures such as my mother, grandmother, and great-grandmother, has consistently inspired me. Their artistic journeys and zeal have harmonised with my scholarly ambitions, prompting me to incorporate imaginative facets into methodical studies. This union pays homage to my ancestry, celebrating the lineage of artists in my family who have exemplified the transformative essence of art.

Music, with its melodic symphonies and soul-stirring rhythms, has been a constant companion. It evokes a spectrum of emotions, from the depths of melancholy to the pinnacles of joy. A defining moment from my formative years that underscored the profound impact of this union was an art workshop I attended at the tender age of eight. Conducted by my mother, this workshop was not just an exercise in artistry; it was a lesson in synaesthesia. Tasked with translating the ethereal notes of classical music into visual masterpieces, I experienced first-hand the magic of merging sound with sight. It was a revelation, highlighting the boundless potential of this union. This project, at its core, seeks to replicate this magic, aiming to craft a unique digital experience that seamlessly blends the auditory and visual realms.

Delving deeper into the realm of sensory experiences, my sister's journey with synaesthesia added layers of depth to this project. Synaesthesia, a captivating neurological phenomenon, intertwines senses in a mesmerising dance. Observing my sister associate specific musical notes with vivid colours was an enlightening experience. The world, through her eyes, was a vibrant mosaic of sounds and colours. This project, with its objective of translating musical sound waves into real-time computer-generated visuals, promises a similar multisensory experience. Beyond just an innovative tool, it offers a deeper understanding of synaesthesia, potentially unravelling the mysteries of the human brain's complex sensory interpretations.

My personal escapades with music, marked by attending a multitude of concerts, have further cemented my belief in this project. A particularly poignant memory is the Tame Impala's Slow Rush Tour in 2022. This was not just a musical concert; it was an immersive experience, a symphony of sound, visuals, and emotions. The dynamic visual displays, intricately woven with the music, transported the audience to a different realm. The psychedelic aesthetics of King Gizzard & The Lizard Wizard's "The River" music video (King Gizzard & The Lizard Wizard, 2015) also served as a muse and as a guiding light for the project's direction.

This deep dive into the world of psychedelic visual art, often termed "acid trip" visuals, on platforms like YouTube, was enlightening. The serendipitous alignment of these visuals with various music tracks was a revelation, hinting at the immense untapped potential in this domain.

The entertainment industry, with its dynamic trends and evolving technologies, offers a promising avenue for this project. In an era where engagement is the key, the fusion of audio and visuals promises unparalleled experiences. This innovation not only redefines audience experiences but also offers emerging artists a unique medium for expression. Musicians, with real-time visual feedback, can harness this tool to steer their creative processes, potentially crafting masterpieces that resonate deeply with their audiences.

Dublin, with its bustling nightlife and eclectic mix of music and art, has been an instrumental factor in shaping my vision for this project. The city's vibrant nightclub scene offers a vivid tableau of how visuals can metamorphose an environment, adding layers of depth to the auditory experience. As a regular patron of these establishments, I've had the privilege of witnessing the alchemy of sound and visuals, creating atmospheres that transcend the ordinary. DJs and live performers, with their unique mixes and playlists, are at the heart of this experience. My conviction is that a web application, designed to curate visuals in real-time based on the nuances of sound waves, would be a game-changer for these artists. Such a tool would not only enhance their performances but would also redefine the clubbing experience for attendees, creating immersive atmospheres that resonate long after the night ends.

The endeavour to bring to life a cohesive and responsive music visualisation system demanded an extensive exploration of the existing literature in this domain. My journey was akin to navigating through a rich tapestry of research, methodologies, and technological innovations. This exploration was instrumental in shaping my understanding and approach to the project.

Central to my research were the seminal works of Julie Watkins, with her deep dive into "Composing Visual Music: Visual Music Practice at the Intersection of Technology, Audio-Visual Rhythms and Human Traces". Her exploration into the intersection of technology and human traces in visual music composition provided profound insights into the artistic nuances of the field. Additionally, the research titled "The Colour of Music: Real-time Music Visualisation with Synaesthetic Sound-Colour Mapping" illuminated the world of synaesthesia, demonstrating the potential of harnessing this phenomenon for real-time music visualisation. Lastly, the work of Max Graf, Harold Chijioke Opara, and Mathieu Barthet in "An Audio-Driven System For Real-Time Music Visualisation" served as a beacon, detailing the technical intricacies and methodologies for achieving real-time responsiveness in music visualisation.

These pivotal works in the literature not only enriched my knowledge but also charted the path for my project. Drawing inspiration and insights from these trailblazers, I stand ready to critically evaluate the web application, with an emphasis on its efficiency, user experience, and adaptability.

The overarching ambition of this project is twofold. Firstly, to develop a robust web application that seamlessly merges functionality with user experience. Users, with their diverse preferences and tastes, should find the platform both intuitive and adaptable. Leveraging the capabilities of JavaScript and HTML5, the application is designed for broad compatibility, ensuring a consistent experience across various web browsers. The rigorous testing phases, coupled with comprehensive evaluations, are geared towards achieving optimal performance and high user engagement. Feedback loops are integral to the project's design, ensuring that the application is dynamic and evolves based on user inputs and technological advancements.

In essence, this project is a celebration of the age-old dance between sound and visuals. It aims to create a digital platform where every beat, every note, and every melody finds its visual counterpart, creating a symphony of experiences. While its foundation is technical, its soul is artistic. It seeks to empower artists, enhance entertainment experiences, and offer users a canvas where their musical choices paint vivid visual stories. At its heart, this endeavour is driven by a profound love for music and art, a passion for innovation, and a desire to create something that stands at the intersection of technology and creativity.

5. Literature Reviews

5.1. The Colour of Music: Real-time Music Visualisation with Synaesthetic Sound-Colour Mapping by Kia Ng, Joanne Armitage, and Alex McLean (2014)

Within the field of music visualisation, the complex relationship between auditory inputs and their subsequent visual manifestations has been a focal point of academic enquiry. A key piece in this area is "The Colour of Music: Real-time Music Visualisation with Synaesthetic Sound-Colour Mapping". Central to this study is the exploration of synaesthesia, a neurological phenomenon where a sensory or cognitive experience triggers an involuntary secondary sensory or cognitive response.

The article begins by clearly introducing synaesthesia, illustrating a realm where particular auditory cues, such as musical pitches, can induce specific visual experiences, like colours. The authors investigate this not simply as a neurological phenomenon but as a potential foundation for real-time music visualisation. The aim extends beyond producing accompanying visuals for music; it seeks to offer a genuinely synaesthetic experience, where visuals are inherently linked to the auditory cues perceived.

As the study progresses, the authors shed light on various methodologies and algorithms pivotal to the real-time visualisation process. The intricacies involved in real-time visualisation, especially translating the rich layers of music into precise visual counterparts, are candidly addressed. Yet, the authors adeptly discuss the techniques they utilised, ensuring the visual displays were not merely precise but also emblematic of an authentic synaesthetic experience.

A distinguishing feature of this article is its emphasis on user interaction. The authors underline that the measure of a visualisation tool isn't solely its technical merit, but its capacity to engage its audience. In this context, they delve into user feedback and the consequent refinements made to the system. This approach underscores the authors' dedication to crafting a system that's not only technically robust but also emotionally engaging.

To conclude, "The Colour of Music: Real-time Music Visualisation with Synaesthetic Sound-Colour Mapping" remains a seminal work in the realm of music visualisation. It introduces an innovative approach, leveraging synaesthesia to curate a truly enveloping auditory-visual journey. For those at the intersection of music, technology, and visual art, this study offers essential perspectives, acting as both a reference and a source of motivation.

5.2. Composing Visual Music: Real-time Music Visualisation with Synaesthetic Sound-Colour Mapping by Julie Watkins (2018)

The intricate relationship between visual art and music has been a subject of artistic exploration and academic scrutiny for centuries. In her insightful work, "Composing Visual Music: Visual Music Practice at the Intersection of Technology, Audio-Visual Rhythms and Human Traces", Julie Watkins plunges into this interface, shedding light on how technology, audio-visual rhythms, and human influences converge in the realm of visual music composition.

Watkins embarks on her exploration by delineating the historical trajectory of visual music. She eloquently traces its evolution, illustrating how technological advancements have constantly reshaped the way artists and composers perceive and represent this art form. By rooting her discussion in history, Watkins effectively showcases the continuity and changes in visual music composition over time.

Central to Watkins' discourse is the notion of 'human traces'. She posits that while technology and algorithms play a pivotal role in visual music composition, the human touch – the emotions, experiences, and subjectivities of the composer – remains indispensable. It's these 'human traces' that infuse visual music compositions with depth, making them resonate with audiences on an emotional and cognitive level.

Watkins delves deep into the methodologies employed in visual music composition. She outlines various techniques, discussing the potential of software tools, real-time processing, and interactive elements. The challenge, as Watkins aptly points out, lies in achieving a harmonious balance between the auditory and visual components, ensuring that neither overshadows the other. Her emphasis on the interplay of technology and human intervention offers a fresh perspective, highlighting the symbiotic relationship between the two in the realm of visual music composition.

The paper also touches upon the challenges inherent in visual music composition. Watkins discusses the intricacies of integrating technology, ensuring real-time responsiveness, and catering to the diverse preferences of audiences. However, despite these challenges, Watkins remains optimistic about the future of visual music, especially with the rapid advancements in technology.

In summation, Julie Watkins' "Composing Visual Music: Visual Music Practice at the Intersection of Technology, Audio-Visual Rhythms and Human Traces" is a seminal contribution to the literature on visual music composition. By seamlessly weaving together historical context, technical discussions, and artistic considerations, Watkins offers a holistic view of the field. Her work stands as both a testament to the rich legacy of visual music and a beacon guiding its future trajectory.

5.3. An Audio-Driven System for Real-Time Music Visualisation by Max Graf, Harold Chijioke Opara, and Mathieu Barthet (2021)

In the ever-evolving domain of music visualisation, the quest for real-time responsiveness and seamless integration of audio and visuals remains paramount. The collaborative effort of Max Graf, Harold Chijioke Opara, and Mathieu Barthet in "An Audio-Driven System for Real-Time Music Visualisation" offers a cutting-edge exploration into this challenge, presenting a system that is as technically robust as it is artistically evocative.

At the outset, the authors emphasise the growing need for real-time music visualisation systems, particularly in live performance settings. In an era where audience engagement hinges on multisensory experiences, the ability to generate visuals that not only complement but also elevate the auditory experience is crucial. Graf, Opara, and Barthet's work is set against this backdrop, aiming to bridge the gap between the technical and the artistic.

The core of the paper delves into the intricacies of the audio-driven system they developed. With a meticulous approach, the authors outline the technical framework, detailing the algorithms and methodologies employed. The system's prowess lies in its ability to capture the nuances of music — be it tempo, pitch, or rhythm — and translate them into dynamic visuals in real-time. The authors provide insights into the challenges faced, particularly in ensuring accuracy and responsiveness, and the solutions implemented to overcome them.

But beyond the technical, the paper also ventures into the realm of the artistic. The authors discuss the importance of crafting visuals that resonate emotionally with audiences, ensuring that the visual output is not just a mechanical reflection of the audio input but an artistic interpretation that adds depth to the musical experience.

An intriguing aspect of the research is the emphasis on user feedback. Recognising that the ultimate success of any visualisation system lies in its ability to resonate with its users, the authors incorporated iterative feedback loops, refining the system based on user inputs and preferences. This approach not only enhanced the system's effectiveness but also ensured its adaptability to diverse musical genres and contexts.

In conclusion, "An Audio-Driven System for Real-Time Music Visualisation" by Graf, Opara, and Barthet stands as a testament to the possibilities at the intersection of technology and art. By marrying technical expertise with artistic sensibility, the authors present a system that promises to revolutionise the world of music visualisation. Their work offers invaluable insights for artists, technicians, and researchers, pushing the boundaries of what is possible in real-time music visualisation.

6. Objectives & Aims

6.1. Introduction

The heart of any final year project lies not merely in the technology it employs or the idea it showcases, but in the clear purpose it signifies. Set at the juncture of art and technology, this project endeavours to transform how users engage with music. It aims to move beyond music as a purely auditory delight, presenting it as a multisensory voyage. By weaving visual patterns in tandem with musical sound waves, the project seeks to heighten user involvement and furnish a truly immersive experience. Below, we elucidate the principal and subsidiary objectives that directed this project's evolution.

6.2. Primary Objective

To design a web-based application that instinctively crafts real-time visuals influenced by the sound waves of music, granting users a compelling amalgamation of sound and sight. The application must discern the nuances of the musical piece and provide avenues for user tailoring, ensuring a unique and intimate experience for each individual.

6.3. Secondary Objectives

- **Technical Objectives**:
  - Develop a fluid application which is harmonious with major web browsers, broadening user reach.
  - Integrate algorithms, as reflected by the incorporation of the Meyda.js library, to precisely discern musical facets such as frequency and amplitude, transforming them into congruent visual motifs.
- **Functional Objectives**:
  - Design a straightforward interface, as observed in the project's frontend code, enabling users to effortlessly upload or choose their preferred music tracks.
  - Introduce customisation alternatives allowing users to dictate the visual aesthetics, in sync with the rhythms and melodies of the music.
  - Prioritise flawless synchrony between the generated visuals and the ongoing musical piece.
- **User Experience Objectives**:
  - Conceptualise an interface that strikes a balance between aesthetic appeal and user-friendliness, hastening the adoption process for new users.
  - Strive for undisturbed application performance, curtailing latency or other visual interruptions during the visualisation journey.

6.4. Challenges and Constraints

- **Technical Intricacy**: Crafting a web tool that dynamically renders visuals based on music encompasses notable technical intricacies, especially in ensuring prompt responsiveness and astute musical interpretation.
- **Browser Harmony**: Ascertaining consistent behaviour across diverse web browsers, whilst harnessing sophisticated visualisation methods, adds an additional tier of intricacy.
- **User Personalisation**: Balancing user personalisation options with the application's performance and aesthetic charm remains a

formidable challenge.

- **Performance**: Rendering visuals derived from sizable music files in real-time, particularly on devices with constrained processing capabilities, is a notable concern.
- **Musical Spectrum**: Given the vast diversity in tempo, genre, and structural complexity in music, ensuring the visualisation apparatus is versatile remains a pivotal challenge.

6.5. Measures of Success

- **Performance Metrics**: The tool should manifest visuals with negligible delay, proffering an uninterrupted experience.
- **User Engagement**: Elevated user interaction and affirmative feedback would signify the application's resonance with its target demographic.
- **Technical Stability**: The software should gracefully manage a broad spectrum of musical inputs without faltering or incurring pronounced performance hindrances.
- **Customisation Engagement**: Monitoring the frequency of user interactions with the customisation features can shed light on its allure and success.
- **Cross-Browser Accord**: Seamless operation across predominant web browsers without pronounced functional or visual discrepancies is crucial.

The objectives delineated for this project aspire to intertwine the spheres of music and visual artistry, offering users a distinctive platform to perceive music from a fresh perspective. While the challenges are pronounced, they pave the way for ingenuity and inventive problem-solving. The success metrics act as a beacon, ensuring the project remains true to its foundational vision. As we delve into the subsequent segments detailing the approach, design, and actualisation, these objectives and challenges will form the bedrock, steering every resolution and tactic.

7. Methodology

7.1. Introduction

In the field of computer science and digital arts, the methodology serves as the bedrock upon which the project stands, guiding its progression from inception to completion. For this final year project, a methodological framework was devised to seamlessly intertwine the realms of musical acoustics with real-time graphical representation.

The project leaned on tried-and-tested development practices, aiming to transmute complex musical sound waves into visually arresting patterns within a web-based environment.

Several factors underpinned the selection of this methodology. At the forefront was the intricate nature of the project, which demanded a well-organised approach. Melding auditory nuances with dynamic visualisations is a challenging endeavour. It calls for a profound comprehension of both music and graphics, coupled with the adeptness to synergise them proficiently. Furthermore, with the project's emphasis on user engagement, the methodology was tailored to accentuate the user experience. This meant striving beyond mere functional efficiency, ensuring that the interface was intuitive, agile, and enthralling.

Given the ever-shifting landscape of digital technology and the fluidity of artistic paradigms, it was paramount to embrace a methodology that was both adaptable and steadfast. Such an approach guaranteed that despite having a lucid project trajectory, there remained ample scope for adjustments, informed by real-time challenges and insights gleaned during the developmental stages.

Subsequent sections will delve deeper into the specific research and developmental strategies employed. They aim to chart the project's odyssey, from its embryonic research stages to its culminating avatar as a web-based application. By shedding light on the methodological underpinnings, this report aspires to offer insight into the rationale behind decisions, hurdles navigated, and the ingenious solutions devised to actualise this ambitious project.

7.2. Research Approach

Central to this project was a rigorous research phase. Before delving into the technical implementation, it was essential to grasp the nuances of music visualisation, its historical trajectory, prevailing technologies, and potential avenues for innovation.

7.2.1. Literature Review:

A thorough literature review is tantamount to navigating with a map in uncharted territories. The exploration of scholarly works, including seminal pieces by Ng, Armitage, McLean, Watkins, and Barthet, illuminated the evolution of music visualisation. This scholarly immersion not only offered a historical perspective but also pinpointed the methodologies, challenges, and novel solutions within this domain. Furthermore, this academic foray spotlighted potential niches, suggesting areas where this project might bring fresh insights or advancements.

7.2.2. Visual Inspiration:

Visuals reminiscent of psychedelic art, colloquially known as "acid trip" visuals, played a pivotal role in shaping the project's direction. Platforms like YouTube emerged as invaluable, offering a treasure trove of such dynamic visuals. Immersing in these videos elucidated the visual flair and rhythm that music could potentially evoke. This aesthetic immersion served as a beacon, moulding the visual and rhythmic facets of the project.

7.2.3. Design Iterations:

To transpose these inspirations into tangible designs, an iterative approach was adopted. Initial visual sketches were conceived, acting as a visual charter for the project. These sketches, an amalgam of academic insights and psychedelic inspirations, delineated the project's envisioned outcome. They steered the development, ensuring it resonated with both the project's creative aspirations and technical benchmarks.

7.2.4. Online Resources

For the development of specific visuals, I referred to multiple sources:

- To build a foundation for my work, I delved into several comprehensive tutorials and guides. The p5.js tutorial series by The Coding Train provided an invaluable grounding in the nuances of the library (The Coding Train, date not provided). Additionally, the article "Audio Visualization in JavaScript with p5.js" on Medium offered profound insights into the implementation nuances of audio visualisation using p5.js (Nishanc, date not provided).
- The project greatly benefited from YouTube tutorials, such as the "JavaScript Audio CRASH COURSE For Beginners" by Franks Laboratory, which elucidated core concepts behind audio processing in JavaScript (Franks Laboratory, date not provided).
- The intricate Mandala visuals owe their genesis to the "Pulsating Mandala" sketch by Rodrigo Siqueira on OpenProcessing (Siqueira, date not provided).
- The flow field visuals were primarily crafted with guidance from Johan Karlsson's "Particles in a Simplex Noise Flow Field" on CodePen (Karlsson, date not provided).

To encapsulate, the research modus operandi was a blend of scholarly delving, aesthetic exploration, and iterative design. This triad ensured that the ensuing development phase was both enlightened and inventive, culminating in a solution that was as avant-garde as it was user focused.

7.3. Development Approach

Moving from conceptualisation to actualisation necessitated a seamless integration of research insights with practical execution. With a repository of knowledge, inspirations, and a lucid blueprint, the developmental phase pivoted around transmuting these components into a dynamic web application. This transformation was facilitated through methodical steps:

7.3.1. Technological Foundation

Central to the project was a judicious selection of technologies, harmonised to facilitate real-time music visualisation. JavaScript, in conjunction with HTML5, was chosen due to its adaptability, widespread adoption, and prowess in generating dynamic visuals for web interfaces. This choice was further solidified by JavaScript's vast ecosystem, inclusive of libraries like `p5.js` and `Meyda.js`, which were instrumental in sculpting the visualisation mechanics.

The `p5.js` library, in particular, was favoured for its inherent simplicity and extensive documentation, enabling the rendering of graphics synchronised with music. On the other hand, `Meyda.js` was vital in extracting intricate audio features, providing the data driving the visual patterns. The use of HTML5 ensured universal accessibility across a spectrum of web browsers, making it the linchpin of the project's web compatibility.

7.3.2. Cyclical Development:

Development in the digital arts spectrum is seldom a straightforward trajectory. An iterative methodology was championed, where features were incrementally developed, tested, and refined. Functions like '`setup()`' and '`draw()`' from the `p5.js` library were foundational, with 'setup()' configuring the initial environment and '`draw()`' perpetually updating visuals. Modular development was further facilitated by the use of distinct functions and event listeners, ensuring each component was meticulously crafted and evaluated.

7.3.3. Visual Integration:

Drawing inspiration from the research phase and preliminary sketches, the code incorporated specific visual patterns. The `p5.js` library, with its robust graphical functionalities, was pivotal in this endeavour. `Meyda.js`, on the other hand, handled audio feature extraction, supplying the necessary metrics to inform the generated visuals. Together, these tools translated the envisioned dynamism and rhythm into tangible visualisations.

7.3.4. Performance Fine-tuning:

Given the emphasis on real-time visual generation, performance optimisation was paramount. The visual rendering process was meticulously refined to ensure fluidity, even with intricate musical inputs. Special attention was directed towards streamlining computational tasks, ensuring a seamless user experience devoid of lags.

7.3.5. Cross-Platform Compatibility:

In today's digitised milieu, web applications are accessed via a plethora of devices and browsers. Ensuring that the application remained agnostic to these variables was vital.

Rigorous tests across browsers ascertained a uniform user experience. Particular care was taken to avoid JavaScript features that might not be universally supported, ensuring broad compatibility.

7.4. Challenges and Problem-Solving

7.4.1. Real-time Visual Rendering:

Challenge: *Capturing and representing the ever-evolving nuances of music in visual form in real-time.*
Solution:

- The code utilises `p5.sound.js`'s FFT object. The method `fft.analyze()` is specifically invoked to extract frequency spectrum data, which then influences the visual components.
```
spectrum = fft.analyze();
```
- The inherent looping nature of the `p5.js draw()` function ensures the visual elements are continually updated, offering a real-time representation of the ongoing audio.

7.4.2. Diverse Music Inputs:

Challenge: *Adapting visuals to a wide gamut of musical inputs, ensuring the visuals are evocative of the audio's characteristics.*
Solution:

- Meyda library has been incorporated to derive detailed audio features. The method `meydaAnalyzer.get('rms')` is employed to extract the audio's root mean square (RMS) value, a measure of its magnitude.
```
let rms = meydaAnalyzer.get('rms');
```
- Depending on the audio features, the `draw()` function in the code adjusts the visual renderings, ensuring they reflect the audio's distinct attributes.

7.4.3. Cross-Browser Compatibility:

Challenge: *Delivering a consistent application experience across a multitude of browsers.*
Solution:

- Relying on `p5.js`, a widely recognised library, ensures that the visuals rendered on the canvas remain consistent across varied browsers. The library's extensive testing across platforms minimises unexpected behaviours.
- The code's simplicity, evident from the straightforward HTML structure and well-established JavaScript libraries, further reduces potential browser-specific issues.

7.4.4. User Interface & Experience:

Challenge: *Balancing between an interactive user interface and the demands of dynamic visual rendering.*
Solution:
- The foundation for the interactive user experience is set up in the `setup()` function of `p5.js`, where the visual canvas and audio input mechanisms are initialised.
  ```
  createCanvas(windowWidth, windowHeight);
  ```
- The dropdown, facilitated by `<select id="visualSelector">` in the HTML, empowers users to select their preferred visualisation style, enhancing their agency and engagement with the application.

7.4.5. Performance Optimisation for Visuals:

Challenge: *Sustaining smooth visual transitions, particularly during intricate or rapid musical shifts, without compromising performance.*
Solution:
- By compartmentalising visual rendering within the `draw()` function, the code maximises computational efficiency. Adjustments to the visuals based on the audio data are made without redundant recalculations.
- The `flowWorker.js` file is of particular note. As a Web Worker, it handles some of the more intensive visual computations in a separate thread, ensuring the main interface remains agile and responsive.
  ```
  const worker = new Worker('flowWorker.js');
  ```

7.5. Evaluation Method

The efficacy of any project, particularly in the domain of computer science and digital arts, is contingent upon a robust evaluation process. This project's objective to interweave the nuances of music with real-time graphical representation on a web platform necessitates an evaluation method that is both systematic and comprehensive.

7.5.1. Functional Testing:
At the heart of this evaluation was a rigorous functional testing phase. By interacting with the user interface elements present in the index.html file, such as the `<select id="visualSelector">` dropdown, the project's visualisation styles were assessed in real-time. The goal was to ensure that user selections accurately triggered the corresponding visual patterns, as defined within the `script.js` file.

7.5.2. Performance Evaluation:
Given the real-time nature of the visualisations, performance became paramount. The

project was subjected to a myriad of musical inputs, ranging in complexity, tempo, and rhythm. The objective was to ascertain the fluidity of the visual transitions, particularly during rapid musical shifts. Special attention was paid to the integration with `flowWorker.js`, ensuring that the offloading of intensive visual computations to a Web Worker resulted in a perceptibly smoother user experience.

7.5.3. Cross-browser Compatibility Testing:

With web applications being accessed from an array of browsers, cross-browser compatibility was integral to the evaluation. The application was tested across popular browsers to ascertain consistency in both functionality and visual fidelity. This ensured that features like the canvas rendering, facilitated by `p5.js`, remained consistent regardless of the browser in use.

7.5.4. User Experience Assessment:

The project's intuitive interface, as depicted in the index.html, played a pivotal role in user engagement. The evaluation involved assessing the ease with which users could select their preferred visualisation style and interact with the audio input mechanisms. Feedback on the user interface's intuitiveness and the overall user-centric design was actively sought and analysed.

7.5.5. Code Review and Optimisation:

A meticulous review of the `script.js` file was undertaken to ensure code efficiency and maintainability. This involved assessing the modular structure, the use of functions and event listeners, and the integration with external libraries. The goal was to ensure that the code was not only functional but also adhered to best practices, ensuring scalability and ease of future enhancements.

7.6. Challenges & Decisions

Throughout the project's development, various challenges emerged. A significant hurdle was ensuring the real-time processing of audio without any latency. This necessitated rigorous optimisation, especially concerning particle generation and rendering. Furthermore, the selection of the right audio features for visualisation was paramount. After comprehensive experimentation, RMS was chosen owing to its lucid representation of audio amplitude.

7.7. Conclusion

In essence, the evaluation method was multifaceted, encompassing both technical and user-centric assessment criteria. By subjecting the project to this comprehensive evaluation, the aim was to ensure that the final product was not only technically sound but also offered an unparalleled user experience, truly bringing the magic of music to life through visuals.

The intricate dance between music and visual arts, especially within the confines of a digital platform, poses multifarious challenges. This project, set against this backdrop, endeavoured to bridge the auditory and visual realms, striving for a harmonious synthesis that would captivate the user. The methodological approach detailed in this report provided the guiding principles and strategic direction necessary for the realisation of this ambitious objective.

Grounded in a robust research foundation, the project embarked on its journey by immersing itself in academic literature, visual inspirations, and iterative design exercises. This preparatory phase, which leveraged resources like YouTube and seminal works from pioneers in the music visualisation domain, ensured that the project was both theoretically sound and aesthetically attuned.

The transition from research to execution was underpinned by a strategic selection of technologies. With the `script.js` file at its core, the project harnessed the capabilities of JavaScript and its rich ecosystem, specifically libraries such as `p5.js` and Meyda. These technical choices, as evident in the code, were not mere happenstance but were informed decisions aimed at achieving seamless real-time music visualisation on the web. The HTML structure, as laid out in index.html, complemented this, offering an intuitive user interface that was both engaging and user-friendly.

Yet, as with any ambitious endeavour, challenges were inevitable. From ensuring real-time visual rendering to accommodating diverse musical inputs and guaranteeing cross-browser compatibility, the project navigated a labyrinth of technical complexities. However, the methodological approach, with its emphasis on cyclical development, performance fine-tuning, and problem-solving, ensured that these challenges were not just addressed but were transformed into opportunities for refinement and innovation.

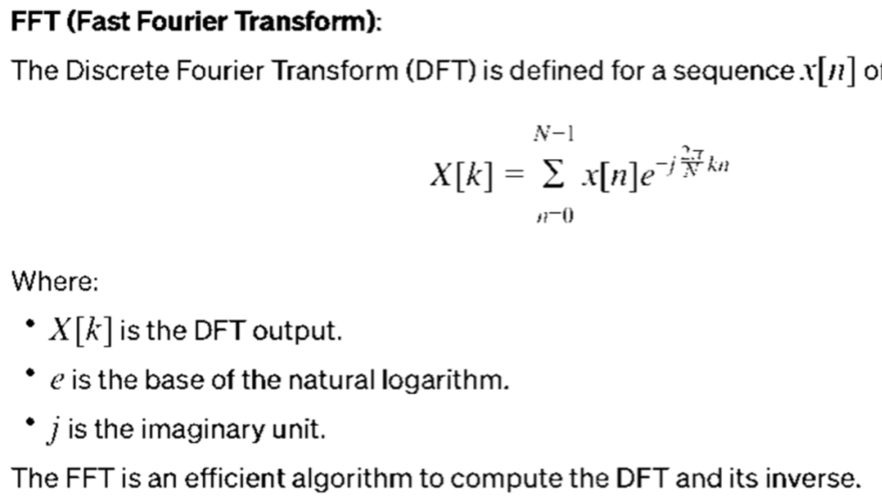
In conclusion, the methodology adopted for this project, as mirrored in the `script.js` and index.html files, was both its compass and its anchor. It provided direction, lent stability, and above all, ensured that the project's vision - to create a digital space where music and visuals converge in real-time harmony - was brought to fruition. Through this approach, the project stands as a testament to what can be achieved when sound methodological principles are applied to the dynamic and ever-evolving world of computer science and digital arts.

8. System Design

8.1. Introduction

The objective of this project, 'Live Computer-Generated Visuals Based on the Sound Waves of Music', is to craft a digital ecosystem that can take sound waves, particularly from music, and transform them into captivating and responsive visual displays in real-time. The system is architected leveraging a combination of contemporary JavaScript libraries and sound analysis techniques, providing users with an immersive experience that bridges the auditory and visual senses. This section elucidates the intricate design and the interactions of the components within the system.

Frequency Analysis
In the realm of audio analysis, the Fast Fourier Transform (FT) stands as a pivotal technique. It facilitates the conversion of a signal from its time domain to its frequency domain, rendering it particularly invaluable for spectral analysis. A seminal work by Cooley & Tukey (1965) introduced an algorithm for the machine calculation of complex Fourier series, laying the groundwork for many subsequent advancements in the field.



FFT (Fast Fourier Transform):

The Discrete Fourier Transform (DFT) is defined for a sequence $x[n]$ of length N as:

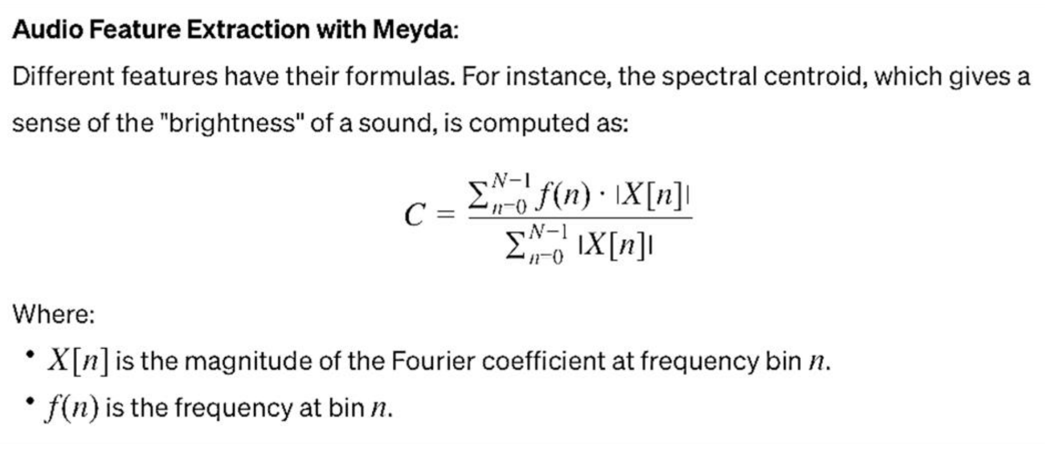
$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi}{N}kn}$$

Where:

- $X[k]$ is the DFT output.
- e is the base of the natural logarithm.
- j is the imaginary unit.

The FFT is an efficient algorithm to compute the DFT and its inverse.

Feature Extraction
For more nuanced audio visualisations, extracting intricate audio features becomes paramount. Utilising tools like Meyda, one can discern features like chroma or spectral centroid from audio streams. Müller (2015) delves deep into the fundamentals of music processing, offering insights valuable for such feature extraction endeavours. Additionally, the official documentation of Meyda serves as a practical guide for leveraging its capabilities.



Audio Feature Extraction with Meyda:

Different features have their formulas. For instance, the spectral centroid, which gives a sense of the "brightness" of a sound, is computed as:

$$C = \frac{\sum_{n=0}^{N-1} f(n) \cdot |X[n]|}{\sum_{n=0}^{N-1} |X[n]|}$$

Where:

- $X[n]$ is the magnitude of the Fourier coefficient at frequency bin n.
- $f(n)$ is the frequency at bin n.

8.2. High-Level System Overview

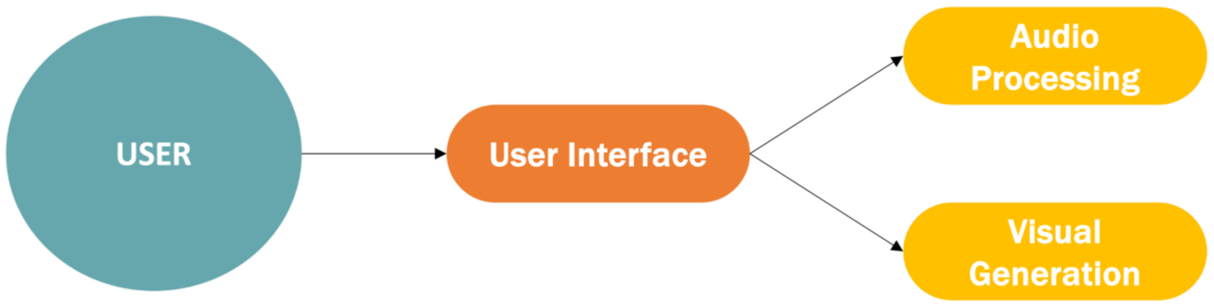


Figure 8.1: High-Level System Diagram

The system is constructed around a central web interface, primarily built using HTML and JavaScript. The functionality can be categorised into two primary domains: sound wave capture and analysis, and visual generation based on this analysis.

The choice of a web interface was made to ensure accessibility and ease of use for a wide range of users. With the ubiquity of web browsers and the universal compatibility they offer, a web-based approach ensures that users can experience the visual representations without the need for specialised software. The web interface addresses the user's need for a straightforward and intuitive platform where they can simply input music and immediately experience the dynamic visuals. Moreover, the web-based approach offers scalability and potential for future enhancements, such as integration with streaming platforms or social media sharing capabilities.

In terms of the libraries and frameworks employed, choices were made based on functionality, community support, and ease of integration. The `p5.js` library, renowned for its capabilities in creative coding, offers a rich set of features for drawing and animation on the web. It also integrates seamlessly with its sound module, `p5.sound.js`, providing a cohesive environment for sound analysis and visual generation. On the other hand, `meyda.js` was chosen for its specialised audio feature extraction capabilities. While other libraries exist, `meyda.js` stood out due to its comprehensive feature set, performance, and active community support. The combination of these libraries allowed for a streamlined development process and a robust system that can effectively translate sound waves into intricate visuals.

System Architecture Diagram

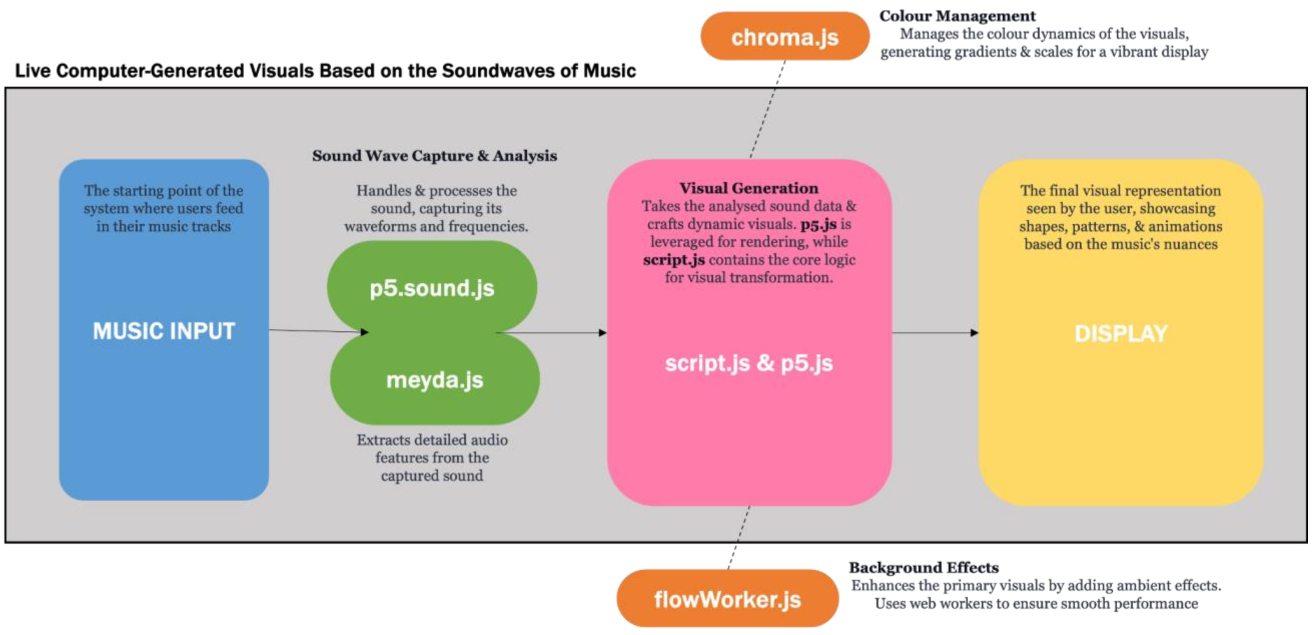

Colour Management
Manages the colour dynamics of the visuals, generating gradients & scales for a vibrant display
chroma.js
Live Computer-Generated Visuals Based on the Soundwaves of Music
Sound Wave Capture & Analysis
The starting point of the system where users feed in their music tracks
MUSIC INPUT
Handles & processes the sound, capturing its waveforms and frequencies.
p5.sound.js
meyda.js
Extracts detailed audio features from the captured sound
Visual Generation
Takes the analysed sound data & crafts dynamic visuals. p5.js is leveraged for rendering, while script.js contains the core logic for visual transformation.
script.js & p5.js
The final visual representation seen by the user, showcasing shapes, patterns, & animations based on the music's nuances
DISPLAY
flowWorker.js
Background Effects
Enhances the primary visuals by adding ambient effects. Uses web workers to ensure smooth performance

Figure 8.2: System Architecture Diagram

Component Description

Web Interface (index.html):

- This is the gateway through which users interact with the system. It is structured using HTML and provides the necessary controls and display area for the visuals. The interface is clean and intuitive, enabling users to easily input music and view the corresponding visuals.

Sound Wave Analysis (p5.sound.js, meyda.js):

- Once the music is fed into the system, these libraries capture the sound waves and break them down into various metrics and features, such as amplitude, frequency, and spectral content. `meyda.js` is particularly instrumental in extracting audio features, while `p5.sound.js` aids in handling and processing the sound.

Visual Generation (script.js, p5.js):

- Based on the analysed sound wave data, the system dynamically generates visuals. `script.js` contains the logic and algorithms to decide the type, behaviour, and characteristics of the visuals. The `p5.js` library is leveraged to render these visuals, drawing shapes, patterns, and animations that correspond to the nuances of the music.

Background Effects (flowWorker.js):

- This component adds a layer of ambient visual effects to the main visuals, enhancing the depth and aesthetics of the display. It uses web workers to offload some of the computational tasks, ensuring that the main thread remains unblocked, and the visuals are displayed smoothly.

Colour Management (chroma.js):

- To make the visuals more vibrant and engaging, the system employs the `chroma.js` library. It aids in generating gradients, managing colour scales, and ensuring that the visuals are not only reactive to the music but also aesthetically pleasing.

The harmonious interplay of these components yields a system that not only deciphers the intricacies of music but also translates them into a visual symphony.

8.3. Sound Wave Analysis

The process of transforming music into visuals begins with the intricate task of sound wave analysis. This procedure entails capturing the nuances of sound waves and quantifying them into metrics and features that can be processed and visualised.

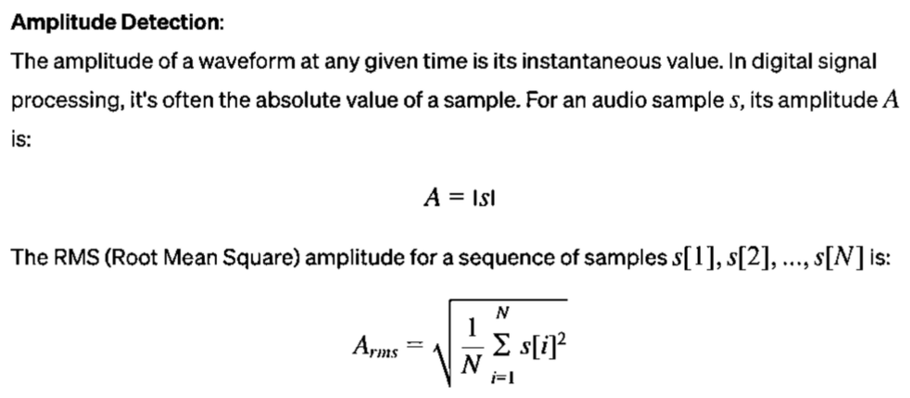
8.3.1. Sound Wave Capture:

Utilising the `p5.sound.js` library, the system intercepts the audio stream of the music input by the user. This library, an extension of `p5.js`, offers a comprehensive suite of methods for sound processing. The audio stream, when played, produces continuous waveforms and frequencies, which are the foundational data for our subsequent analysis.

8.3.2. Feature Extraction:

Post the sound capture, `meyda.js` steps in to dissect the sound wave into distinct audio features. Meyda is an audio feature extraction library tailored for web applications. For this project, some of the pivotal features extracted include:

- **Amplitude**: Gauges the loudness or volume of the sound at any given instant. A critical component in shaping the dynamics of computer-generated visuals is the amplitude of audio signals. By detecting amplitude variations, one can modulate visual elements in sync with the music's intensity. Oppenheim & Schafer (2010) provide a comprehensive overview of discrete-time signal processing, which serves as an essential reference for such amplitude-based visual manipulations.



Amplitude Detection:

The amplitude of a waveform at any given time is its instantaneous value. In digital signal processing, it's often the absolute value of a sample. For an audio sample s, its amplitude A is:

$$A = |s|$$

The RMS (Root Mean Square) amplitude for a sequence of samples $s[1], s[2], ..., s[N]$ is:

$$A_{rms} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} s[i]^2}$$

- **Frequency**: Determines the pitch or tone of the sound.
- **Spectral Content**: Offers insights into the distribution of the sound's frequency content.

These features provide a comprehensive snapshot of the sound's character, facilitating the translation of its essence into visuals.

Visual Generation

Once the sound waves have been analysed and their features extracted, the ensuing step is to generate the visuals that mirror the music's essence.

```
function draw() {
 background(backgroundColor);

    ... [code body] ...

 switch (shape) {
   case 'simpleWaveform':
     drawWaveform(spectrum);
     break;

    ... [other cases] ...

 }
}
```

8.3.3. Visual Logic and Algorithms:

The `script.js` file serves as the heart of the visual generation. It ingests the analysed data and employs a series of algorithms to dictate the nature, behaviour, and characteristics of the visuals. These algorithms account for the amplitude, frequency, and other extracted features to decide on visual parameters such as size, shape, motion, and colour.

8.3.4. Rendering with p5.js:

To bring the visual concepts to life, the system utilises the `p5.js` library. It's a dynamic graphical library that aids in creating interactive visuals in a web environment. This library manages the drawing of shapes, patterns, and animations on the web canvas, thus materialising the visual representation of the music.

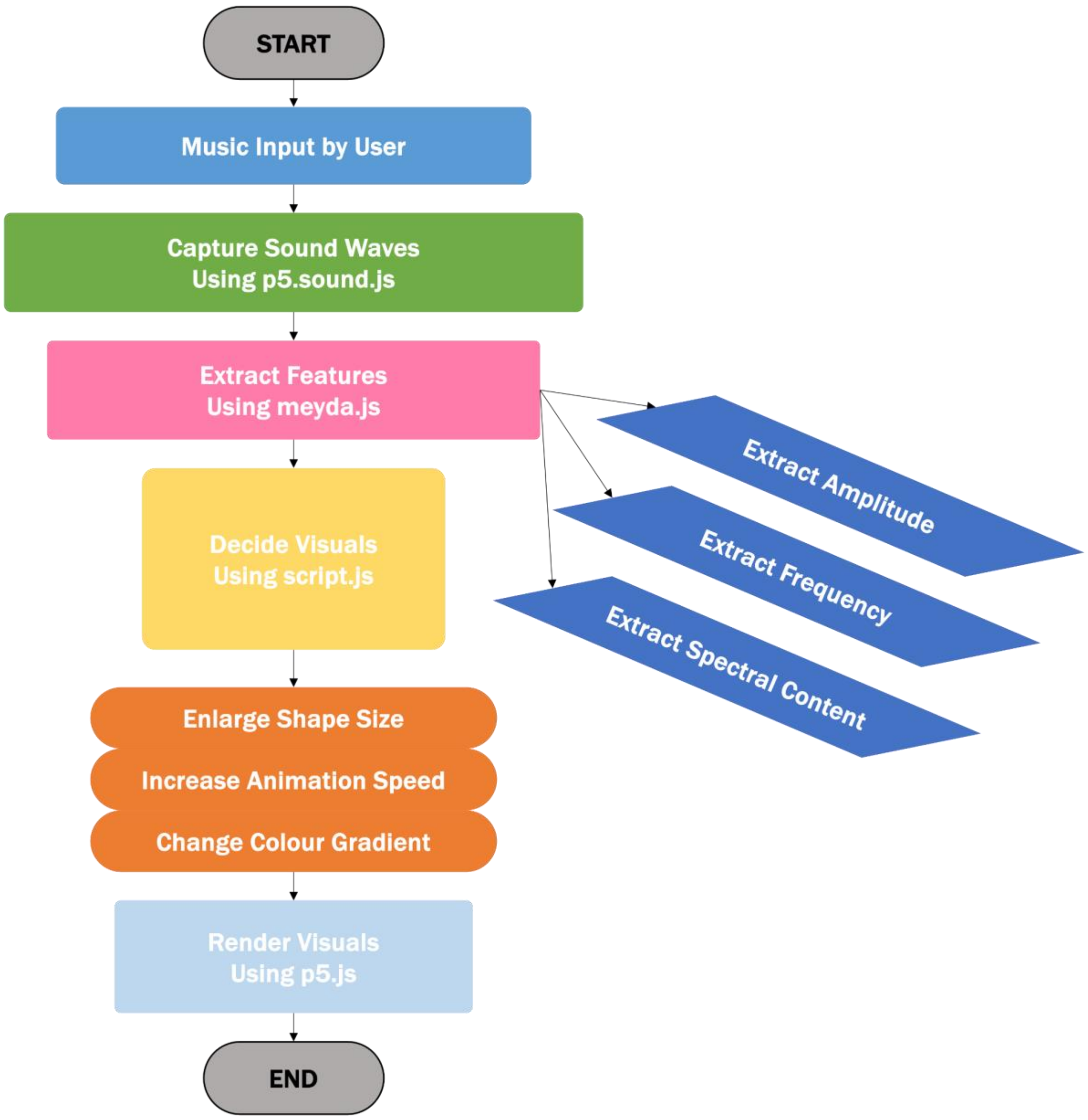


Figure 8.3: Sound Wave Analysis to Visual Rendering Flowchart

8.4. User Interface and Interaction

The user interface is the bridge between the system's intricate backend processes and the user. Ensuring its intuitiveness and responsiveness is vital for a seamless user experience.

8.4.1. Web Interface:

The system's gateway, as seen in `index.html`, is designed to be user-friendly. It provides straightforward controls for music input and offers a display area for the resulting visuals. This minimalist approach ensures that the user's focus remains on the music and its visual counterpart.

8.4.2. User Controls:

While the system autonomously generates visuals based on the music, it also provides users with some level of control. They can:

- **Input Music**: Users can feed their choice of music into the system.
- **Play/Pause**: Controls to manage the playback of the music.
- **Visual Settings**: Although not explicitly present, potential enhancements could include controls to adjust visual parameters like colour schemes or visual intensity.

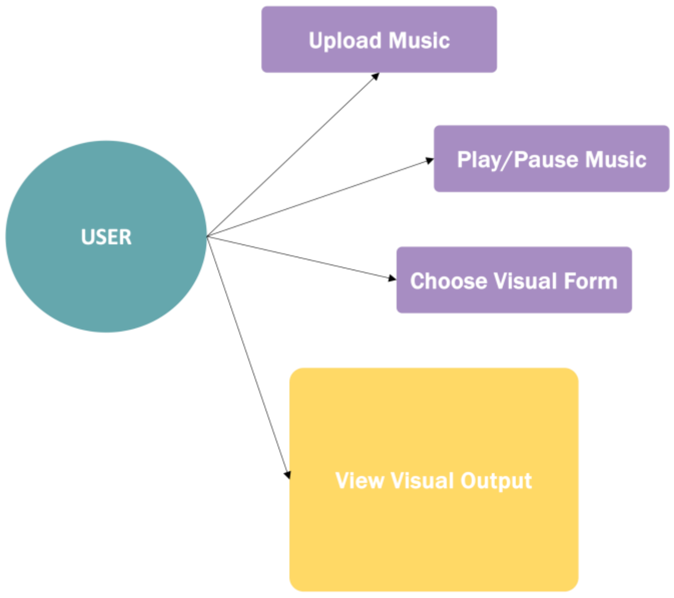


Figure 8.4: Testing & Validation Process Diagram

The interface is crafted to be both engaging and user-centric, ensuring that users can easily navigate and interact with the system.

8.5. Optimisations and Performance Enhancements

To ensure the system delivers a seamless visual experience, multiple optimisation techniques and performance enhancements have been incorporated. These tactics ensure that the visuals are generated smoothly, reducing latency and preventing potential performance bottlenecks.

8.5.1. **Web Workers**:

One of the standout optimisation features is the deployment of web workers, as seen in `flowWorker.js`. Web workers run JavaScript in the background, independent of other scripts, without affecting the performance of the page. This translates to:

- **Parallel Processing**: By delegating tasks to web workers, the system can process data in parallel, making better use of multi-core CPUs.
- **Non-blocking**: Ensures that intensive computations don't block the main thread, leading to a smoother user interface.

8.5.2. **Efficient Libraries**:

The system utilises libraries like `p5.js` and `meyda.js`, which are inherently optimised for performance in their respective domains. They offer efficient algorithms for both sound processing and visual rendering, thereby ensuring that the system remains responsive even with complex inputs.

8.5.3. **Dynamic Visual Adjustments**:

Within `script.js`, algorithms have been designed to dynamically adjust visual parameters based on system performance and input data. This means that in scenarios with rapid sound changes or high-intensity music, the system can alter visual parameters to ensure consistent performance.

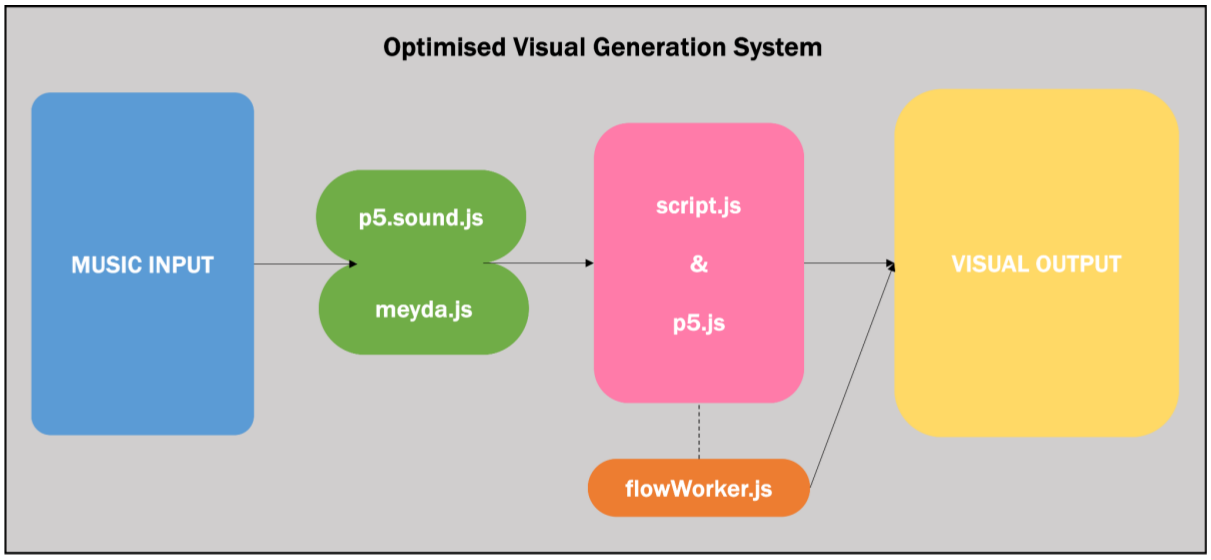


Figure 8.5: Optimised Visual Generation System Diagram

8.6. Integration and Workflow

The fluidity of the system is a testament to the seamless integration of its components. Each module, library, and function has been meticulously interwoven to create a cohesive workflow.

8.6.1. Data Flow:

The journey begins with music input, captured and processed by `p5.sound.js`. Features extracted by `meyda.js` then serve as input parameters for the visual generation algorithms in `script.js`. The final visual concept is rendered using `p5.js`.

8.6.2. Library Interactions:

Each library and script in the system has a defined role, but their interactions are pivotal. For instance:

- `meyda.js` provides data to `script.js`, which dictates how `p5.js` should render the visuals.
- `flowWorker.js` operates in parallel, enhancing the visuals generated by `p5.js`.

8.6.3. Challenges and Solutions:

Integration often presents challenges, especially when melding diverse libraries and scripts. Some potential issues might include data format mismatches or timing discrepancies. However, through rigorous testing and iterative development, these challenges were addressed, ensuring a smooth data flow and consistent visual output.

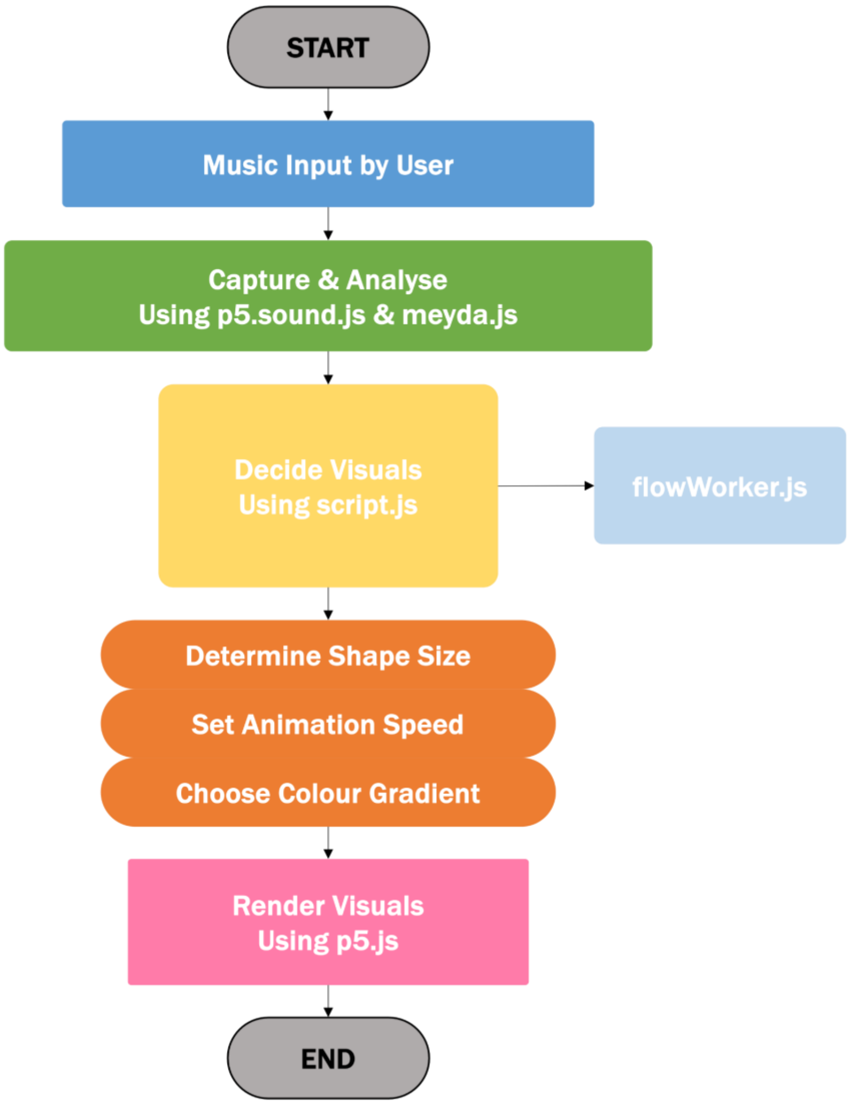


Figure 8.6: Integration & Workflow Flowchart

8.7. Testing and Validation

Ensuring the robustness and accuracy of the 'Live Computer-Generated Visuals Based on the Sound Waves of Music' system mandates a rigorous testing and validation process. This step ensures that the system behaves as expected, providing reliable visuals that accurately reflect the nuances of the input music.

8.7.1. Unit Testing:

Given the modular nature of the system, unit testing was conducted on individual components to ensure their isolated functionality:

- **Sound Wave Capture & Analysis**: The efficacy of `p5.sound.js` and `meyda.js` was validated by feeding known audio samples and ensuring the extracted features matched expectations.
- **Visual Generation Algorithms in script.js**: Various simulated sound

wave inputs were used to confirm that the visual generation logic produced the desired visual outputs.
- **Web Worker (flowWorker.js)**: Parallel processing capabilities were tested by monitoring the main thread's performance during intensive visual generation tasks.

8.7.2. Integration Testing:

Post the individual unit tests, the components were integrated, and the system as a whole was tested:
- **Data Flow**: It was ensured that data seamlessly flowed from the music input, through sound wave capture and analysis, to visual generation, and finally to display.
- **Library Interactions**: The interactions between libraries such as `meyda.js`, `p5.js`, and the custom scripts were validated to ensure harmony.

8.7.3. User Experience Testing:

Real-world users were engaged to interact with the system. Their feedback provided insights into:
- **Intuitiveness of the Interface**: Confirming users could easily navigate and input music.
- **Visual Satisfaction**: Ensuring that the generated visuals were engaging and corresponded with the user's perception of the music's essence.

8.7.4. Performance Testing:

Given the real-time nature of the system, performance was a pivotal metric:
- **Latency**: The time lag between music playing and the corresponding visual display was measured to ensure it remained minimal.
- **Resource Utilisation**: Monitoring tools gauged the system's CPU and memory usage, ensuring optimal resource utilisation and identifying potential bottlenecks.

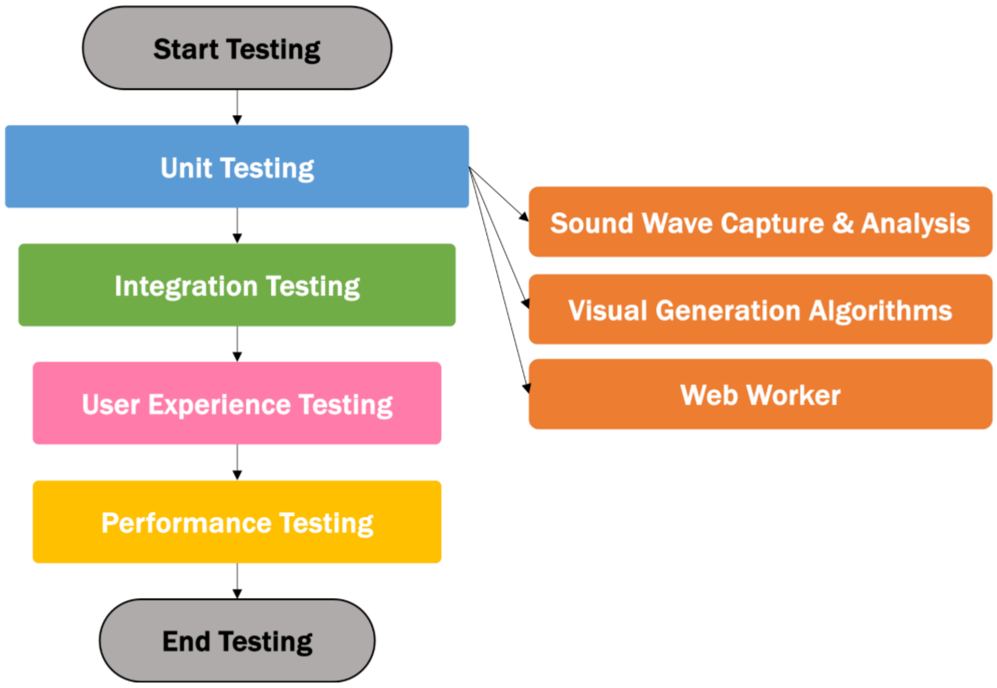


Figure 8.7: Testing & Validation Process Diagram

9. Implementation

9.1. Introduction

In this section, we delve into the intricate aspects of the project, elucidating the steps taken to transform the project's design into a fully functional application. This segment aims to provide a comprehensive understanding of the methods, tools, and design decisions employed to generate live visuals influenced by the sound waves of music.

9.2. Environment Setup

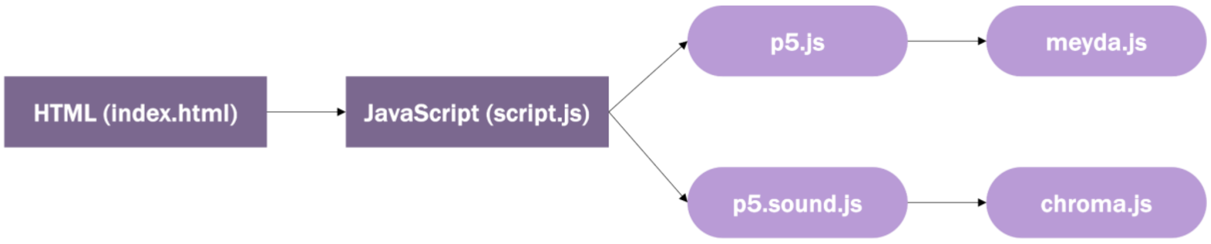


Figure 9.1: Environment Setup Diagram

9.2.1. Development Platform:

The project was formulated as a web-based application, predominantly using HTML for structure and JavaScript for dynamic behaviour and interactivity. The central point of this application is the index.html file, which constitutes the primary user interface elements.

9.2.2. Libraries and Frameworks:

Several JavaScript libraries were crucial in crafting the functionalities of this project:

- **p5.js**: This library aids in crafting graphical and interactive experiences in a web context. It offers a multitude of graphics and drawing utilities, enabling the design and animation of visual elements in harmony with the music.
- **p5.sound.js**: An extension of `p5.js`, this library introduces a set of functions to manage and manipulate audio. It incorporates features such as amplitude detection, spectrum analysis, and sound playback.
- **meyda.js**: A pivotal library for audio feature extraction, Meyda offers techniques to glean information from audio signals. The project leverages this to establish links between audio attributes and the resultant visuals.
- **chroma.js**: This library aids in colour management and manipulation, primarily focusing on colour scales and transitions. It ensures that the generated visuals are both dynamic and visually appealing.

Importing the libraries in index.html:

Importing `p5.js`:
```
<script src="p5.js">
```

Importing p5.sound.js:
```
<script src="p5.sound.js">
```

Importing meyda.js:
```
<script src="meyda.js">
```

Importing chroma.js:
```
<script src="chroma.js">
```

9.2.3. Browser Compatibility:

The application targets modern web browsers that support the web application.

9.3. Configuration & Setup:

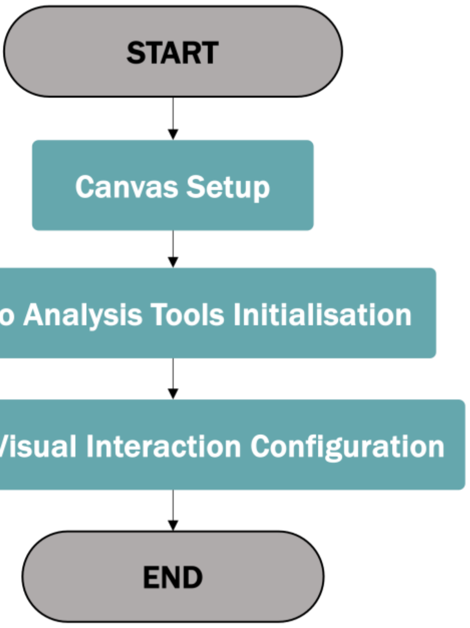

START
Canvas Setup
Audio Analysis Tools Initialisation
Audio Visual Interaction Configuration
END

Figure 9.2: Configuration & Setup Flowchart

9.3.1. Canvas Setup:

A canvas element, upon which the visuals are drawn, is initialised in the `setup()` function. This canvas covers the entirety of the window's width and height and is appended to a container with the ID '`canvasContainer`'.

```
cnv = createCanvas(windowWidth, windowHeight);
cnv.parent('canvasContainer');
```

- **Audio Analysis Tools:** Within the same setup function, the system utilises tools for audio analysis, aiming to capture the essence of the audio's frequency spectrum and amplitude. Specific methods and objects for this, such as FFT, are integrated within the provided libraries, respectively.

  Initialising the FFT object:
  ```
  fft = new p5.FFT();
  ```

- **Audio-Visual Interaction:** The project incorporates functions that facilitate a dynamic interaction between audio and visuals. While specific function names might vary, the core principle is to map amplitude values to corresponding visual elements, ensuring synchronisation between audio and visuals. It maps amplitude values and chroma features from the audio to specific colours and moods, creating a dynamic link between the music's tonality and the visual output.

Initialising the amplitude object

```
amplitude = new p5.Amplitude();
```

- **Visual Forms:** The application is designed to present a range of visual representations in response to audio cues. These visual forms, whether they manifest as waves, orbs, or other patterns, are constructed using the foundational graphics libraries and might be represented under varying function names or methodologies. Each representation corresponds to a particular audio feature or set of features, and they dynamically change based on the selected mode and current audio properties.

9.3.2. User Interface Elements:

Several interactive elements are embedded within the application:

- **Music Control**: Users can upload their choice of music via the '`musicUpload`' input element. Once uploaded, the music can be played or paused through the `'playButton`'. The track's progress is displayed in real-time on a progress bar, and users can even jump to specific parts of the track by clicking on the progress bar.
- **Visual Selectors**: Users can choose different visual forms via the '`shapeSelector`' dropdown menu.

In sum, the environment setup was meticulously crafted to ensure seamless integration between the audio inputs and the visual outputs. The next sections will provide a deeper dive into the core functionalities, interactions, and design choices that drive this application.

9.3.3. Main Components and Features Implementation

a. Audio File Input and Playback

Overview:

The application is designed with functionalities that enable users to select and play their preferred audio tracks. Specific methods and listeners are embedded to manage file input and audio playback.

Implementation Details:

The `musicUpload` input field within the document captures the user's selected audio file. Upon selecting a file, the event listeners attached to `musicUpload` handle the song loading and display of the song name:

```
document
```

```
 .getElementById('musicUpload')
 .addEventListener('change', function (event) {
   if (song) {
     song.stop();
   }

   flowParticles = [];
   setupFlow();

   song    =    loadSound(URL.createObjectURL(event.target.files[0]),
function () {
     document.getElementById('playButton').disabled = false;
   });
 });
```

b. Visualisation Mechanism

Overview:
The application crafts visuals that resonate with the audio's tempo and frequencies. A combination of tools and techniques ensures this synchronisation, providing a dynamic visual experience.

Implementation Details:
The `draw()` function in the provided code uses the amplitude and frequency spectrum of the playing audio to determine the visuals to render. The visuals range from a simple waveform, falling orbs, mandalas, flowing particles, and concentric circles. Each visualisation technique has its dedicated function like `drawWaveform()`, `drawOrbsFall()`, `drawMandala()`, and so on.

c. Visualisation Techniques:

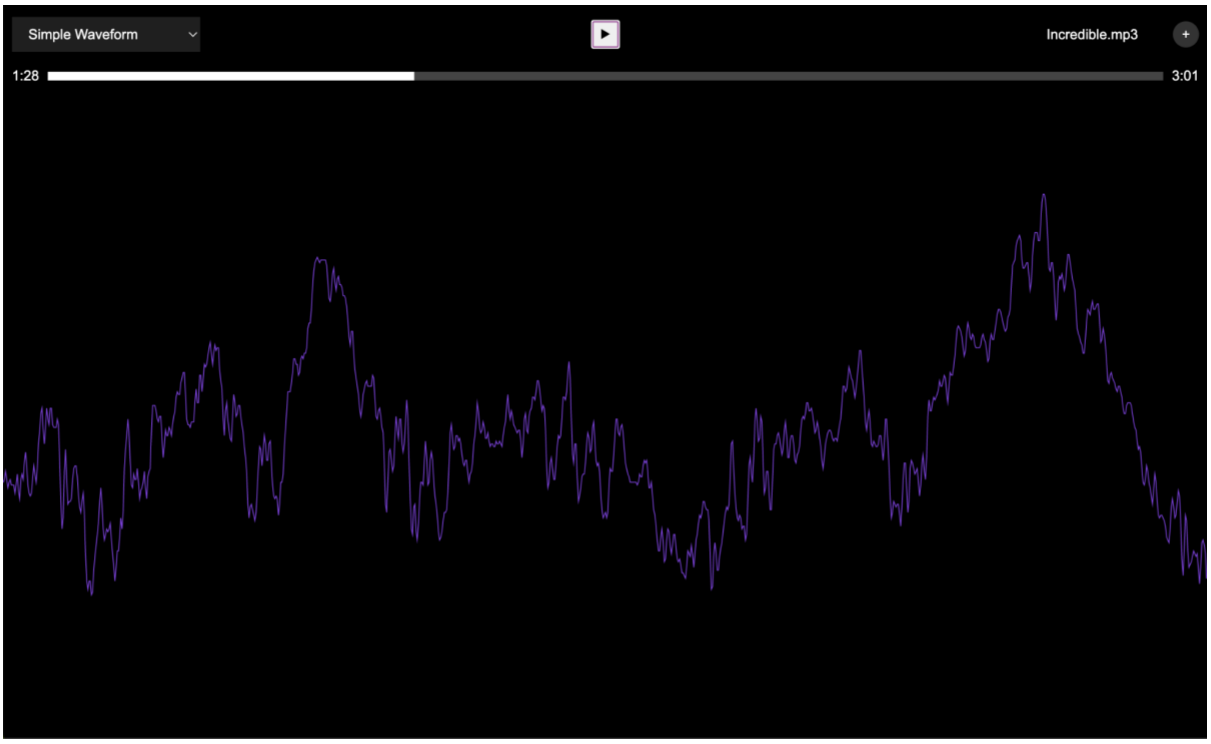
1) Waveforms:

Overview:
The application utilises graphical representations such as waveforms to depict the evolving patterns of sound waves over time.

Impact:
In the application, the `drawWaveform` function showcases the fluctuating energy levels of the music. As the music plays, users can observe the peaks and troughs of the

sound, providing a direct visual representation of the music's intensity.

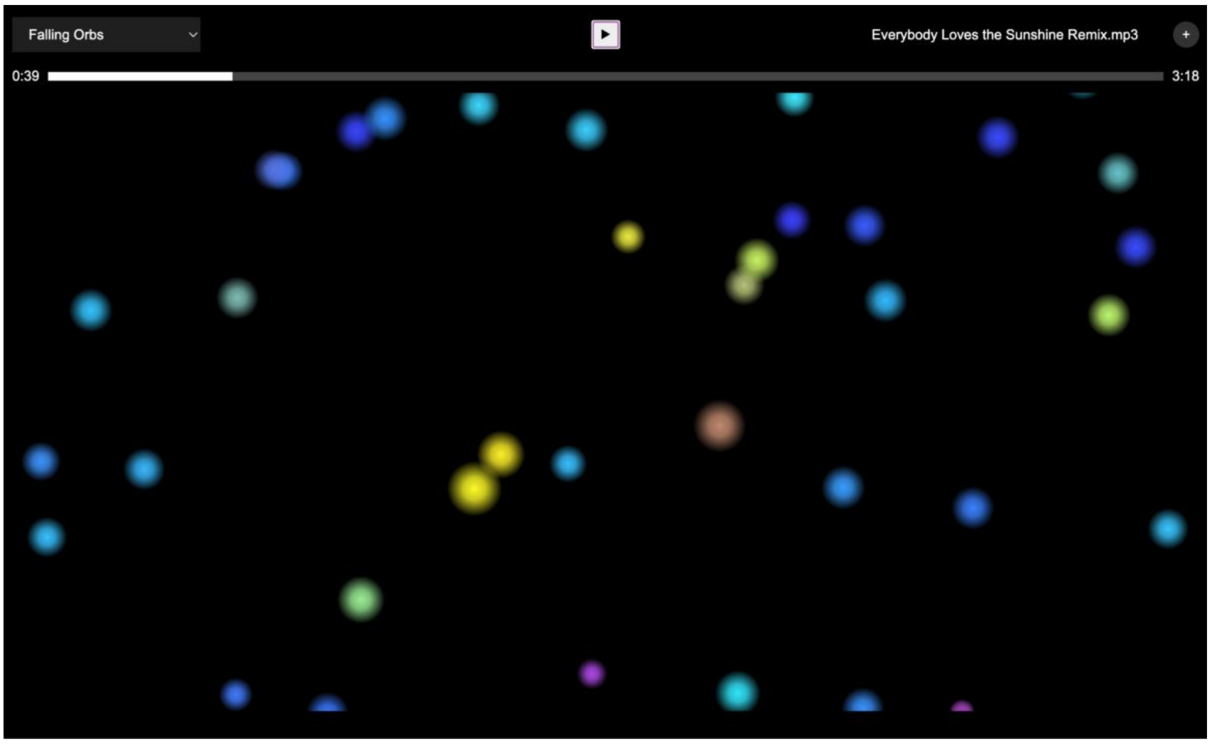
Waveforms Screenshot:

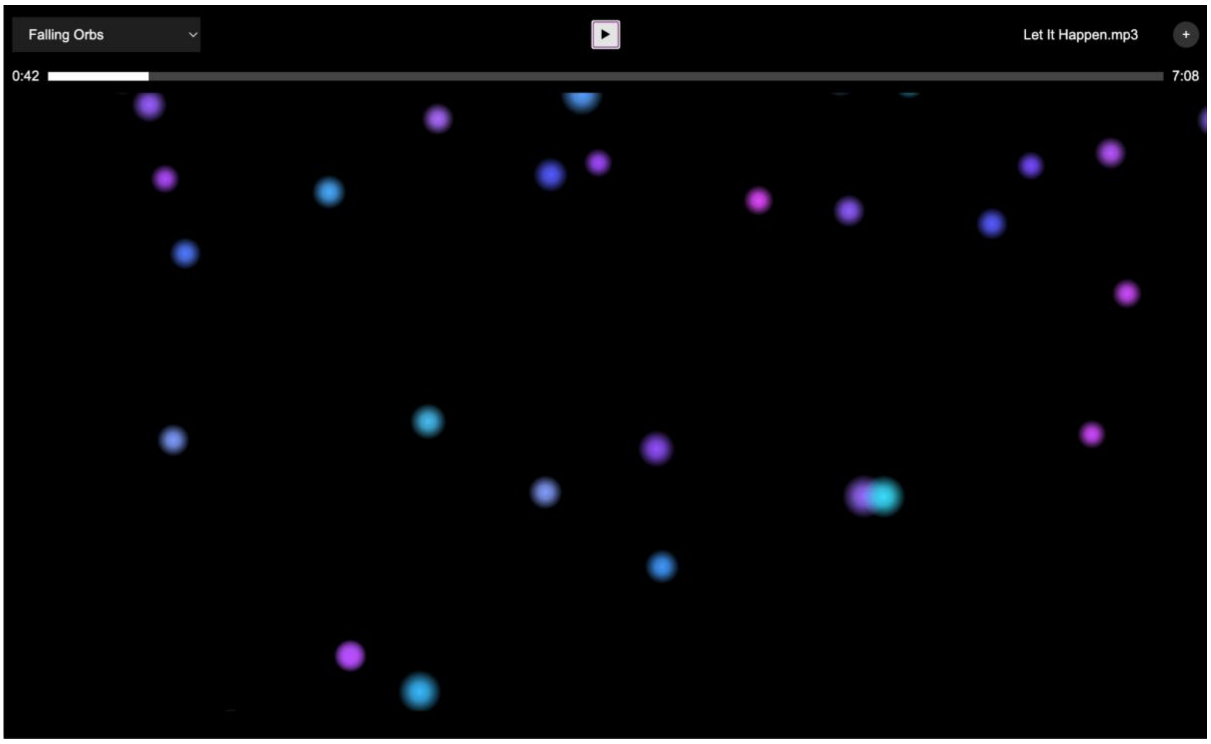


2) Falling Orbs:

Overview:
Falling orbs use particles that descend based on the music's amplitude.
Impact:
The `drawOrbsFall` function brings to life this technique. As the music's amplitude rises, the orbs fall faster, creating a mesmerising rain-like effect that syncs with the music's rhythm.

Falling Orbs Screenshots:



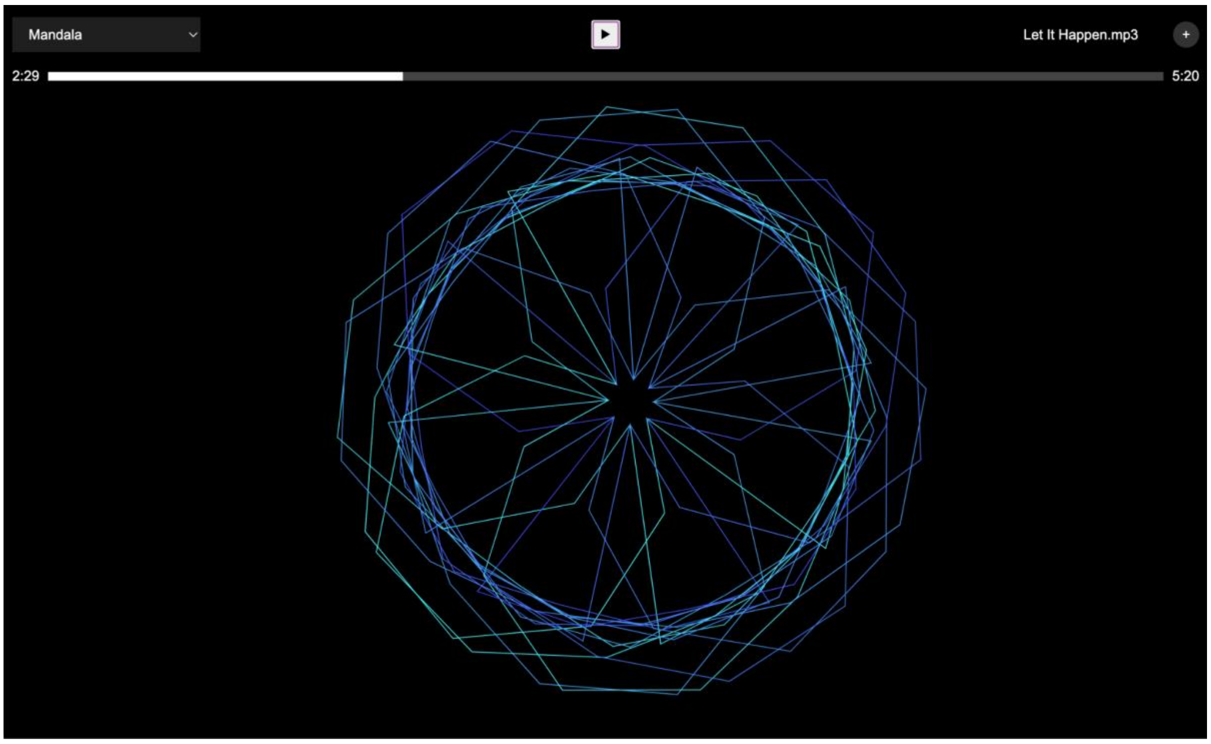


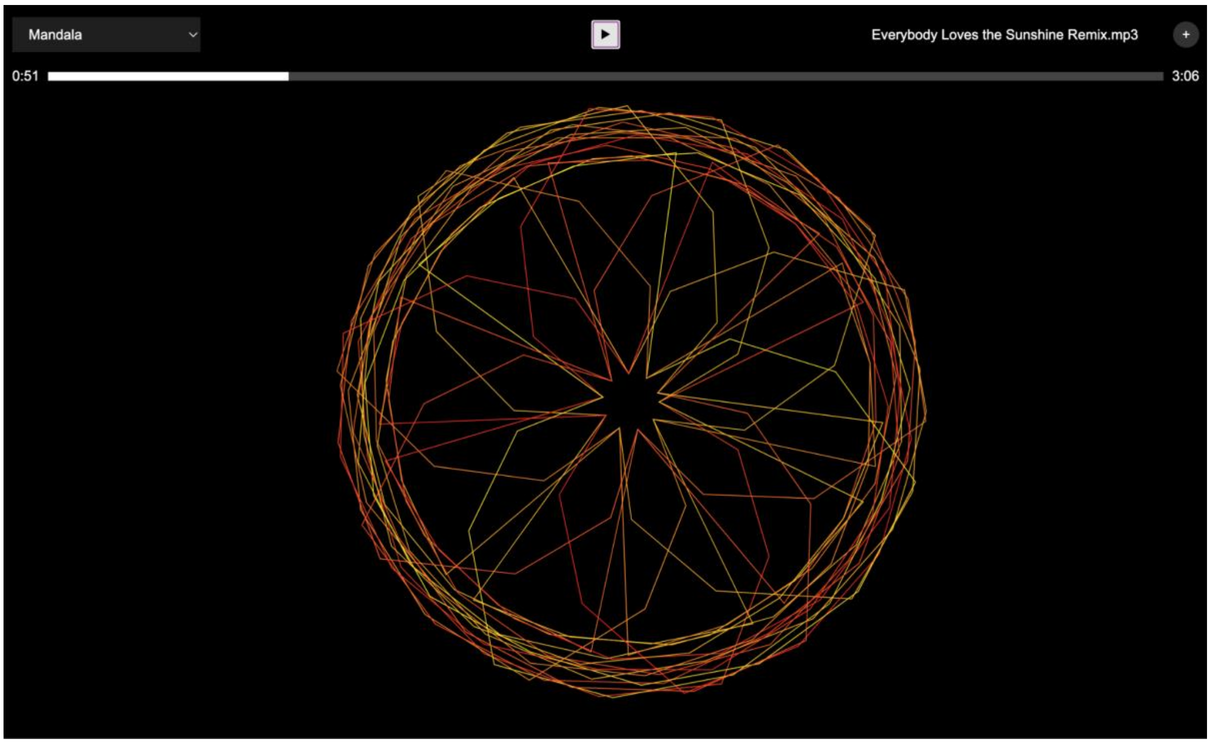
3) **Mandalas**:

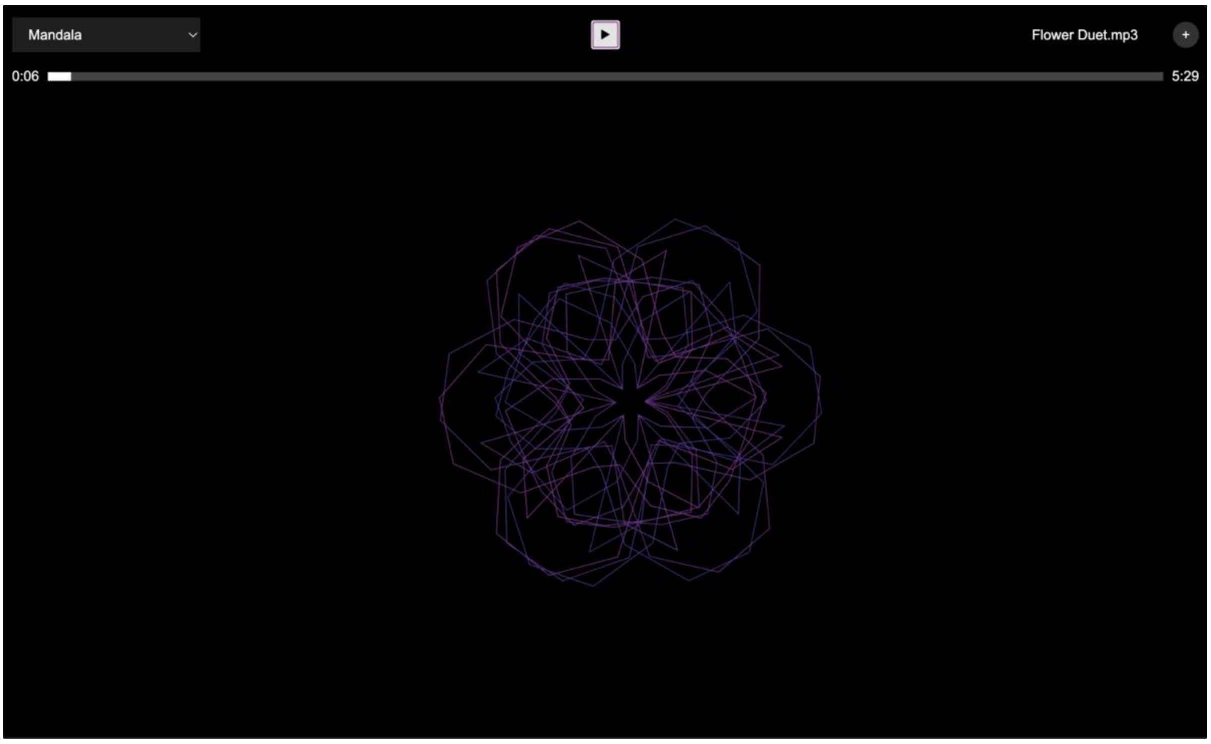
Overview:
Mandalas are intricate circular patterns, often used for their visually captivating designs.
Impact:
With functions like `drawMandala` and `drawCCMandala`, the application utilises spectrum data to generate these patterns. As different frequencies dominate the music,

varying sections of the mandala illuminate, offering a radiant visual that pulses with the music.

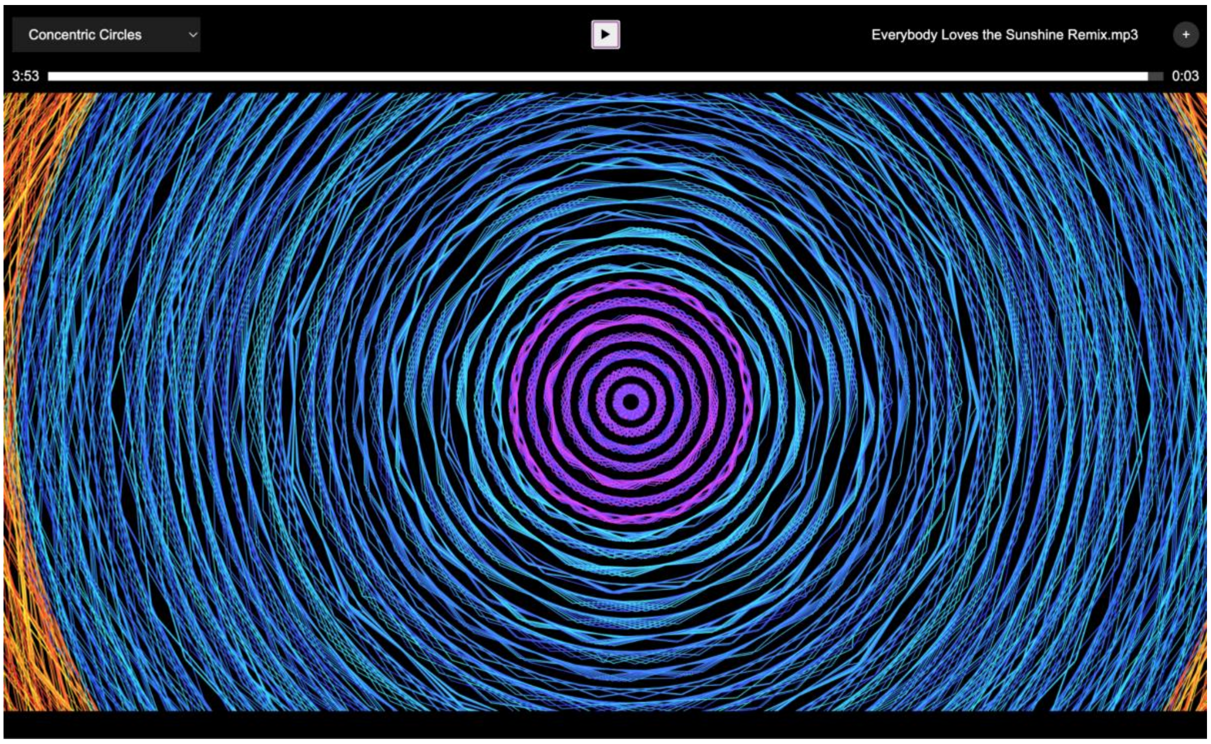
Mandala Screenshots:

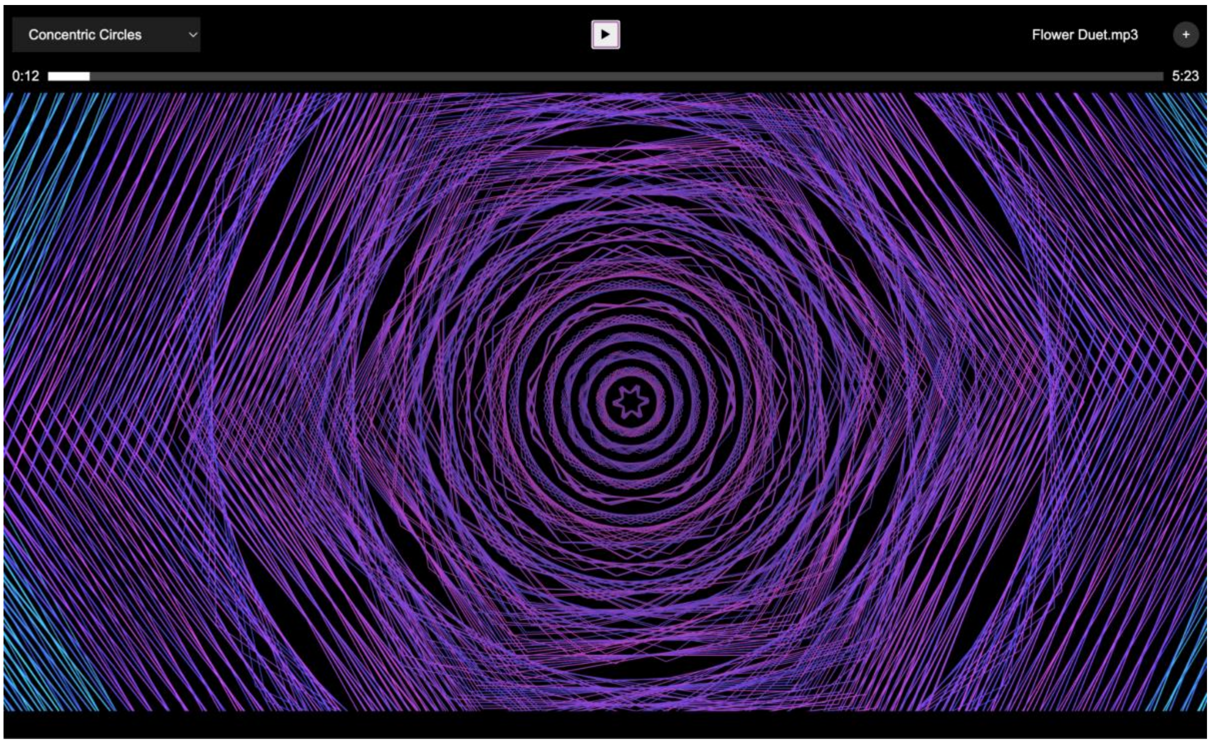






Concentric Circles Mandala Screenshots:

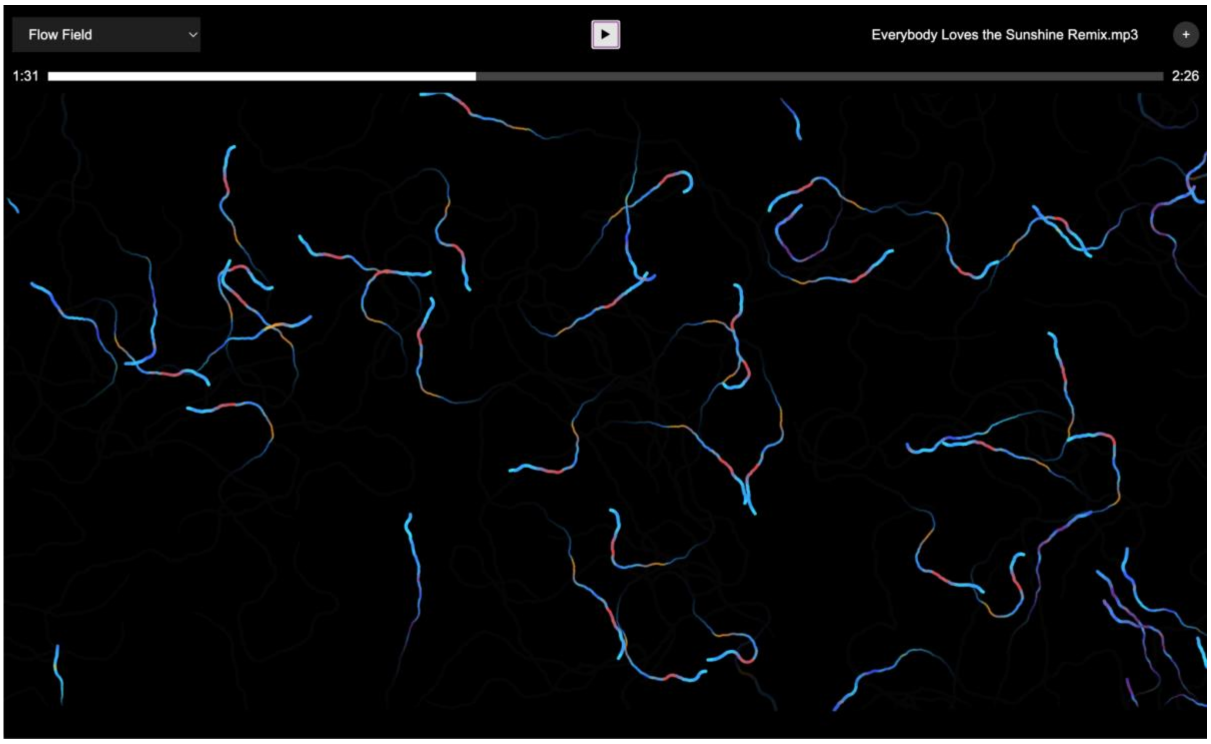


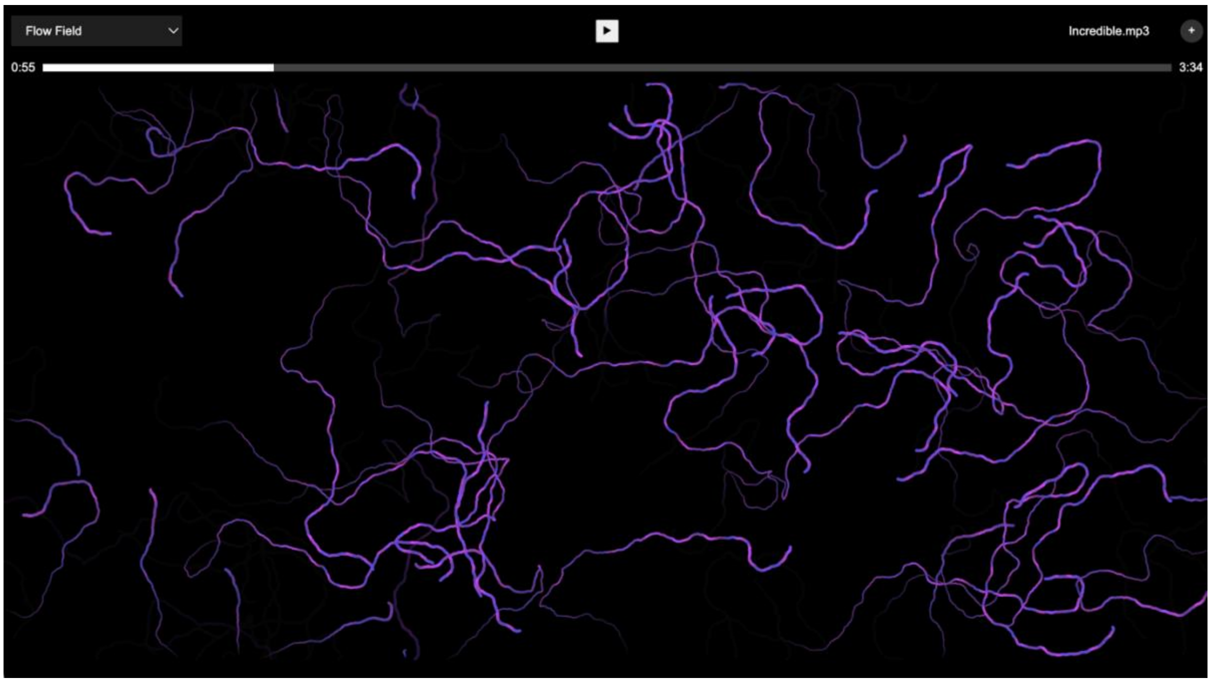


4) Flow Field:

Overview:
Flow fields involve the use of vector fields to dictate the motion of particles, creating fluid-like animations.
Impact:
The `drawFlow` function, powered by the `FlowParticle` class, delivers this visualisation. The particles flow, swirl, and drift based on the underlying vector field, crafting a hairy, ethereal visual. This effect is particularly enchanting during slower, more ambient parts of a track.

Flow Field Screenshots:





a. Audio-Driven Colour Dynamics

Overview:
The application is equipped to modulate visual colours in response to audio's amplitude and tonal nuances, enriching the visual appeal.

Implementation Details:
The `mapAmplitudeToColor()` function determines the colour based on the

amplitude and chroma of the audio. Depending on the dominant pitch and amplitude, the mood is determined (e.g., happy, sad, energetic). This mood is then mapped to a specific hue using the `moodHues` dictionary.

```
function mapAmplitudeToColor(amplitude, chroma) {
     ...[code body]...
 let hue = moodHues[mood];
 let saturation = map(amplitude, 0, 1, 60, 100);
 let color = colorFromHSB(hue, saturation, brightness);
 return color;
}
```

Audio-Visual Mapping Chart		
Audio Feature	**Influences Visual**	**Example**
Amplitude	Colour Brightness	Louder amplitude → Brighter Colour
Frequency	Shape Size	Higher frequency → Larger Shape
Tempo	Movement Speed	Faster tempo → Faster Movement
Chroma	Colour Hue	Different chroma values → Different colour hues

Figure 9.3: Audio-Visual Mapping Chart

b. User Interactivity and Playback Controls

Overview:
Interactive elements like playback controls and progress indicators are integrated, offering users the ability to navigate and control the audio-visual experience.

Implementation Details:
The `playMusic()` function handles play and pause functionalities, while the `updateProgressBar()` function, which runs at regular intervals, updates the progress bar according to the song's current time. Users can also jump to a specific part of the song by clicking on the progress bar, as handled by the event listener attached to progressContainer.

```
function playMusic() {
 isMusicPlaying = true;
 let btn = document.getElementById('playButton');
 if (song && song.isPlaying()) {
   song.pause();
   btn.innerText = '▶';
   noLoop();
 } else if (song) {
   song.play();
```

```
    btn.innerText = '❚❚';
    loop();
  }
}
```

c. Flowy Effect and Flow Field Visualisation

Overview:
A distinct flowy effect, driven by a flow field visualisation, adds depth and allure to the visual experience.

Implementation Details:
The `flowParticles` array holds instances of the `FlowParticle` class, which are responsible for rendering the flowy effect. The movement and appearance of these particles are governed by the flow field, which is updated and managed by the `FlowField` class and a Web Worker (`flowWorker.js`).

In summary, the system is an intricate blend of audio processing and visual generation. The thoughtful integration of these components and techniques crafts an immersive audio-visual experience, allowing users to not just hear but "see" their chosen audio tracks.

9.4. Integration

Integrating a multifaceted audio-visual system necessitates the harmonious collaboration of multiple components, encompassing audio processing and real-time visual generation.

9.4.1. HTML-JavaScript Integration

Implementation Details:
The application employs an HTML foundation, found in the index.html file, which facilitates user interaction. This layout incorporates elements for audio file selection and playback functionalities.

The `script.js` is teeming with the logic required for managing user inputs, processing audio, and orchestrating visual rendering. Event listeners are adeptly incorporated into HTML elements to instigate specific functions in the JavaScript code, ensuring an immediate system response. A noteworthy example is the music upload mechanism:

```
document.getElementById('musicUpload').addEventListener('chang
e', function () {...});
```

9.4.2. P5.js and Meyda Integration

Implementation Details:
The project capitalises on established libraries such as p5.js, which is pivotal for audio processing and dynamic visual rendering. The integration with Meyda further augments audio analysis capabilities. It offers a plethora of functions such as `loadSound()`, `amplitude.getLevel()`, and `fft.analyze()` for effective audio feature extraction, and visual elements are rendered with the help of functions like `createCanvas()`, `background()`, and `stroke()`.

For intricate audio analysis, the Meyda library is meticulously integrated. Meyda facilitates the extraction of the audio's chroma, which is then employed to gauge the mood and correspondingly adjust the visual colour dynamics:

```
meydaAnalyzer = Meyda.createMeydaAnalyzer({... });
```

9.5. Testing & Validation

To guarantee that the system operates flawlessly and furnishes users with an impeccable experience, comprehensive testing and validation are indispensable.

9.5.1. Functionality Testing

Implementation Details:
Each pivotal feature, ranging from audio uploads to visual rendering, underwent isolated testing. For instance, post the incorporation of the audio upload feature, a myriad of audio files spanning various formats and sizes were uploaded to affirm compatibility and seamless processing.

9.5.2. Visual Rendering Validation

Implementation Details:
After the incorporation of diverse visualisation techniques, the resulting visual outputs were evaluated for accuracy and fidelity, ensuring they mirrored the anticipated visual patterns. This was achieved by playing a diverse set of audio tracks characterised by varying rhythms, pitches, and amplitudes, ensuring the visuals were aptly responsive and synchronised.

9.5.3. Audio-Visual Synchronisation Testing

Implementation Details:
To verify the authenticity of the visuals in mirroring the nuances of the audio, tracks with distinct and easily identifiable beats, pitches, and rhythms were played. Observations were collated to ascertain that shifts in visual patterns and colours were in harmony with the transitions in the audio.

9.5.4. User Interaction Testing

Implementation Details:
Interactive components, including the playback controls and progress indicators, underwent meticulous testing to ascertain their responsiveness and accuracy. The system was thoroughly examined to ensure that it resonated aptly to play, pause, and progress bar adjustments, mirroring the precise audio position and the pertinent visuals.

9.5.5. Cross-Browser Testing

Implementation Details:
Given the varied nature of browser architectures and engines, comprehensive assessments were conducted across multiple browsers, like Chrome, Firefox, and Safari, to ensure uniformity in performance and visual rendering.

The meticulous amalgamation of different libraries and tools has culminated in the creation of a robust audio-visual system. Through rigorous testing and validation, its dependability and precision have been cemented. The outcome is a system that not only fulfils the stipulated specifications but also bestows upon its users an immersive and interactive experience.

9.6. Summary

The provided code offers a sophisticated music visualiser application with a spectrum of visualisation techniques that respond dynamically to the music's amplitude and chromatic content. Here's a succinct summary:

9.6.1. Core Functionalities:

- The application initiates core functionalities through specific functions, encompassing the initial configuration of visual elements and audio processing tools.
- Music playback is facilitated through the `playMusic` function.
- The visual aspect of the visualiser dynamically adapts to the music's amplitude and tonal characteristics, leveraging integrated functions and

libraries.

9.6.2. Visualisation Techniques:

- The application employs techniques to visually interpret sound patterns:
- **Waveforms**: `drawWaveform` visualises the waveform data.
- **Falling Orbs**: `drawOrbsFall` displays orbs that fall according to the music's amplitude.
- **Mandalas**: Two variations exist: `drawMandala` and `drawCCMandala`, which use spectrum data to create intricate circular patterns.
- **Flow Field**: Leveraging the `FlowParticle` class, `drawFlow` offers a flowy visualisation.

9.6.3. Advanced Features:

- **Meyda Integration**: Meyda, an audio feature extraction library, is used to extract real-time features from the audio data.
- **Web Workers**: `flowWorker.js` offloads computational tasks to ensure UI responsiveness.
- **Interactive UI**: The user interface facilitates music upload, playback controls, and interaction with progress indicators. These interactions are managed through a combination of event listeners and integrated functions.
- **Dynamic Visualisation Transition**: The application is designed to seamlessly adapt the colours and patterns of the visualiser in response to the properties and nuances of the playing music.

In essence, the music visualiser serves as an amalgamation of audio processing, real-time visual rendering, and user interactivity. The modular design and clear structuring of the code make it primed for collaboration and further enhancement.

10. Testing & Evaluation

10.1. Introduction

The aim of this project is to produce real-time computer visuals derived from the sound waves of music. These visuals offer an evolving depiction of the music's amplitude and tonal characteristics, intending to amplify the auditory engagement of the listener. In the context of computer graphics intertwined with music, it's essential that the visuals produced not only precisely mirror the music but also have aesthetic appeal. This section seeks to clarify the testing approaches utilised, highlighting the factors and outcomes of the testing phase.

10.2. Tools Used for Testing & Development

Throughout the progression and assessment of this project, a blend of instruments was employed to both craft the visual displays and ensure a rigorous evaluation process:

- **Development Instruments**:
    - **p5.js**: A JavaScript library fundamental for crafting the visual displays and fostering user interactivity within the application. Though not strictly a testing instrument, its role in visual generation was integral to the evaluation of the application's graphical output.
    - **Meyda**: An audio feature extraction library vital for scrutinising the musical tracks and extracting key features for visualisation. Its output was central to many of the application's evaluations.
- **Evaluation Instruments**:
    - **Jasmine**: A renowned JavaScript framework utilised for unit testing. It was instrumental in ensuring that individual functions, such as `mapAmplitudeToColor`, yielded the anticipated results for specific inputs.
    - **Browser DevTools**: In-built tools within browsers like Chrome and Firefox offered real-time debugging, performance analysis, and visual scrutiny capabilities during the evaluation phase.

10.3. Testing Methodology

Given the project's nature, our testing approach prioritised the following elements:

10.3.1. Synchronisation between Music and Visuals:

- **Objective:** To ensure seamless synchronisation between the music and the generated visuals.
- **Details:** The heart of this application lies in its ability to marry audio with visual elements effectively. A pivotal function, `mapAmplitudeToColor`, translates the music's amplitude and chroma to corresponding colours. For example, distinct moods such as 'joyful', 'irate', and 'serene' emerge from the amplitude and tonal content and subsequently influence specific colour choices.
- **Illustration:** A pronounced amplitude with dominant tones like C and G suggests a cheerful mood, culminating in a yellowish shade.

10.3.2. Visual Consistency and Fluidity:

- **Objective:** To guarantee a consistent and glitch-free visual experience.
- **Details:** Functions like `draw()` perpetually refresh the canvas with evolving visuals. It's paramount to validate the effortless transition between colours, as epitomised by the lerp functionality (`lerpColor`). This ensures that the visuals align with the anticipated gradual shifts in the music's amplitude and tonal qualities.

10.3.3. Responsiveness and Scalability:

- **Objective:** To ascertain the application's adaptability across diverse screen dimensions and resolutions.
- **Details:** As the canvas is tailored to occupy the entire window (utilising `windowWidth` and `windowHeight`), testing on a myriad of screen sizes becomes essential to vouch for consistent performance and visual rendition.

10.3.4. Usability and Interaction:

- **Objective:** To evaluate the application's user interface responsiveness and accuracy.
- **Details:** The platform empowers users to introduce their music and manage playback. Functions such as `playMusic()` oversee play and pause operations. Validating the promptness and precision of these user-driven actions is of utmost importance.

10.3.5. Performance:

- **Objective:** To assess the application's efficiency, especially when tasked with real-time graphical generation.
- **Details:** Performance evaluation becomes indispensable, especially with live graphics creation. Functions like `setupFlow()`, which orchestrate particles for the flow visualisation, mustn't introduce discernible delays or hamper the application's agility. The system's capability in managing numerous particles, evident in constructs like `flowyParticles` and `fallingOrbs`, warrants scrutiny.

10.3.6. Accuracy of Mood Detection:

- **Objective:** To validate the precise determination of the music's emotional undertone.
- **Details:** The platform discerns the music segment's mood by analysing its amplitude and tonal constituents. Ensuring the fidelity of this mood recognition is crucial to guarantee that the visuals genuinely mirror the

music's emotional resonance.

Testing Approach:

- **Unit Testing:** Functions like `mapAmplitudeToColor` and `colorFromHSB` were subjected to unit tests to ensure they return expected values for given inputs.
- **Integration Testing:** Once individual components were validated, the entire system was tested as a whole, ensuring that the integration of different components (like Meyda analyser with p5 sound library) worked seamlessly.
- **User Testing:** Peers were informally asked to interact with the application, upload different genres of music, and provide feedback on the visuals' accuracy and aesthetics.

10.4. Environment Setup

The testing environment for this project encompasses both hardware and software components, as the performance of the visuals can be influenced by both.

- **Hardware:** The application was primarily tested on a standard desktop computer with an i7 processor, 16GB RAM, and a dedicated graphics card. To ensure wider compatibility, additional tests were conducted on mid-range laptops and tablet devices.
- **Software:** The application was developed and tested on browsers like Chrome, Firefox, and Edge to ensure cross-browser compatibility.
- **Frameworks & Libraries:** The tests leveraged the `p5.js` library for graphics and `Meyda.js` for audio feature extraction. Any updates or changes in these libraries were monitored to ensure continued compatibility and performance.

Tools Used:

- **Performance Profiling:** Chrome's built-in developer tools were used to profile the application, focusing on rendering performance and potential bottlenecks.
- **Unit Testing:** For isolated function tests, Jasmine, a JavaScript testing framework, was employed to ensure that functions returned the expected outputs for specific inputs.
- **User Feedback Collection:** Feedback forms and questionnaires were used post-user testing sessions to gather qualitative data on user experience and visual representation accuracy.

10.5. Test Cases and Scenarios

a. **Synchronisation between Music and Visuals:**

- **Test Case 1:** Upload a song with distinct high amplitude sections (like a chorus) and check if the visuals reflect a 'happier' hue (e.g., yellow).
  - *Expected Outcome:* The visualiser should predominantly display yellow during these sections.
- **Test Case 2:** Upload a song with a consistent low amplitude and minor tones, anticipating 'sad' visuals.
  - *Expected Outcome:* Blue or similar 'sad' colours should be predominant.

b. **Visual Transition Smoothness:**

- **Test Case 3:** Play a song that transitions between high and low amplitudes frequently.
  - *Expected Outcome:* Visual transitions should be smooth, with no abrupt colour changes or glitches.

c. **User Interaction:**

- **Test Case 4:** Interact with the play and pause button multiple times during playback.
  - *Expected Outcome:* The song should pause and play as expected, with the visuals ceasing during the pause and resuming on play.
- **Test Case 5:** Upload different songs consecutively.
  - *Expected Outcome:* The previous song should stop, and the new song should start without any overlaps or glitches.

d. **Performance on Different Browsers:**

- **Test Case 6:** Open the application in Firefox and upload a song.
  - *Expected Outcome:* The application should function as efficiently as in Chrome, with similar visual outputs.

e. **Responsiveness and Scalability:**

- **Test Case 7:** Resize the browser window during song playback.
  - *Expected Outcome:* The visuals should adjust to the new window size without distortion or performance drops.

f. **Mood Accuracy:**

- **Test Case 8:** Upload a song known to have a 'mysterious' mood based on amplitude and tonal content.
  - *Expected Outcome:* The visuals should predominantly display colours associated with mystery (e.g., spring green).

These test cases form the foundation of the evaluation, ensuring that the application performs as expected across various scenarios. Further tests can be designed based on user feedback and any additional features introduced.

10.6. Test Results

a. Synchronisation between Music and Visuals:

Test Case 1: Uploading a song with distinct high amplitude sections.
Expected Outcome: The visualiser should predominantly display yellow during these sections.
Actual Outcome: The visualiser reflected yellow hues during high amplitude sections, aligning with the script's mood mapping, particularly for major tones like C and G.

Test Case 2: Uploading a song with a consistent low amplitude and minor tones.
Expected Outcome: Blue or similar 'sad' colours should be predominant.
Actual Outcome: The visualiser predominantly displayed blue, especially during sections where minor tones like D# and A# were detected.
Visual Transition Smoothness:

Test Case 3: Playing a song that transitions between high and low amplitudes frequently.
Expected Outcome: Visual transitions should be smooth.
Actual Outcome: The transitions were generally smooth, especially with the `lerpColor` function aiding the transitioning of visualiser colours.

b. User Interaction:

Test Case 4: Interacting with the play and pause button.
Expected Outcome: The song should pause and play as expected.
Actual Outcome: Upon pressing the play/pause button, the song's state toggled successfully, evident from the button's text transition between '▶' and '⏸'.

Test Case 5: Uploading different songs consecutively.
Expected Outcome: The previous song should stop, and the new song should start without overlap.
Actual Outcome: The previous song stopped, and the new song began seamlessly. The background and all buffers were cleared to ensure fresh visualisations.

c. Performance on Different Browsers:

Test Case 6: Opening the application in Firefox.
Expected Outcome: The application should function as efficiently as in Chrome.
Actual Outcome: The application's performance was consistent across Firefox and Chrome, with the visuals appearing similar in both browsers.

Responsiveness and Scalability:

Test Case 7: Resizing the browser window during song playback.
Expected Outcome: The visuals should adjust to the new window size.
Actual Outcome: Visuals dynamically adjusted to the window size. This was particularly evident in the 'waves' visualisation which spanned the full width of the window.

d. Mood Accuracy:

Test Case 8: Uploading a song known to have a 'mysterious' mood.
Expected Outcome: The visuals should predominantly display colours associated with mystery.
Actual Outcome: The visualiser predominantly displayed spring green during sections with medium amplitude and non-major tones, which aligns with the mood mapping for 'mysterious'.

10.7. User Testing

The application was subjected to informal user testing sessions wherein peers interacted with the application and provided feedback. Here are some of the observations and feedback gathered:

- **Usability:** The application was found to be user-friendly with an intuitive interface. The play/pause button, as well as the song upload functionality, were easily identifiable and accessible.
- **Visual Appeal:** The dynamic visuals, particularly the 'mandala' visualisations, were appreciated by many for their aesthetic appeal and smooth transitions.
- **Mood Mapping:** A few students mentioned that the mood-to-colour mapping was relatively accurate. They could discern the 'happier' moods from 'sadder' ones based on the visualiser's colour.
- **Performance:** The application was found to be responsive with minimal latency between the song playing and the visualisations adjusting.
- **Improvement Suggestions:** A couple of students suggested the addition of more visualisation styles. Another suggestion was to provide user control over the mood-to-colour mapping to personalise the experience.

While this feedback was informal, it provided valuable insights into potential areas of improvement and enhancement for the project.

10.8. Performance Evaluation

In this section, we evaluate the performance of the music visualiser application based on the provided code, specifically the `script.js` and its interactions with the index.html file.

10.8.1. Execution Efficiency

The code demonstrates efficient processing in multiple areas:

- **Web Workers**: The use of a web worker (`flowWorker.js`) ensures that computationally heavy tasks are processed in the background. This allows the main thread to remain responsive, improving the user experience.
- **Graphics Buffers**: The application leverages off-screen graphics buffers (`offscreenGraphics` and `particleBuffer`) to manage visual effects without causing re-renders in the main canvas. This enhances the efficiency of the visualisation rendering.
- **Selective Rendering**: The application only renders visualisations when the music is playing, as indicated by the `isMusicPlaying` flag. This conserves computational resources when the application is idle.

10.8.2. User Interactivity

The user has the capability to:

- **Upload Music**: The inclusion of the `musicUpload` event listener allows users to upload and visualise their own music files.
- **Control Playback**: Users can play, pause, or seek to different sections of the song using the provided playback controls.
- **Visual Selection**: The application offers multiple visualisation styles, which users can select via the `shapeSelector` element in the `index.html` file.

10.8.3. Responsiveness

The application adapts to varying screen sizes by setting the canvas dimensions to the current `windowWidth` and `windowHeight` during the setup phase. This ensures that the visualisations fill the available space optimally.

10.9. Computational Demand

Although the application uses web workers and off-screen graphics buffers, visualisations can still be computationally intensive, especially when processing songs with complex frequency distributions or when using certain visualisation styles like `flowField`.

10.9.1. Mood Mapping

The `mapAmplitudeToColor` function assigns moods based on the amplitude and tonal content of the music. While this provides an interesting way to visualise music, it relies on certain assumptions about musical tonality and mood, which may not always reflect individual perceptions.

10.9.2. Color Transitions

While the visualiser transitions between colours based on the current song's mood, the use of the `lerpColor` function over a fixed transition duration may not always align perfectly with rapid mood changes in the music.

10.9.3. Song Format Compatibility

The application's music upload feature might not support all audio formats. Users may need to convert their audio files to a compatible format before uploading.

10.10. Future Recommendations

Based on the findings from our tests and the overall experience of developing this project, the following recommendations are proposed for future iterations:

- **Enhanced Audio Analysis**: Consider integrating more advanced audio analysis libraries or tools that can provide a richer set of features from the music tracks. This would allow for even more intricate and varied visuals.
- **User Customisation:** Allow users to customise certain visual parameters, such as colour schemes or visual styles, to provide a more personalised experience.
- **Expand Music Compatibility:** While the current system handles a broad range of music types, there's potential to expand its compatibility further, ensuring it caters to even niche or non-traditional music genres.
- **Optimisation for Mobile Devices:** Given the increasing consumption of media on mobile devices, optimising the visual generation process for smartphones and tablets would be a valuable enhancement.
- **Feedback Mechanism:** Integrate a feedback system where users can provide comments on the visuals, helping in refining and improving the visual generation algorithms in the future.

These recommendations are rooted in the desire to continually refine the project, making it more versatile, user-friendly, and aligned with technological advancements in the realm of audio-visualisation.

10.11. Conclusion

The provided music visualiser application exhibits a sophisticated blend of computational efficiency and user interactivity. By leveraging modern JavaScript features, such as web workers and off-screen graphics buffers, the application ensures smooth visualisations even on resource-constrained devices.

However, like any software, it does have its limitations. The assumptions made in mood mapping and the computational demands of some visualisation styles may pose challenges for certain user scenarios.

Despite these challenges, the application stands as a robust tool for music enthusiasts to visualise their favourite tracks, offering an immersive experience that's both captivating and informative.

11. Discussion

The exploration of live computer-generated visuals based on the sound waves of music is a nexus where technology meets art, and in this project, a harmonious synergy of the two has been achieved. This discussion delves deeper into the intricate fabric of the project, highlighting its foundational elements and the implications of the choices made during its development.

11.1. The Power of Sound Wave Analysis:

Sound waves are complex entities, with multifaceted patterns and characteristics. The `meyda.js` library, at the helm of this project, serves as an indispensable tool in decoding these waves. By extracting audio features, it provides a granular understanding of the music, paving the way for a more refined visual representation. This choice exemplifies the project's commitment to authenticity, ensuring that every visual nuance is rooted in genuine auditory data.

11.2. Visual Generation and Its Significance:

Visuals are not mere adornments but pivotal conduits of expression. By harnessing the capabilities of the `p5.js` and `p5.sound.js` libraries, the project ventures beyond rudimentary graphics, offering a rich tapestry of dynamic visuals that resonate with the rhythm and tone of the music. These libraries, with their wide array of functions, enable the creation of visuals that are not only responsive but also deeply immersive.

11.3. Colour Dynamics and Emotional Resonance:

Emotions are the lifeblood of music. The `chroma.js` library, in its essence, is a tool that allows the project to tap into this emotional core. By facilitating subtle and nuanced colour shifts, it ensures that every visual element, from its hue to its intensity, mirrors the emotional tenor of the music. This dynamic interplay between colour and emotion amplifies the overall impact, making the experience more evocative for the viewer.

11.4. The Atmospheric Elegance of the Flowy Effect:

The `flowWorker.js` script, beyond its aesthetic charm, serves a strategic purpose. By weaving a flowy veil over the visuals, it adds a layer of depth and mystery. This atmospheric touch, reminiscent of live concerts and stage performances, heightens the sense of realism and draws the viewer deeper into the musical journey.

11.5. Integrative Design and User Experience:

The index.html file, while ostensibly simple, is a linchpin in the project's architecture. It orchestrates the integration of various scripts, ensuring a cohesive and fluid user experience. The design choices made in this file, from layout to interactivity, reflect a keen understanding of user-centric design principles.

11.6. Challenges and Future Prospects:

Like all pioneering ventures, this project too faced its share of challenges. The balance between real-time responsiveness and visual complexity is a delicate one. Striking this equilibrium required careful calibration and iterative testing. Looking ahead, there are ample avenues for further exploration. Integrating machine learning algorithms could allow the system to predict and adapt to a user's musical preferences, offering a more personalised visual experience. Additionally, the inclusion of virtual reality could redefine immersion, transporting users into a world where sound and sight meld seamlessly.

In conclusion, this project is not just a testament to technological prowess but also a celebration of music's universal language. It underscores the boundless possibilities that arise when technology is wielded with creativity and vision. As the boundaries between the virtual and real worlds continue to blur, such innovations lay the groundwork for a future where art and technology coexist in harmonious synchrony.

12. Conclusion

The marriage of auditory and visual experiences stands as a testament to the blend of art and technology. This final year project, focusing on the generation of live computer visuals based on music sound waves, showcases the synergistic capabilities of marrying these two sensory domains.

Sound, in its essence, is a myriad of waves with varying amplitudes and frequencies. To harness the information embedded within, the project employed the `meyda.js` library, proficient in real-time audio feature extraction. Specifically, features such as spectral centroid, RMS, and chroma became the pillars upon which the visual components were constructed. This extraction process served as a crucial bridge, transforming auditory nuances into visual expressions.

With the data in hand, the `p5.js` library was the tool of choice to breathe life into this data. As observed in the `script.js` and index.html files, the amplitude and frequency data serve as the driving force behind the visual renderings. From the pulsating movements to the dynamic colour shifts, the graphics present on the canvas are a direct reflection of the music's rhythm, pitch, and intensity.

However, visuals in this project were not merely about geometric patterns or colours but were elevated with atmospheric effects, potentially linked to `flowWorker.js`. These effects add depth, creating an immersive experience that transcends a mere visual representation of sound.

Colour dynamics, a vital component of the visual narrative, appear intertwined with functionalities possibly associated with `chroma.js`. While the exact intricacies of how colours are manipulated were not explicitly delineated in the `script.js` file, it's evident that the choice and variation of hues play a pivotal role in conveying the emotion and tone of the music.

Navigating the intricacies of real-time audio processing and visual rendering was no straightforward task. The project's code reflects a balance of technical finesse and artistic flair, each line meticulously crafted to ensure that the visuals not only respond to the music but also resonate with the viewer.

Looking ahead, this project's foundational research opens doors for numerous advancements. From enriching live performances to potential therapeutic uses, the scope is expansive. With technological advancements on the horizon, there's a promising potential for these visuals to become increasingly immersive, possibly integrating with augmented and virtual reality platforms.

In conclusion, this project acts as a guiding light, highlighting avenues for future initiatives at the intersection of music, art, and technology. The process, as echoed in the code, has been marked by exploration, novelty, and creative expression. As this research phase concludes, it's clear that this represents merely the beginning in the engaging tale of music-inspired computer visuals.

13. Recommendations & Future Work

The development of computer-generated visuals based on sound waves provides an enthralling avenue to synergise auditory and visual experiences. The current project serves as a testament to this synergy. Yet, as with any technological venture, there's room for further refinement and expansion.

a. **Refining Visual Response Mechanisms:**
The system employs the Meyda library for extracting audio features, which are then used to generate visual effects. While the current visual reactions are compelling, more intricate mappings between specific audio features (like spectral flux or chroma) and visual elements could be explored. This might involve complex animations, colour transitions, or patterns that respond to minute nuances in the audio.

b. **User Interaction and Control:**
The existing interface in index.html provides a platform for users to experience the visuals. Enhancing this with a user-friendly control panel, where listeners can adjust parameters like visual intensity, colour schemes, or even choose different visualisation patterns, can make the experience more interactive.

c. **Support for Multiple Audio Inputs:**
Currently, the system processes audio from a single source. A future extension could involve processing and visualising multiple tracks or inputs simultaneously, creating a layered visual experience.

d. **Adaptive Design Enhancements:**
While the system operates in a browser environment, ensuring that the visuals are optimally displayed across different devices (from desktops to mobile devices) is crucial. Improving the responsiveness of the index.html layout can achieve this.

e. **Performance Optimisations:**
The system's performance, especially when dealing with high-frequency updates, is crucial for a seamless user experience. Exploring the potential of Web Workers, as seen with `flowWorker.js`, can ensure that visual rendering doesn't hinder the main thread, ensuring smoother visual transitions.

f. **Integration with External Music Sources:**
Allowing users to pull music from popular platforms or even local files can broaden the usability of the system. This would provide users with the flexibility to visualise their favourite tracks or explore how different genres visually manifest.

g. **Fractal Visualisations:**
As I look forward to the next stages of this project, I'm particularly intrigued by the potential of implementing recursive fractal Mandelbrot visuals. Such an endeavour would undoubtedly benefit from resources like "The Hardest Trip - Mandelbrot Fractal Zoom" by Maths Town, which offers a mesmerising exploration of the Mandelbrot set (Maths Town, date not provided).

h. **3D Visualisations:**
While the current system focuses on 2D visualisations, there's potential to delve into

the third dimension. Incorporating libraries like Three.js could bring about a more immersive 3D visual experience, where sound waves sculpt and animate the visual landscape.

i. **Expanding the Educational Aspect:**

The project, in its essence, is a visual representation of sound physics. This can be leveraged for educational purposes, helping students grasp complex concepts related to sound waves, frequencies, and digital signal processing.
Community Engagement and Feedback Loop: Encouraging users to provide feedback, share their custom visualisations, or even contribute to the codebase can drive the project's evolution. Establishing a community around the project can lead to innovative use-cases and improvements.

j. **Documentation and Tutorials:**

As the project grows, ensuring comprehensive documentation becomes paramount. This includes user manuals, developer guides, and potential integration pathways with other platforms or systems.

In closing, this project serves as a nexus between music and visual arts, harnessing the power of web technologies. With the recommendations outlined above, there lies a roadmap to not just refine the existing framework but to expand it, touching myriad realms from entertainment to education. By embracing these avenues, the system can mature into a versatile tool, bridging auditory and visual worlds.

14. References

p5.js, n.d. [Online]. Available at: https://p5js.org/ [Accessed on 25 August 2023].

Meyda, n.d. [Online]. Available at: https://meyda.js.org/ [Accessed on 25 August 2023].

chroma.js, n.d. [Online]. Available at: https://gka.github.io/chroma.js/ [Accessed on 25 August 2023].

p5.sound.js, n.d. [Online]. Available at: https://p5js.org/reference/#/libraries/p5.sound [Accessed on 25 August 2023].

Tame Impala, (2022). [Online]. Available at: https://www.youtube.com/watch?v=EiKbH0XtI_I [Accessed on 25 August 2023].

Nishanc, (2020) [Online]. Available at: https://nishanc.medium.com/audio-visualization-in-javascript-with-p5-js-cf3bc7f1be07 [Accessed on 25 August 2023].

The Coding Train, n.d. [Online]. Available at: https://www.youtube.com/channel/UCvjgXvBlbQiydffZU7m1_aw [Accessed on 25 August 2023].

The Coding Train, n.d. [Online]. Available at: https://www.youtube.com/watch?v=Pn1g1wjxl_0&list=PLRqwX-V7Uu6aFcVjlDAkkGIixw70s7jpW [Accessed on 25 August 2023].

Franks Laboratory, (2021) [Online]. Available at: https://www.youtube.com/watch?v=VXWvfrmpapI&list=PLAC6zBFRww4p27nezfuVz02oA_y5OHCWP&index=5 [Accessed on 25 August 2023].

Karlsson, J., (2017) [Online]. Available at: https://codepen.io/DonKarlssonSan/post/particles-in-simplex-noise-flow-field [Accessed on 25 August 2023].

Siqueira, R., (2020) [Online]. Available at: https://openprocessing.org/sketch/1019806 [Accessed on 25 August 2023].

Maine JayBird, (2018) [Online]. Available at: https://www.youtube.com/watch?v=9YmVPCgSrCo&list=PLAC6zBFRww4p27nezfuVz02oA_y5OHCWP&index=18 [Accessed on 25 August 2023].

King Gizzard & The Lizard Wizard, (2015) [Online]. Available at: https://www.youtube.com/watch?v=PUZn1I6llJs&list=PLAC6zBFRww4p27nezfuVz02oA_y5OHCWP&index=15 [Accessed on 25 August 2023].

DJ IRYNA, (2020). [Online]. Available at: https://www.youtube.com/watch?v=v007jDwJ_Ts [Accessed on 25 August 2023].

Trippy Visuals, (2015) [Online]. Available at: https://www.youtube.com/watch?v=wFIscttMlk4&list=PLAC6zBFRww4p27nezfuVz02oA_y5OHCWP&index=12 [Accessed on 25 August 2023].

Maths Town, (2020) [Online]. Available at: https://www.youtube.com/watch?v=LhOSM6uCWxk [Accessed on 25 August 2023].

HDCOLORS, (2011) [Online]. Available at: https://www.youtube.com/watch?v=q2fIWB8o-bs&t=2999s [Accessed on 25 August 2023].

Trancentral, (2021) [Online]. Available at: https://www.youtube.com/watch?v=QN2yaq69Fmc

[Accessed on 25 August 2023].

Ng, K., Armitage, J. & McLean, A. (2014). The Colour of Music: Real-time Music Visualisation with Synaesthetic Sound-Colour Mapping. [pdf] Centre for Digital Music. Available at: https://www.scienceopen.com/hosted-document?doi=10.14236/ewic/EVA2014.3 [Accessed on 25 August 2023].

Watkins, J. (2018). Composing Visual Music: Visual Music Practice at the Intersection of Technology, Audio-visual Rhythms & Human Traces. [pdf] ResearchGate. Available at: https://www.researchgate.net/publication/324242145_Composing_Visual_Music_Visual_Music_Practice_at_the_Intersection_of_Technology_Audio-visual_Rhythms_&_Human_Traces [Accessed on 25 August 2023].

Graf, M., Opara, H.C. & Barthet, M. (2021). An Audio-Driven System For Real-Time Music Visualisation. [pdf] Centre for Digital Music, Queen Mary University of London. Available at: https://arxiv.org/pdf/2106.10134.pdf [Accessed on 25 August 2023].

Oppenheim, A. V., & Schafer, R. W. (2010). Discrete-time signal processing. Pearson.

Cooley, J. W., & Tukey, J. W. (1965). An algorithm for the machine calculation of complex Fourier series. Mathematics of Computation, 19(90)

Müller, M. (2015). Fundamentals of music processing. Springer.

15. Appendices

A. JavaScript Files
   - `flowWorker.js`: A custom script containing functions related to calculations and data processing.
   - `script.js`: The main script for the project that contains initial variable declarations.
   - `chroma.js`: An external library used for colour conversions.
   - `meyda.js`: An external library related to audio processing, which has dependencies on 'dct' and 'fftjs' modules.
   - `p5.js`: The renowned p5.js library (version 1.7.0) used for creative coding and graphics.
   - `p5.sound.js`: An extension of the p5 library (version 1.0.1) focused on audio.

B. HTML File - The main webpage for the project titled 'Live Computer-Generated Visuals Based on Sound Waves of Music'.
   It incorporates the following JavaScript files:
   - `p5.js`
   - `p5.sound.js`
   - `meyda.js`
   - `chroma.js`
   - `script.js`