Masterarbeit zum Thema

Konzeption und Implementierung einer interaktiven Datenvisualisierung zur Recherche von musikalischen Rekonstruktionen

Zur Erlangung des Grades

Master of Science

im Studiengang Audiovisual Arts Computing
an der Technischen Hochschule Ostwestfahlen-Lippe

Erstprüfer / Betreuer
Prof. Dr.-Ing. Aristotelis Hadjakos

Zweitprüfer
Prof. Dipl.-Des. Heizo Schulze

Eingereicht im Sommersemester 2023
von Yannick Schmitz
Matrikel-Nr. 15456096

Inhaltsverzeichnis

Abkürzungsverzeichnis

AR	Augmented Reality
API	Application Programming Interface
CD	Compact Disc
CPU	Central Processing Unit
CSV	Comma-Separated Values
CSR	Cover Song Retrieval
D3	Data-Driven Documents
DJ	Discjockey
DXR	Data Extended Reality
FPS	Frames Per Second
IBM	International Business Machines Corporation
ID	Identifikation
iOS	Internetwork Operating System
JS	JavaScript
JSON	JavaScript Object Notation
KI	Künstliche Intelligenz
MER	Music Emotion Recognition
MIR	Music Information Retrieval
PKCE	Proof Key for Code Exchange
UI	User Interface
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VR	Virtual Reality
WebGL	Web Graphics Library

Abbildungsverzeichnis

1. Einleitung

1.1 Motivation

Der Musikkonsum hat sich zunehmend in den Alltag integriert. Die Möglichkeit Musik auf Abruf nutzen zu können, vereinfacht den Konsum erheblich. Streaming-Plattformen wie Spotify, Amazon Music, Apple Music nutzen clevere Algorithmen, um die Benutzerfreundlichkeit und Nutzungsdauer zu steigern. Interpreten überschwemmen den Markt mit zielgerichteten Musikstücken. Ausgerichtet auf die Mehrheit, bestimmen Algorithmen die Entwicklung der Musikproduktion. Dazu wird überwiegend auf Vorhandenes zurückgegriffen und bereits erfolgreiche musikalische Hit-Patterns fließen in die Produktionsprozesse ein, um die Masse zu überzeugen. Der moderne Ausdruck dieser Handlungsweise wird als musikalische Interpolation bezeichnet. Mit den generierten Audioeigenschaften der Spotify-API lassen sich diese Zusammenhänge auf der klanglichen Ebene weitestgehend zurückverfolgen. Auch vergleichbare Musiktitel können schnell herausgefiltert werden (Vgl. Gernitz 2018: 5-8). Was fehlt, ist eine benutzerfreundliche Anwendung, mit der sich vergleichbare Musiktitel leicht finden lassen. In dieser Masterarbeit soll ein Lösungsansatz für diese Problemstellung entwickelt werden.

Die Entwicklung und Verbesserung solcher Anwendungen ist ein sehr relevanter Bestandteil in der Medienindustrie. Die Vermarktung und Vertreibung digitaler Audio-Inhalte können mit diesen Anwendungen optimiert werden. Mit der API von Spotify wird es dem Entwickler möglich gemacht, in den Prozess der Audioanalyse mit einzusteigen und ein geeignetes Clustering zu entwerfen. Die Programmierschnittstellen sind in der Entwicklung derzeit eines der wichtigsten Werkzeuge bei der Implementierung fortschrittlicher Softwarelösungen. Sie dienen als Kommunikationsmittel höherer Programmiersprachen und beschränken sich nicht auf einen spezifischen Anwendungstyp. Die vorliegende Masterthesis liefert daher einen Einblick in den Implementierungsprozess einer interaktiven Datenvisualisierung, die ein tieferes Verständnis für die Programmierschnittstelle der Spotify-API vermittelt.

1.2 Zielsetzung und Gliederung der Arbeit

Schwerpunkt dieser Masterthesis ist die Entwicklung einer interaktiven Datenvisualisierung mit dem thematischen Hintergrund musikalischer Rekonstruktionen. Anhand der Audio-Features der Spotify-API werden die musikalischen Eigenschaften klassifiziert und eine Suchanfrage korrelierender Songs durchgeführt. Zudem liefern diese Audio-Features den Ausgangspunkt der Datenvisualisierung. Das Leitmotiv der Anwendung ist die individuelle Erkundung verwandter Musiktitel. Die Studie untersucht folgende Fragestellung: Können Rekonstruktionen musikalischer Werke mit geeigneten Audio-Features erkannt und visualisiert werden?

Ziel dieser Masterarbeit ist es, die Konzeptionierung und Implementierung der Anwendung zu diskutieren und anschließend zu evaluieren. Behandelt werden die Themenfelder der musikalischen Rekonstruktion und die Gestaltung der Benutzeroberfläche. Zudem werden die Hintergrundprozesse der Anwendung genauer erläutert und im Kontext der Spotify API-Datensätze gebracht. Diese Masterarbeit richtet sich an Entwickler und Datenwissenschaftler, die einen Einblick in die nicht-lineare Repräsentation audiospezifischer Datensätze erhalten möchten. Zudem liefert die Anwendung einen Mehrwert für Musikinteressierte und Medienschaffende, die in den Musikdatenbanken von Spotify eine spezifizierte Recherche betreiben wollen. Aus der zu untersuchenden Fragestellung lassen sich neue Ansätze und Perspektiven für die Ausarbeitung interaktiver Kommunikationsmittel gewinnen.

Im Grundlagenteil dieser Arbeit werden die Eigenschaften allgemeiner Musikinhalte im Kontext der Music Emotion Recognition und Music Information Retrieval gebracht. Das darauffolgende Kapitel listet die infolgedessen resultierenden Parameter mit den Erkenntnissen des Spotify-Research Teams auf. Darauffolgend werden die unterschiedlichen Formen reproduzierter Musiktitel genannt und daraus eine allgemeine Definition musikalischer Rekonstruktionen hergeleitet. Der dritte Abschnitt der Grundlagen skizziert die Vorgehensweisen und Faktoren einer Datenvisualisierung. Diese Erkenntnisse werden im Verhältnis der zu erstellenden Datenvisualisierung gegenübergestellt.

Um begründete Ansätze und Inspirationsquellen für die zu erstellende Visualisierung zu finden, werden im dritten Kapitel nach geeigneten Referenzen gesucht. Diese gilt es unter den Faktoren des Datenmappings, Clusterings sowie der Darstellung von Rangfolgen zu betrachten. Im Kapitel der Konzeptionierung stehen die Auswahl der Diagrammtypen und die Darstellung der Datensätze zur Diskussion. Im Teil der Konzeptionierung werden die angefertigten Wireframes begründet und anhand dessen der Ablaufplan und die Funktionsweisen der Benutzeroberfläche dargelegt. Im Implementierungsprozess steht die Erforschung der Wahl einer geeigneten Entwicklungsumgebung im Mittelpunkt. Dazu werden verschiedene Systeme vorgestellt und mit den Anforderungen der eigenen Anwendung abgewogen. Im zweiten Teil dieses Kapitels werden die informationstechnischen Prozesse skizziert und die Funktionsweise des Suchalgorithmus für die Erfassung musikalischer Rekonstruktion beschrieben. Zum Schluss werden die Ergebnisse dieser Arbeit zusammengetragen und ein Lösungsansatz für die Fragestellung hergeleitet. Es folgt ein Resümee aus den gewonnenen Erkenntnissen und ein Ausblick auf potenziell zukünftige Projekte und Studien.

2. Thematische Grundlagen der Visualisierung

2.1 Eigenschaften von digitalen Audio-Inhalten

Mit der Ausweitung größerer Datenbestände digitaler Tonträger bedarf es neuer Analysemethoden, um individuell angepasste Wiedergabelisten zu erstellen. In der Datenverarbeitung sind diese mit ausgefeilten Algorithmen realisierbar. Dies geschieht durch die Datenerhebung des Nutzerverhaltens. Dazu werden aus den Suchanfragen und Verlaufslisten personalisierte Datensätze erstellt und anschließend analysiert. Dieses Kapitel beschreibt verschiedene Vorgehensweisen und Analysemethoden, wobei insbesondere die Methoden des Streamingdienstes Spotify näher beleuchtet werden.

Seit den ersten Kompressionsverfahren digitaler Audiodaten ab 1995 haben sich die Distributionskanäle ausgedehnt. Die Tendenzen, Tonträger wie CDs und Vinyl-Platten zu sammeln, wurde durch Abonnements der Streaming-Dienstleister abgedrängt. Musikverlage und -vertriebe verlieren an Bedeutung. Für Musikschaffende eröffnen sich Risiken und Chancen. Durch die Vielzahl an Kanälen wird eine breite Menge angesprochen. Der einzelne Stream eines Musiktitels ist jedoch schlechter vergütet als der Verkauf eines physischen Tonträgers. Daher sind Musikschaffende daran interessiert, den Erwartungen der breiten Menge zu folgen und Marktanalysen durchzuführen (Vgl. Kunz 2020: 1).

Der Musik-Streamingdienst Spotify ist in vielen Ländern Marktführer und bietet ein breites Spektrum an Musikdatenbanken und Analyse-Tools. Vor allem Software-Entwickler profitieren von den frei nutzbaren Programmierschnittstellen. Diese Ressourcen sind vor allem für die Erstellung individueller Wiedergabelisten und Musikvorschläge nutzbar (Vgl. Kunz 2020: 3). Um die Vorschläge zu generieren, bedarf es einer Klassifizierung der einzelnen Musikdaten. Die Forschungsrichtung *Music Emotion Recognition* ist ein Teilgebiet der *Music Information Retrieval.* Diese baut darauf auf, Musikinformation auf der Ebene der menschlichen Emotionen zu extrahieren. Der Bereich setzt sich aus Themenfeldern wie der Musiktheorie, der digitalen Signalverarbeitung, der Psychologie und dem maschinellen Lernen zusammen.

Ausgangsmaterial für die Analysen sind Audiosignale, Partituren oder Texte. Als Beispiel lässt sich anhand der Konsonantenharmonie und Tonart eine positive oder negative Assoziation des Musikstücks ableiten (Panda 2021: 2).

Um viele Konsumenten an das Unternehmen zu binden versucht Spotify das Musikerlebnis jederzeit neu zu gestalten. Das Entwicklerteam verfolgt die Strategie den Zugang der riesigen Musikbibliothek durch personalisierte Empfehlungen zu erleichtern. Ermöglicht wird dies durch das kollaborative Filtern vergangener Hördaten aller Benutzer. So lassen sich Verbrauchsmuster erkennen, die mit den individuelle Suchverlauf verknüpft werden. Solche inhaltsbezogenen und nicht auditive Vorgehensweisen geben wenig Spielraum für Neuerscheinungen oder unbekannten Musiktitel. Diese werden vereinzelt in den Verbrauchsmuster adaptiert und besitzen wenig Aussichten auf Erfolg. Da täglich ca. 40.000 Musiktitel neu in dem Streaming-Dienstleister aufgenommen werden, führt dies zu einem Kaltstartproblem für Neuerscheinungen. Daher ist Spotify darauf angewiesen auf zusätzliche Informationsquellen zurückzugreifen (Vgl. Panda 2021: 2).

Der Musik-Nachrichtendienst The Echo Nest bietet dem Spotify-Algorithmus eine computerbasierte Datenextraktion zur Verfügung, welche auf digitaler Signalverarbeitung beruht. Diese Algorithmen ermöglichen Audiosignale nach Eigenschaften wie der Tanzbarkeit und weiteren Faktoren einzuschätzen. Bis heute ist dieses Verfahren fortgeschritten und kommerziell einsetzbar. Es existieren jedoch weiterhin Herausforderungen zuverlässige Schlussfolgerungen über die emotionale Einordnung einer Musik zu treffen (Vgl. Panda 2021: 2).

Aus der Music Emotion Recognition lassen sich Ansätze für den Vergleich korrelierender Musiktitel finden. Die definierten Audioeigenschaften der Spotify-API können dazu genutzt werden musikalische Werke voneinander abzugleichen und Ähnlichkeiten herauszufiltern. In der folgenden Liste werden die relevanten Audioeigenschaften für die Datenvisualisierung im Einzelnen zusammengefasst:

Gruppe	Kategorie	Beschreibung	Wertebereiche
Mood		Beschreibt die Stimmung bzw. Laune des gesamten Songs.	
	Danceability	Basierend auf Rhythmus-Stabilität, Tempo, Schlagstärke und dem gesamten Takt-Zyklus wird die Tanzbarkeit ermittelt.	0.0 bis 1.0 (float)
	Valence	Beschreibt die Fröhlichkeit bzw. vermittelte Positivität.	0.0 bis 1.0 (float)
	Energy	Aufbauend auf die wahrgenommene Lautstärke, Dynamikbereich und gesamte Entropie wird die Intensität bzw. Aktivität gemessen.	0.0 bis 1.0 (float)
	Tempo	Erfasst in Schlägen pro Minute wird das gesamte Tempo geschätzt	BPM (int)
Properties		Allgemeine Eigenschaften über die Audiospur	
	Loudness	Gesamtlautstärke gemessen in Dezibel	-60.0 bis 0.0 db (float)
	Speechiness	Ermittelte Präsenz von gesprochenen Worten. (0.66 bis 1.0 = Audiospur besteht ausschließlich aus gesprochenen Worten, 0.33 bis 0.66 = Audiospur enthält musikalische und gesprochene Elemente, 0.0 bis 0.33 = Audiospur besitzt keine Sprachelemente, nur Musikinstrumente).	0.0 bis 1.0 (float)
	Instrumental-ness	Anteil der gespielten Instrumente in Abwägung des Gesangs. Dabei liegt der Fokus zwischen Rap und Gesang	0.0 bis 1.0 (float)
Notation		Umfasst die Eigenschaften, die für die Erfassung musikalischer Formen verantwortlich sind.	
	Key	Registriert die jeweilige Tonart des Liedes (-1 = No Key detected, 0 = C, 1=C#, … 11=B)	-1 bis 11 (int)
	Time Signature	Erfasst die Taktart des Liedes (1=1/4Takt, 2=2/4Takt, 7=7/4)	1 bis 7 (int)
	Mode	Unterscheidet den Song in der jeweiligen Tonart (0=Minor und 1=Major)	
Context		Beschreibt das Umfeld der Audioaufnahme	
	Liveness	Geschätzte Präsenz eines Publikums anhand der Aufnahme	0.0 bis 1.0 (float)
	Acousticness	Misst den akustischen Anteil einer Spur in Gegenüberstellung eines elektronisch produzierten Songs	0.0 bis 1.0 (float)

Tabelle 1 Audio-Features der Spotify-API (Vgl. developer.spotify.com 2022: Get Track´s Audio Features)

Die Mechanismen, die bei MIR-Lösungen greifen, sind zu einem Teil komplexe Prozesse. Für die Erfassung der musikalischen und klanglichen Eigenschaften werden mehrere signalverarbeitende Algorithmen eingesetzt. Partiell wird auf bestehende Komponenten zurückgegriffen, wie beispielsweise die Spracherkennung. Beim maschinellen Lernen werden in dem Prozess überwiegend zusätzlich entwickelte Funktionen mit eingebunden. Aufgrund des Fortschritts steigender Rechenleistung und des möglichen Zugriffs auf große Datenbanken konnte sich die Methode des Deep Learning durchsetzen. Große Datensätze lassen sich durch KI-gestützte Features verarbeitet und auswerten (Vgl. Panda 2021: 2).

Wie dieses Kapitel zeigt, sind die extrahierten Audioeigenschaften der Spotify-API ein zentraler Faktor, um Songs auf emotionaler Ebene zu klassifizieren. Diese Funktionen liefern daher einen effektiven Ansatz, Musiktitel auf Basis der Wirkung miteinander zu vergleichen. Die oben dargestellte Liste bildet die Grundlage für die Wahl der zu vergleichenden Parameter in der Datenvisualisierung. Inwiefern eine Gegenüberstellung und Analyse mit den Parametern möglich ist, wird in der Phase der Konzeptionierung deutlich (s. Kapitel 4). Für die Extrahierung von Informationen digitaler Audio-Inhalte sind unzählige weitere Ansätze vorhanden. Die Parameter der Spotify-API bieten einen schnellen und einfachen Zugriff dieser Daten.

2.2 Rekonstruktionen in der Musik

Mit der Entwicklung leistungsstarker Heimcomputer ermöglichen es Medienschaffende, hochwertige Audioproduktionen durchzuführen. Musiker sind in der Lage, mit einem geeigneten Audio-Interface, Mikrofon und Laptop Musik zu produzieren. Mit wenigen Limitierungen liefern sample-basierte Orchesterinstrumente und Synthesizer ein umfangreiches Repertoire an Möglichkeiten. Mit zunehmendem Arbeitsspeicher und Prozessorleistung sind mehr Audiospuren verfügbar und können digital verarbeitet werden. Durch die Einführung zahlreicher Streaming-Plattformen ist die Verbreitung und Veröffentlichung von Musik für jeden Musikschaffenden einfacher gestaltet (Jacke 2006: 114).

Die Abhängigkeit von Musiklabels und Publisher ist stark gesunken. Diesen riesigen Pool an neuen Musikinhalten schöpft die Breite der Musikbranche aus. Potenzielle Hit-Songs werden stündlich produziert. Die wenigsten Musikstücke schaffen es in die Charts. Auf erfolgreiche Musik zurückzugreifen und daraus neue Stücke zu entwerfen, vereinfacht diesen Vorgang. Das angestrebte Ziel der musikalischen Rekonstruktion besteht überwiegend darin, eine maximale Reichweite an Konsumenten anzusprechen (Jacke 2006: 115).

Die in dieser Arbeit beschriebene Anwendung zur Musikrecherche soll in der Lage sein, musikalische Zusammenhänge herauszufiltern und zu visualisieren. Wie in Abschnitt 2.1 genannt, sind die Audioeigenschaften der Spotify-API ein ausschlaggebender Faktor für die Gegenüberstellung zweier Songs. Im Folgenden werden die Kategorien und Faktoren musikalischer Zusammenhänge beschrieben sowie anschließend eine allgemeine Definition der musikalischen Rekonstruktion hergeleitet.

2.2.1 Reproduktion: Sampling, Remix und Mashup

Die Reproduktion von Materialien startete in den 1830er Jahre mit der Entwicklung der Fotografie. Mittels eines Phonogramms wurden ab 1870 die ersten Versuche unternommen, Tonsignale aufzuzeichnen. Mit diesen Medien war es möglich, die Gegenwart auf auditiver und visueller Basis zu archivieren. Diese Art der mechanischen Reproduktionen gaben zunächst wenig Spielraum für die Umgestaltung beziehungsweise Verfälschung aufgezeichneter Materialien. 1920 wurden erste Ausführungen erbracht, aus mehreren Fotografien Collagen und Montagen zu erstellen. Dieser erste Schritt der Wiederverwertung führte zu einem neuen Werk mit einer eigenen Interpretation. Mit dem Verfahren Tonaufnahmen zu duplizieren, wurde 1970 das Sampling für die Musikkomposition entdeckt. Der technologische Fortschritt verbesserte das Equipment und eröffnete zunehmend die Möglichkeiten, Tonmaterial zu manipulieren. Somit konnten weitere kreative Prozesse freigesetzt werden, die zu einer Verzerrung des ursprünglichen Werkes führten (Vgl. Navas 2012: 17).

Mit der Einführung der ersten Heimcomputer 1980 des Herstellers IBM wurde die Bearbeitung digitaler Medien für viele zugänglich. Die noch kostspieligen Computer boten Programme wie Photoshop an, mit denen in Echtzeit Bilder präpariert wurden. Gleichzeitig ergab sich die Möglichkeit, analoge Tonspuren mit einem Audio-Digital-

Wandler zu digitalisieren, welche ebenfalls eine Bearbeitung mittels eines Computers vornehmen lies. Die Audiobearbeitung und der Audioschnitt wurden unkomplizierter. Computerbasierte Systeme eröffneten neue Ansätze in der Mediengestaltung. Heimcomputer integrierten sich mehr im kreativen Prozess und wurden ein Teil der Pop-Kultur (Vgl. Navas 2012: 20-22).

Mit der digitalen Bearbeitung kam eine weitere Form der musikalischen Neuinterpretation zustande. Die ersten Prinzipien des sogenannten Remixes lassen sich auf die Kulturhochburg New York City zurückführen. Ziel war es, eine neue Mischung zwischen dem Originalwerk und der eigenen Interpretation zu erstellen. Ein Remix setzt sich aus mehreren Teilstücken eines Werkes zusammen, welche sich zu einer neuen Klangcollage vernetzen. Es bildet daher die darauffolgende Stufe des zuvor angesprochenen Samplings. Diese kreative Methodik führte zu einer neuen Generation von Musikproduzenten in der westlichen Hochkultur. Die Konzepte sind auf jedes Musikgenre übertragbar und finden sich überwiegend im Hip-Hop und in der elektronischen Tanzmusik wieder (Vgl. Navas 2012: 20-22).

Die nächste Ebene des Samplings und Remixes bildet das Mashup. Im Wesentlichen bezieht sich ein Mashup nicht auf ein spezifisches Werk, sondern setzt sich aus mehreren Werken zusammen. Eine weit verbreitete Herangehensweise ist, Gesangsspuren mit Instrumental-Spuren unterschiedlicher Werke zu kombinieren. Ein prominentes Beispiel ist das Mashup *A Stroke of Genie-us* des DJs *Freelance Hellraiser*. Kombiniert werden bei diesem Mashup Songs der Interpreten *The Strokes* und *Christina Augilera*. Durch die Überlagerung der beiden Werke ergibt sich eine neue Fassung mit einem Wiedererkennungswert. Die Grundidee eines Mashups ist ein Werk, in einen anderen stilistischen Kontext zu bringen. Dazu sind vorwiegend Songs verknüpft, welche musikalisch miteinander funktionieren und aus einem anderen Stil oder Genre stammen (Vgl. Navas 2012: 93).

Wie bei einem Remix, besteht die Grundlage eines Mashups aus vielen Samples. Diese umfassen meist ganze Formteile, wie Refrain oder Strophen. Im Zusammenspiel ergeben diese eine neue Wirkung. Die Motivation hinter Mashups besteht primär im künstlerischen Sinne und beabsichtigt keinen gesellschaftskritischen Hintergrund (Vgl. Navas 2012: 93).

2.2.2 Coversongs

Die Idee einer Coverversion eines musikalischen Werkes besteht darin, Melodie und Text von einem anderen Interpreten wiederzugeben. Die Grenze zwischen der Imitation und eigener Interpretation ist meist fließend und daher nicht immer eindeutig zu definieren. Melodie und Text können stark abgewandelt sein und die Rechtsfrage bei einem Duplikat ist dabei nicht immer eindeutig. Coverversionen sind daher individuell zu betrachten und die Auslegung des kreativen Mitwirkens vom Interpreten abhängig. Unter dem Gesichtspunkt der Schöpfungshöhe ist zu klären, ob das neue Werk das Urheberrecht verletzt oder die Interpretation einer eigenen Leistung übersteigt. Diese Debatte tritt dann in Kraft, sobald Paraphrasen eines anderen Werkes Ähnlichkeiten aufweisen. Im Bereich der *Cover Song Retrieval* wird untersucht, unter welchen Bedingungen und Faktoren Coverversionen zu erkennen sind. Diese Unterkategorie der *Music Information Retrieval* konzentriert sich auf die Zusammenstellung extrahierter Musikinformation in der Konstellation vieler musikalischer Werke. Bei CSR handelt es sich nicht um Strategien der Extrahierung, sondern um das Verknüpfen vergleichbarer Songs. Bei diesem Prozess werden auf Datensätze bestehender Audio-Feature-Repräsentationen zurückgegriffen. Die Vielfalt und Variationen dieser Datensätze gestaltet die Erstellung einer geeigneten Konstellation zu einer komplexen Angelegenheit (Vgl. Liem 2009: 1-2).

Um Coverversionen in Vergleich zu setzen, sind Audioinhalte in ihren Merkmalen zu definieren. Diese lassen sich in vielen Dimensionen zerlegen, die teilweise in geringfügigen Variationen vom Originalwerk abweichen. Subtile Differenzen zeigen sich beispielsweise in der klassischen Musik. Während einer musikalischen Aufführung sind Klangfarbe, Tempo, Dynamik, Artikulation und viele weitere Faktoren stets einzigartig. Darin besteht die Absicht, Partitur und Instrumentierung des Werkes überwiegend originalgetreu wiederzugeben. In der populären Musik liegt das Bestreben überwiegend darin, eine neue Version mit eigener Interpretation aufzuzeichnen. Ein Merkmal bei der Charakterisierung eines Klangs ist das sogenannte Timbre. Diese Eigenschaft beschreibt die allgemeine Textur und Klangfarbe eines Songs. Der Unterschied zeichnet sich durch die Vorgehensweise und Instrumentierung einer Produktion bzw. einer Performance aus. Schwerpunkt der Vorgehensweise besteht in den Tonaufnahme- und Verarbeitungstechniken. Faktoren wie die Methode der Mikrofonierung, Aufnahmegerät und klangverarbeitende Prozesse wirken bei der Gestaltung einer Produktion mit. Zum anderen ist bei der Instrumentierung darauf zu achten, was als Klangquelle vor dem

Mikrofon geschieht beziehungsweise welche Sample-Instrumente zum Einsatz kommen. Ein weiteres Merkmal eines musikalischen Werks ist die Geschwindigkeit und Betonung. Kleinere Schwankungen in der Geschwindigkeit und rhythmischen Struktur einer Interpretation wirken sich auf Ausdruckskraft und Kontext des Musikstücks aus. Sollten in der Interpretation Songabschnitte wegfallen, wiederholt, neu hinzu oder in der Reihenfolge verändert werden, so ist die Rede von einer Modifikation der Struktur (Vgl. Serra 2010: 4-5).

Eine weitere Möglichkeit ein musikalisches Werk in seiner Form zu modellieren, ist das Stück in seiner Tonart zu transponieren. Dies geschieht entweder aus ästhetischen Gründen oder wird der Stimmlage der Instrumentierung oder Gesangs angepasst. Eine feinere Veränderung der Stimmung erfolgt mittels Harmonisierung. Durch die Änderung der Akkordtypen durch das Ersetzen, Hinzufügen und Weglassen, lassen sich Songs in andere Musikstile umschreiben. Ein Pop-Stück wird beispielsweise zu einer jazzigen Ballade, wenn die bestehenden Akkorde durch in der Jazzharmonik übliche modale Akkorde ersetzt werden. Aussprache und Text sind ebenfalls in der Harmonisierung einzuordnen. Die Interpretationsweise und Änderung der Sprache durch den Sänger erlauben es, ein Publikum mit einem anderen kulturellen Hintergrund anzusprechen. Ein weiteres Merkmal der Modifikation ist der Rauschanteil, welcher beim jeweiligen Aufnahmeverfahren entsteht. Dieser sondert sich vom harmonischen Signalanteil ab und zeichnet sich im Tonmaterial wieder. Gründe für den Rauschanteil sind beispielsweise die Anwesenheit eines Publikums, das während der Aufnahme jubelt und klatscht. Codierungs-Artefakte, die Audiokomprimierung und die Art der Wiedergabe sind weitere Faktoren, die zu einem Rauschanteil beitragen (Vgl. Serra 2010: 4-5).

Wie im vorherigen Kapitel genannt, lassen sich Coverversionen in Kategorien einordnen. Zum Unterschied stehen beim Sampling, Remixing und Mashup die Wiederverwertung bestehender Audiomaterialien im Vordergrund. Aus diesem Kapitel resultiert, dass über den genannten Formen weitere Variationen eines Musikcovers bestehen. Ein Cover ist nicht rein aus einer Replikation eines Audio-Ausschnittes definiert, sondern bietet ein breitgefächertes Spektrum zwischen dem Original und der eigenen Interpretation. Das Konzept eines Covers besteht darin, aus einer musikalischen Leistung eines bestehenden Werkes ein Neues zu entwickeln (Vgl. Magnus 2022: 3).

2.2.3 Interpolation

Interpolation bedeutet in der Musikwissenschaft eine Umgestaltung oder Verfälschung des Originaltitels. Dies ist in vielerlei Hinsicht erreichbar. Auf der musikalischen Ebene sind Änderungen an der Partitur oder am Metrum vorzunehmen. Veränderungen in der Klangfarbe sind durch mechanische oder digitale Verfahren möglich. Abgrenzend zu einer Coverversion besteht bei diesem Verfahren die Absicht, ein eigenes neues Werk zu entwerfen - nicht zu verwechseln mit dem sogenannten Sampling, bei dem die ursprünglichen Aufnahmen eines Musiktitels in direkter Form verwendet werden. Auf Basis der Rhythmik, Melodik oder Harmonie werden Veränderungen vorgenommen, die durch das Einspielen oder Einsingen des jeweiligen Interpreten erfolgen (Vgl. Schürmer 2018: 159-161).

In der modernen Popkultur dient die Interpolierung häufig zur Vermeidung von Lizenzproblemen. Durch die globale Kommerzialisierung in der Musikindustrie wird regelmäßig auf bestehende erfolgreiche Musikwerke zurückgegriffen. Das angestrebte Ziel der Musikindustrie besteht darin, mit seiner Produktion eine maximale Reichweite an Konsumenten anzusprechen. In Fachkreisen stellt sich die Frage, ob das Adaptieren eines bestehenden Werkes maßgeblich einen künstlerischen Anspruch erhebt und diese Praktiken eine Eigenleistung erfordern. Begriffe wie *Kommerzialität* und *Kopie* finden häufig Verwendung beim Aufgreifen und Aufbereiten von Fremdmaterial. Dieses komplexe Phänomen lässt sich ambivalent in der *kulturellen Erinnerung* und *Hommage* eines bestehenden Werkes deuten. Das Ziel der musikalischen Interpolation liegt darin, fremdes Material in einen neuen Kontext einzubetten und für ein neuen Zweck aufzubereiten. Ein korrespondierendes Verständnis der Interpolation bietet der *Eklektizismus*. Er betrachtet die Übernahme äußerer Schemata wertfrei. Eklektizismus unterscheidet zwischen eigener Weiterentwicklung und Imitation. Die Verwertung fremden Materials ist entweder positiv oder negativ einzuordnen (Vgl. Petschow 2014: 1).

2.2.4 Definition und Einordnung der musikalischen Rekonstruktion

In diesem Abschnitt werden relevante Aspekte zusammengefasst, um eine Definition der musikalischen Rekonstruktion herzuleiten. Der Begriff Rekonstruktion ist als “[...] das Erschließen und Darstellen, Wiedergeben von etwas Geschehenem in den Einzelheiten seines Ablaufs” (Dudenredaktion o.J.: Rekonstruktion) und “[...] das Wiederherstellen,

Nachbilden (des ursprünglichen Zustandes von etwas)" (Dudenredaktion o.J.: Rekonstruktion) erklärt. Im wirtschaftlichen Sinne ist es das "[...] Umgestalten, Modernisierung" (Dudenredaktion o.J.: Rekonstruktion). Erstere Bezeichnung beschreibt den Prozess, während die anderen beiden Definitionen sich auf das Ergebnis einer Rekonstruktion beschränken.

Im Kontext musikalischer Werke sind diese Definitionen mehrfach zu deuten. Ein Werk lässt sich wiedergegeben, wiederherstellen oder umgestalten. Mit dem Bezug vorangegangener Unterkapitel lassen sich diese Eigenschaften den entsprechenden Gegebenheiten zuordnen. Das Sampling bietet eine Art der Reproduktion. Bei dieser Form wird auf bestehendes Material zurückgegriffen und mit Hilfe eines konkreten Aufnahmeverfahrens archiviert. Diese Vorgehensweise ist als das Wiederherstellen bestehender Mittel zu verstehen, da die Materialien je nach Verfahren und Ansatz in einem anderen Kontext gebracht werden. Eine Coverversion bildet die Wiedergabe eines musikalischen Werkes. Das Ziel besteht darin, Melodie und Text zu interpretieren. Der Übergang zwischen der eigenen Interpretation und Imitation ist im Einzelnen nicht eindeutig zuzuordnen. Die letzte Form der Rekonstruktion beschreibt die Umgestaltung eines bestehenden Werkes. Dies trifft bei der musikalischen Interpolation zu. Wie in Unterkapitel 2.2.3 beschrieben, besteht der Zweck der Interpolation in der Umgestaltung oder Verfälschung eines ursprünglichen Werkes.

Diese Erkenntnisse sind im Kontext der eigenen Anwendung gegenüberzustellen und folgende Aussagen treffen: Das Ziel der Anwendung liegt darin, musikalische Werke herauszufiltern die entweder gesampelt, gecovert oder interpoliert sind. Nicht auszuschließen sind musikalische Werke, die kaum bis keine korrelierenden Fragmente anderer Werke aufweisen. Zu berücksichtigen ist, dass die Algorithmen und definierten Musikeigenschaften der Spotify-API nicht alle Faktoren verwandter Werke erfassen. Unter Bezugnahme dieser Tatsachen ist die zu erstellende Anwendung in der Lage, annähernde Wahrscheinlichkeiten musikalischer Rekonstruktionen aufzudecken. In dieser Arbeit gilt es zu untersuchen, ob die Anwendung einen praktikablen Nutzen für die Recherche korrelierender Werke besitzt. Music Information Retrieval bietet viele Ansätze, musikalische Werke auf emotionaler und datenbasierter Weise auszuwerten. Die Audioeigenschaften der Spotify-API liefern eine erprobte und anwendungsfreundliche Grundlage für die Umsetzung einer interaktiven Visualisierung.

2.3 Datenvisualisierung

Erkenntnisse aus großen Datenmengen zu gewinnen, ist ein komplexer Prozess. Muster in Daten zu entschlüsseln und Erkenntnisse daraus hervorzuheben ist eine Fähigkeit, die mit erweiterten Analysetechniken und technischem Fortschritt einhergeht. Die Verarbeitung mit computergestützten Systemen reduziert den Arbeitsaufwand für Daten-Wissenschaftlern. Softwarelösungen wie Excel, Tableau oder PowerBI bieten eine Vielzahl von Gestaltungsmöglichkeiten zur Einbettung von Daten in einen visuellen Kontext. Für die Visualisierung von Daten ergeben sich immer neue Routineprozeduren. Heuristiken und Deklarationen von Datensätzen stellen die zentralen Aspekte der Visualisierung dar. Wenn eine Informationsvisualisierung zum Forschungszweck angefertigt wird, ist sie aus vielen Perspektiven zu betrachten. Dabei gilt es zu klären, ob der Anspruch der Glaubwürdigkeit und das Objektivitätsideal erfüllt sind. Die gestalterische Sichtweise beschäftigt sich mit der semantischen Bedeutung der Daten und greift dabei sowohl persönliche als auch gesellschaftliche Aspekte auf. Die Wahl einer geeigneten Darstellung bildet die Verknüpfung zwischen dem Empfänger und dem Datensatz. Die Gestaltung des Datenmappings in seiner Form und Farbgebung bewirkt dabei eine neue Deutung eines vorliegenden Sachverhalts. Die Beantwortung der Frage des „Warum?“ liegt beim Verfasser selbst. Schlussfolgernd ist die Aufgabe einer visuellen Darstellung die Beantwortung einer Hypothese, welche eine eindeutige Aussage widerspiegelt (Vgl. Frey 2017: 2).

Die Visualisierung von Daten erfolgt auf verschiedene Arten. Beispielsweise kann zwischen computerbasierten Systemen und statischen Infografiken unterschieden werden. Während erstere durch Animationstechniken zahlreiche Interaktionsmöglichkeiten bieten, verfolgen letztere das primäre Ziel, den Betrachter auf übersichtliche Weise zu informieren. Ziel einer Datenvisualisierung ist die Kommunikation eines bestimmten Sachverhalts. Aussagen und Ideen sind durch das Mapping der Daten ausgedrückt. Welche Erkenntnisse und Schlüsse aus dem jeweiligen Informationsgehalt hervorgehen, hängt vom Empfänger ab. Erläuterungen und eine ausdrucksstarke Repräsentation der Daten stützen Aussage und Wirkung des Autors. Um die Datensätze entsprechend der visuellen Aspekte besser eingliedern zu können, ist eine umfassende Klassifizierung der Daten notwendig. Je nach Auswertung, sind bei diesem Vorgang die Daten in die jeweilige Ebene einzuordnen. Diese dimensionale Zuordnung ist nötig, um Zusammenhänge auf sachlicher und verständlicher Weise wiederzugeben.

Um eine geeignete Datenvisualisierung umzusetzen, ist eine umfangreiche Recherche durchzuführen. Dies erfolgt durch die Einbindung bewährter Referenzmodelle (Vgl. Nazemi 2021: 477-479).

2.3.1 Datenklassifizierung

Wie im vorherigen Abschnitt angesprochen, ist die Klassifikation ein entscheidender Schritt, um Daten in einen visuellen Kontext einzubinden. Dazu lassen sich die Daten in drei verschiedene Datentypen unterteilen: Nominale Datentypen gliedern die Datensätze in Kategorien oder Namen. Diese sind im Normalfall nach dem lexikalischen System geordnet (es gilt: = oder ≠). Die Kategorisierung ist überwiegend farblich zu differenzieren. Bei ordinalen Daten handelt es sich um kategorisierte Datentypen, welche einen bestimmten Wert aufweisen. Diese Datentypen folgen einer natürlichen Ordnung und sind entweder kontinuierlich oder diskret (es gilt: < oder >). Quantitative Daten arrangieren sich nach einem arithmetischen System und sind mit Hilfe mathematischer Operationen skalierbar (es gilt: +, -, / oder *). Dies sind Wertebereiche, in denen die entsprechenden Werte zugeordnet werden. Das beschriebene Modell der Datenklassifizierung ist je nach Nutzen und Verbreitung verschieden. Diese Art der Datenzuordnung ist wissenschaftlich fundiert und bietet eine Grundlage für die Klassifizierung der Datensätze (Vgl. Nazemi 2021: 481-482).

Kategorisierte Datentypen sind je nach Datensatz in verschiedene Dimensionen unterteilt. Die Rede ist von einer ein- oder zweidimensionalen Darstellung. Darüber hinaus handelt es sich um multidimensionale Darstellungen. Eindimensionale Daten sind in diskreten Sequenzen zusammengefasst. Ereignisketten sind ein Beispiel für eindimensionale Daten, welche die Zuordnung einer zeitlichen Abfolge darstellt. In der zweidimensionalen Darstellung sind Daten in Abhängigkeit zueinander. Innerhalb dieser Form der Darstellung sind temporale, fließende oder planar-räumliche Repräsentationen möglich. Da sich diese Darstellung immer auf zwei Variablen beschränkt, wird diese als XY-Diagramm bezeichnet. Neben dem XY-Diagramm sind zahlreiche weitere Diagrammtypen vorhanden, wie beispielsweise Balken-, Kreis- oder Liniendiagramme (Vgl. Nazemi 2021: 482).

Unabhängig vom Diagrammtyp können je nach Vorgehensweise alle drei Variablentypen nominal, quantitativ oder ordinal vertreten sein. Multidimensionale Datensätze bieten erweiterte Konzepte, um mehr als zwei Variablen in einer Visualisierung zu platzieren. Dies lässt sich anhand einer Fertigungsstraße veranschaulichen, in denen unzählige Sensoren verbaut sind. Dabei erzeugen die verbauten Sensoren einen ständigen Fluss an Daten, welche transferiert und ausgewertet werden. Dabei umfasst eine Fertigungsstraße hunderte an Variablen, die zum größten Teil in Echtzeit die Daten messen und aufzeichnen (Vgl. Nazemi 2021: 483-484).

Durch die Erhöhung der Variablen steigt die Komplexität der jeweiligen Visualisierung. So ist die gezielte Interpretation hoher Datensätze mit einer geeigneten Visualisierungstechnik möglich. Eine dreidimensionale Projektion der Daten ist für die wenigsten Anwendungsfälle geeignet, da Verhältnisse - bedingt durch die Perspektive - falsch eingeschätzt werden. Ein Lösungsansatz bietet die Visualisierung durch Matrizen. In diesem Fall wird auf zweidimensionale Diagrammtypen zurückgegriffen und eine weitere Variable hinzugefügt. Diese unterscheidet sich anschließend in Form, Farbe, Größe, Bewegung oder weiteren Darstellungsweisen. Eine besondere Form bei der Datenrepräsentation innerhalb eines computergestützten Systems bietet die Interaktion. In dieser Form besteht die Möglichkeit, Datensätze bei Bedarf dynamisch anzupassen. Durch das ausgelöste Ereignis vom Betrachter liefert die Visualisierung eine neue Perspektive auf den vorliegenden Sachverhalt (Vgl. Nazemi 2021: 487-489).

2.3.2 Data-Storytelling

Mit dem Fortschritt der digitalen Medien und leistungsstarken Rechner-Einheiten spielt das Thema Big-Data eine immer wichtigere Rolle. Daten-Wissenschaftlern stehen durch den weltumspannenden Austausch an Informationen immer mehr Daten bereit. In der Wissenschaft tragen Datensätze zur Problemlösung wichtige Erkenntnisse und Analysen bei. Vor allem als Mittel des medialen Austauschs werden beim Adressaten neue Erfahrungen über einen dargelegten Sachverhalt ausgelöst. In der Wirtschaft ist das Fachwissen von Datenwissenschaftlern derzeit gefragt und trägt zentral zur Entscheidungsfähigkeit innerhalb der Unternehmensstruktur bei (Vgl. Neifer 2020: 1033-1035).

Der Verarbeitungsprozess Datensätze in Unternehmensstrategien einzubinden, ist ein aufwändiges Verfahren. Häufig sind irrelevante Stakeholder in eine Darstellung miteinbezogen. Zur Folge sind Sachverhalte nicht zielgerichtet kommuniziert. Darunter leidet die Glaubwürdigkeit der jeweiligen Datenvisualisierung. Data-Storytelling ist ein weitreichendes Gebiet, in dem rhetorische Mittel bei der Umsetzung der Visualisierung im Zentrum stehen. Die Absicht von Data-Storytelling ist, Aussagen auf effektive Weise durch die Erhebung, Aufbereitung und Gestaltung der Daten zu vermitteln. Die Fähigkeit durch die Datenvisualisierung eine Weiterentwicklung zu fördern, wird von narrativen Aspekten gestützt. Diese Aspekte verhelfen, zukünftige Entscheidungen besser einzuordnen und konstruktive Veränderungen vorzunehmen. Der Autor beziehungsweise Erzähler einer Visualisierung erfüllt die Rolle eines Dolmetschers, um zwischen Sachverhalt und Betrachter als Sender und Empfänger zu vermitteln. Mit der Bezugnahme des Hintergrunds und Herkunft der angesprochenen Zielgruppe ist es das Ziel von Data-Storytelling, den Sachverhalt auf verständliche Weise überzeugend zu kommunizieren (Vgl. Neifer 2020: 1033-1035).

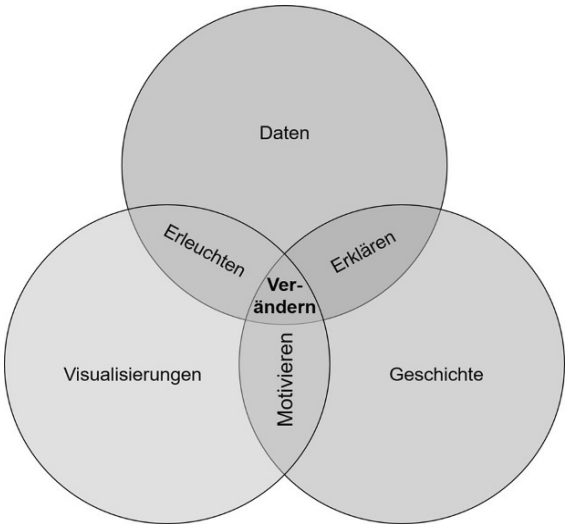


Abbildung 1 Die drei Schlüsselelemente des Data Storytellings (Dykes 2016)

Wie aus der Abbildung 1 hervorgeht, lässt sich Data-Storytelling in folgende drei Felder gliedern: Daten, Visualisierung und Geschichte. Diese Felder beschreiben zum einen die Komponenten, aus denen sich die erzählerische Methodik einer Datenvisualisierung zusammensetzt. Zum anderen veranschaulicht die Abbildung die Beziehung und Wirkung der einzelnen Felder zueinander. Die visuelle Komponente in Verbindung der Geschichte erfüllt eine unterhaltende Funktion. Diese bietet dem Betrachter ein Motiv, sich mit der Visualisierung und Thematik auseinanderzusetzen. Die Geschichte in Bezug zu den

Daten erklärt den vorliegenden Sachverhalt. Durch die dargestellten Fakten und Inhalte werden logische und kausale Zusammenhänge kommuniziert. Die Kombination aus der Visualisierung und den Datensätzen beleuchten den vorliegenden Sachverhalt unter einer anderen Perspektive. Die Aussage der Visualisierung wird mit Fakten verifiziert und entwickelt sich zu einer Tatsache. Der Betrachter erlebt eine neue Erfahrung. Im Kern dieses Modells steht die Veränderung. Die Absicht der erzählerischen Methodik besteht demnach darin, eine Veränderung beim Betrachter auszulösen (Vgl. Neifer 2020: 1035-1036).

Ein Ziel von Data-Storytelling ist es, Sachinformationen prägnant und effektiv zu vermitteln. Visualisierungstechniken verhelfen, die Aufmerksamkeit des Betrachters auf das Wesentliche zu beschränken. Dem Betrachter ist eine klare Aussage zu vermitteln. Um diese Ziele zu erreichen, sind drei grundlegende Problemstellungen zur Visualisierung im Vorfeld aufzugreifen: Zum einen ist festzustellen, an wen sich die Visualisierung richtet. Es ist vorteilhaft, zukünftige Zielgruppen in Stakeholder zu kategorisieren und relevante Eigenschaften und Interessen festzulegen. Aus den gewonnenen Informationen lässt sich ableiten, welche Hintergrundinformationen vom Betrachter vorauszusetzen sind, und welche bereitgestellt werden. Darüber hinaus liefern die Informationen den Zusammenhang zwischen sachlichem Kontext und Betrachter (Vgl. Knaflic 2015: S. 20-23).

Die zweite Problematik, die sich bei der Entwicklung einer Visualisierung stellt, ist die Beziehung zwischen dem Ersteller und Betrachter. In diesem Abschnitt geht hervor, unter welchen Bedingungen und in welchem Kontext die Visualisierung veröffentlicht wird. Innerhalb einer persönlichen Präsentation steht der Ersteller der Visualisierung in direkter Verbindung zum Betrachter. Der Ersteller ist somit in der Lage, auf Fragen einzugehen und weitere verbale Erläuterungen mitzuliefern. Ein Gegenbeispiel findet sich bei der Veröffentlichung einer Visualisierung innerhalb einer Anwendung. Unter diesen Gegebenheiten besteht kein direkter Austausch zwischen Betrachter und Ersteller. Im Regelfall sind keine verbalen Erläuterungen und Rückfragen bei der Veröffentlichung möglich. Sachverhalte und Kernaussagen sind daher verständlich darzustellen (Vgl. Knaflic 2015: S. 20-23).

Aus der letzten Problematik geht hervor, welche Informationsinhalte dem Betrachter mitgeteilt werden, um erwartete Reaktion auszulösen. Falsche Annahmen sind zu vermeiden, indem relevante von irrelevanten Aussagen unterschieden werden. Dem Betrachter sind, je nach gewünschtem Effekt, Informationen vorzuenthalten. Zudem ist es empfehlenswert, bisherige Erfahrungen und Hintergrundwissen des Betrachters aufzugreifen. Im Fall einer interaktiven Visualisierung ist die Gestaltung einer Benutzeroberfläche ein wichtiger Aspekt. Es gilt dabei zu beachten, ein gewohntes und erprobtes UI-Design für die Visualisierung zu entwickeln. Bei der semantischen Bedeutung von Symbolen, Farben, Mustern etc. ist es von Bedeutung, auf Vorerfahrungen des Betrachters einzugehen. Die Farbgestaltung ist auf viele Arten zu deuten. Beispielsweise wird die Farbe Rot zum einen mit Gefahr, Warnung und Blut assoziiert. Im Gegensatz dazu ist Rot jedoch auch mit Liebe und Wärme verbunden. Bei der gestalterischen Ausarbeitung sind daher Kontext und Gegebenheiten der jeweiligen Zielgruppe spezifisch zu berücksichtigen (Vgl. Knaflic 2015: S. 20-23).

2.3.3 Einordung der Datenvisualisierung in diese Arbeit

Das Ziel der Datenvisualisierung ist eine gezielte Repräsentation des Sachverhalts. Im Fall der in dieser Arbeit beschriebenen Datenvisualisierung ist dies der Vergleich musikalischer Rekonstruktionen. Die Aufgabe besteht darin, die Datensätze so auszuwählen und anzuordnen, dass diese vom Benutzer auf verständliche Weise präsentiert werden. Bei der Ausarbeitung der Konzeptionierung sollten die wesentlichen Aspekte einer effektiven Datenvisualisierung berücksichtigt werden. Dazu gehört, die benötigten Datensätze der Spotify-API zu klassifizieren und anschließend in den Kontext einzubetten. Unter Bezugnahme der aufgelisteten Audioeigenschaften aus Abschnitt 2.1 lassen sich die Tabelleninhalte den spezifischen Datentypen zuordnen.

Bei den Gruppierungen der Kategorien (Mood, Properties, ...) handelt es sich um nominale Daten. Wie im Kapitel der Datenklassifizierung angesprochen, sind diese kategorisch voneinander zu unterscheiden. Im Regelfall sind nominale Daten farblich differenziert und nach lexikalischen Systemen geordnet. Die eigentlichen Kategorien (Danceability, Valence, …) lassen sich ebenfalls den nominalen Datentypen eingliedern. Die dazugehörigen Messdaten der Kategorien gehören im Regelfall zu den ordinalen Datentypen und folgen den natürlichen Ordnungsgrößen. Die entsprechenden Wertebereiche (0.0 bis 1.0, -60.0 bis 0.0 db, …) der jeweiligen Messdaten sind den

quantitativen Datentypen zuzuordnen. Diese folgen dem arithmetischen System und sind durch mathematische Operationen skalierbar.

Die Anzahl der in einer Visualisierung dargestellten Dimension hängt von den ordinalen und nominalen Datentypen ab. Im Fall der zu erstellenden Anwendung *Comparify* liefert die Liste aus Abschnitt 2.1 die ersten Anhaltspunkte, welcher Datentyp zur jeweiligen Dimension zuzuordnen ist. Dabei lassen sich die Zugehörigkeiten der Kategorien in einer Dimension zusammenfassen. Bei den Kategorien hängt die Dimensionalität von der jeweiligen Einheit und Wertebereich ab. So sind Werte wie Schläge pro Minute nicht mit der gemessenen Lautstärke kompatibel. Diese Einheiten sind in separate Dimensionen zu verschachteln, um Wahrnehmungsverzerrungen beim Vergleich einzelner Messdaten zu vermeiden.

Aus diesen Erkenntnissen lässt sich ableiten, dass es sich bei der Visualisierung um eine mehrdimensionale Darstellung handelt. Zudem besteht die Möglichkeit, die jeweiligen Dimensionen durch Bewegung und Interaktion auszudrücken, da es sich bei der Visualisierung um eine interaktive Anwendung handelt. Die Datensätze sind zudem dynamisch und lassen sich nach den Gegebenheiten anpassen. Dies ist notwendig, da der Benutzer in der Lage sein muss, eine Auswahl an musikalischen Werken für die Gegenüberstellung zu treffen. Die Entscheidung eines geeigneten Diagrammtypens ist eine weitere Komponente, die bei der Erstellung einer geeigneten Visualisierung eine wichtige Rolle spielt. Die Aufgabe des Diagrammtypens besteht im Fall der musikalischen Rekonstruktion darin, einen Vergleich zweier Werke zu setzen. Demnach ist die Anforderung des Diagramms, alle Faktoren darzustellen und visuell abzugleichen. Der Betrachter ist somit in Lage, die Verhältnisse korrelierender Songs einzuschätzen und Unterschiede zu erkennen.

Ziel der narrativen Methodik in der Datenvisualisierung ist es, die Aufmerksamkeit auf das Wesentliche zu beschränken und die zentrale Aussage der entsprechenden Visualisierung zu verdeutlichen. Wie im vorigen Kapitel beschrieben, stehen in der Datenwissenschaft verschiedene Modelle und Ansätze zur Verfügung. Abbildung 1 aus dem Kapitel veranschaulicht die Zusammenhänge zwischen den einzelnen Themengebieten der Data Science. Die Aspekte zu entwickelnden Datenvisualisierung lassen sich in folgende Themengebiete einordnen: Der Sachverhalt der Geschichte besteht

darin, durch die Gegenüberstellung musikalischer Werke Rekonstruktionen und Ähnlichkeiten zu erforschen. Das Grundlagenkapitel liefert mit der Thematik der musikalischen Rekonstruktionen die Kernaussage der Geschichte.

Auf der Ebene der Visualisierung besteht das Ziel, die Geschichte für den Betrachter optisch greifbar zu machen. Als Werkzeuge dienen dazu gestalterische Mittel der informationstechnischen Rahmenstrukturen. Die von Spotify definierten Audioeigenschaften aus Abschnitt 2.1 bilden auf der Ebene der Daten den Ausgangspunkt für die Datenmodellierung. Dabei gilt es, die einzelnen Kategorien und die zugehörigen Wertepaare zu klassifizieren und sinnvoll für den Betrachter bereitzulegen. Die Aufgabe der visuellen Komponenten der Anwendung besteht darin, den Sachverhalt der musikalischen Rekonstruktion zu kommunizieren. Entwicklertools bieten dazu zahlreiche gestalterische Mittel bei der Umsetzung einer interaktiven Visualisierung. Die konkrete Auswahl der gestalterischen Mittel wird in den folgenden Kapiteln dieser Arbeit erforscht und begründet.

Die Absicht der drei genannten Aspekte Geschichte, Daten und Gestaltung in Abhängigkeit zueinander ist mit der eigenen Vorgehensweise wie folgt zu betrachten: Der Betrachter erfährt durch die Visualisierung eine Veränderung. Die Visualisierung führt zu neuen Erkenntnissen, da der Sachverhalt der musikalischen Rekonstruktion aus einer anderen Perspektive erfasst wird. Die Stakeholder der interaktiven Datenvisualisierung sind Musikinteressierte und Medienschaffende, die ein Lösungsansatz bei der Recherche verwandter Musiktitel suchen.

Das Kapitel bietet eine Grundlage für die Ausarbeitung der Konzeptionierung. Durch die Gegenüberstellung der recherchierten Methodik von Data-Storytelling konnten neue Erkenntnisse erworben werden. Um die Umsetzung der Visualisierung aus anderen Blickwinkeln zu betrachten, erfolgt im nächsten Kapitel der Vergleich verwandter Arbeiten unter gestalterischen Gesichtspunkten. Diese Referenzen dienen als Inspirationsquelle und liefern innovative Ansätze.

2.4 Datenvisualisierung mit einem computergestützten System

Interaktive und innovative Datenvisualisierungen können mit zahlreichen Frameworks und Bibliotheken erstellt werden. Die benötigten Programmierkenntnisse hängen von der gewählten Umgebung ab. In Visualisierungstools wie Spotfire oder Data-Wrapper lassen sich die Datensätze über eine grafische Benutzeroberfläche in das Programm laden. Für die visuelle Aufbereitung stehen dem Autor in dieser Software verschiedene Möglichkeiten zur Verfügung. Dazu bieten die Programme von der Wahl eines Diagrammtyps bis hin zur gestalterischen Ausarbeitung viele Aspekte. Für komplexere und spezifische Anforderungen sind programmierorientierte Werkzeuge notwendig. Tableau und Google Charts bieten Hybridlösungen an, in denen die jeweiligen Software-Features mit den eigenen programmierten Skripten über eine Benutzeroberfläche verknüpft werden. Die Programmierung übernimmt in diesen Szenarien die Aufgabe der Datenaufbereitung und des Implementierungsprozesses komplexerer Problemlösungen. Mit Bibliotheken wie D3.js und Dygraphs werden Visualisierungen erzeugt, die ohne eine grafische Schnittstelle auskommen. Die Bibliotheken liefern zahlreiche Methoden, in denen die aufbereiteten Daten in einem Canvas gezeichnet werden. In dem Methodenaufruf definieren sich die gestalterischen Komponenten wie Farbe, Skalierung und weitere Einstellungen. Durch die individuelle Implementierung der Variablen und Funktionen in der Prozedur ist der Autor nicht auf die gegebenen Einstellungen der Benutzeroberfläche limitiert. Die Umgebungen enthalten demnach eine Vielzahl von Features, um einzelne Diagrammvorlagen und die gestalterische Aufbereitung nach den individuellen Bedürfnissen anzupassen (Vgl. Nair 2016: 3-5).

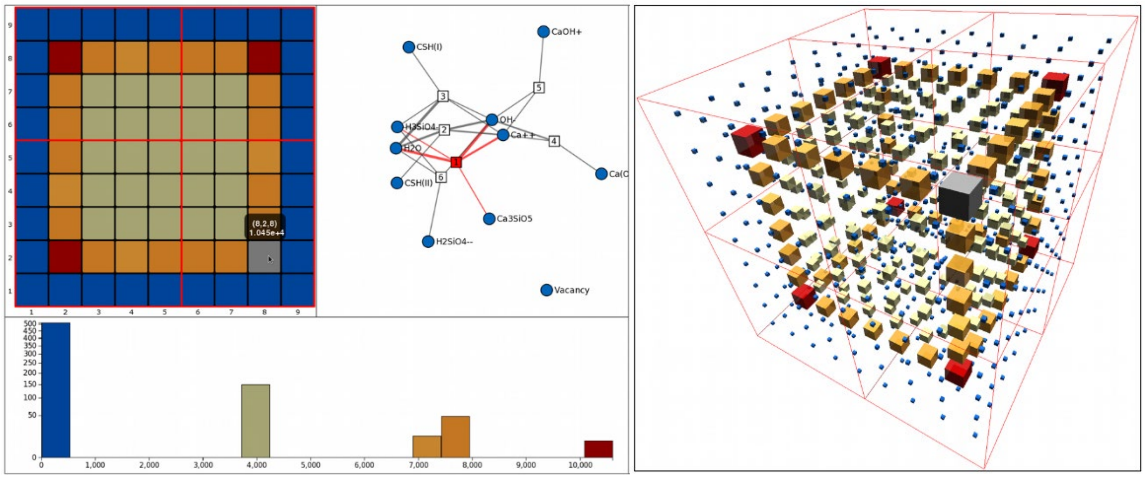


Abbildung 2 zwei- und dreidimensionale Datenvisualisierung (Griffen 2015: 188)

Welche Umgebung für die jeweilige Visualisierung geeignet ist, richtet sich nach den Anforderungen des Sachverhalts. Benötigt das Datenmapping eine dreidimensionale Perspektive, sind viele der genannten Visualisierungstools nur bedingt verwendbar. Die Umgebung IRIS liefert mit D3.js eine Erweiterung der dreidimensionalen Datendarstellung. In diesem Programm werden die zweidimensionalen Informationen über einen Webserver mit Node.js aufgegriffen und in einer dreidimensionalen Szene umgewandelt (S. Abb. 2). Ziel dieser Anwendung ist es, einen Sachverhalt mit mehreren Ebenen in einer grafischen Darstellung zusammenzufassen. Dies ermöglicht eine schnelle Gegenüberstellung einer zwei- und dreidimensionalen Darstellung (Vgl. Griffen 2015: 187).

In Abbildung 2 sind die Phasen bei der Zersetzung von Zementpasten und Beton in mehreren Diagrammen dargestellt. Diese Darstellung wird mit D3.js erzeugt und illustriert den vorliegenden Sachverhalt aus der zweidimensionalen Perspektive. In der von IRIS erzeugten dreidimensionalen Visualisierung (s. Abb. 2) sind alle Diagramme in einer Darstellung zusammengefasst. Aus dieser Gegenüberstellung lässt sich folgende Erkenntnis ableiten: Mit einem Visualisierungstool, welches eine dreidimensionale Szene unterstützt, lassen sich mehrere Ebenen in einer Grafik einbetten. Eine räumliche Darstellung unterstützt zudem die Sichtweise aus mehreren Perspektiven, jedoch verfälschen dreidimensionale Darstellung die Vergleichbarkeit der Datenpunkte (Vgl. Griffen 2015: 187-188).

2.4.1 Webspezifische und native Datenvisualisierung

Eine interaktive Visualisierung ermöglicht einen erweiterten und direkten Zugang zum Sachverhalt. Durch das Einwirken eines Akteurs auf ein computergestütztes System entsteht ein Austausch zwischen dem Betrachter und der Visualisierung. Im Bereich der Mensch-Computer-Interaktion wird diese Handlung als Schnittstelle zwischen Anwender und Computer bezeichnet. Eingaben in diese Systeme erfolgen üblicherweise über Peripheriegeräte wie Maus und Tastatur. Entwicklungsumgebungen wie D3.js und Unity unterstützen viele Interaktionsmöglichkeiten. Zugleich können die Vorteile einer dreidimensionalen Visualisierung genutzt werden. D3.js richtet sich dabei auf die Entwicklung webspezifischer Anwendungen und kann daher auf jedem webfähigen System ausgeführt werden. Unity bietet die Möglichkeit, sowohl auf native als auch webspezifische Systeme zuzugreifen. Bei der Entwicklung mit nativen Betriebssystemen

wie *Windows, Android, iOS* kann auf die plattformspezifischen Funktionen zurückgegriffen werden. Um webspezifische Anwendungen mit der Game-Engine zu kompilieren, nutzt *Unity* die Programmierschnittstelle *WebGL*. Mit dieser Bibliothek lassen sich dreidimensionale Grafiken im Webbrowser hardwarebeschleunigt wiedergeben (Vgl. Gu 2018: 1105-1106).

Die Umgebungen bieten eine gute Voraussetzung für die Implementierung und Umsetzung der zu erstellenden Datenvisualisierung. Ein wichtiger Faktor bei der Wahl einer geeigneten Umgebung ist die Programmoptimierung. Weist eine interaktive Visualisierung mit dynamischen Datensätzen eine hohe Latenz auf, führt dies zur Störung des Nutzererlebnisses. Dies äußert sich beispielweise in einer langen Ladezeit der Datensätze oder ruckeligen Bildwiedergabe. Zudem unterbricht irreführendes UI-Design den gewünschten Effekt der Motivation (Vgl. Pasch 2009: 3).

(a) Unity (b) D3.js

Abbildung 3 Datenvisualisierung mit Unity und D3.js (Kromer 2016: 2)

Um die Programmoptimierung für das künftige Projekt zu berücksichtigen, liefert der Performancetest von *Kromer* für die Unity-Game-Engine und D3.js-Bibilothek folgende Resultate: Die Tests wurden auf mobilen Endgeräten durchgeführt. Diese Systeme weisen eine hohe gesellschaftliche Relevanz auf, besitzen jedoch eine geringere Rechenleistung als Desktopanwendungen. Unter den getesteten Faktoren zählen die CPU-Auslastung, FPS-Leistung und Lesegeschwindigkeit von CSV-Dateien. Beim Versuchsaufbau wurden die Endgeräte Nexus 9, Galaxy S6 E, iPhone 6s Plus und iPad Air 2 fünf Minuten vor Testbeginn in den Ruhestand versetzt. Mit dem Ergebnis der CPU-Auslastung benötigt Unity 22 % weniger Rechenleistung als D3.js mit 38 %. Als Folge einer hohen CPU-Auslastung produzieren die Endgeräte viel Wärme und benötigen mehr Strom, was zu einer niedrigen Akkulaufzeit führt (Vgl. Kromer 2016: 1-5).

Bei der Wiederholrate der Bildfrequenz überzeugt Unity mit einer durchschnittlichen Rate von 57 Bildern pro Sekunde. D3.js zeigt hingegen eine Wiederholrate von 51 Bildern pro Sekunde auf. Die Wiederholrate eines computergestützten Systems beschreibt die Abfolge der gerenderten Bilder und ist ein wichtiger Faktor bei der Wiedergabe von Animationen. Um eine flüssige Bewegung sicherzustellen, sollte eine Wiedergaberate von unter 16 Bildern pro Sekunde nicht unterschritten werden. Eine hohe Rate führt wiederum zu einer detailgenauen Bildwiedergabe und gilt somit als Qualitätsmerkmal einer interaktiven Anwendung. Die Lesegeschwindigkeit einer CSV-Datei liegt bei D3.js durchschnittlich bei 5,17 Millisekunden. Bei der gleichen CSV-Datei benötigt Unity 15,17 Millisekunden. Unter diesen Bedingungen performt die Bibliothek D3.js besser. Eine geringe Lesegeschwindigkeit führt zu einer längeren Ladezeit. Wie zu Beginn erläutert, resultiert aus einer langen Ladezeit eine Störung des immersiven Erlebens. Die gemessenen Wertebereiche der Umgebungen sind im Verhältnis zu gering, um von einem Störfaktor auszugehen. Daraus lässt sich schließen, dass Unity im Vergleich zu D3.js aus Sichtweise der Performance bessere Ergebnisse liefert. Die Performancetests beider Entwicklungsumgebungen sind jedoch gut genug, um den immersiven Fluss des Anwenders nicht zu stören (Vgl. Kromer 2016: 1-5). Im folgenden Kapitel werden die Gestaltungsmöglichkeiten und Render-Features der Unity-Engine in Bezug einer Datenvisualisierung vorgestellt.

Nach der Gegenüberstellung der beiden Frameworks wurde für den Rahmen dieser Arbeit die Unity-Engine ausgewählt. Die höhere Bildfrequenz und geringere CPU-Auslastung der Engine ist für ein störungsfreies Nutzererlebnisses überzeugend. Zudem beschränkt sich das Framework nicht auf eine Plattform und ist daher universell einsetzbar. Obwohl webbasierte Anwendungen in der Regel einfacher zugänglich sind als native Anwendungen, bieten native Anwendungen aufgrund ihrer engen Integration mit dem Betriebssystem des Geräts, auf dem sie ausgeführt werden, oft eine bessere Benutzererfahrung. Durch die Verwendung von Systemfunktionen und höhere Rechenleistung können native Anwendungen auch komplexere Funktionen und Visualisierungen bereitstellen, die eine beeindruckendere und immersivere Erfahrung bieten. Letztendlich hängt die Wahl zwischen webbasierten und nativen Anwendungen von den spezifischen Anforderungen der jeweiligen Anwendung ab und davon, welche Art von Benutzererfahrung erreicht werden soll.

2.4.2 Datenvisualisierung in der Unity-Engine

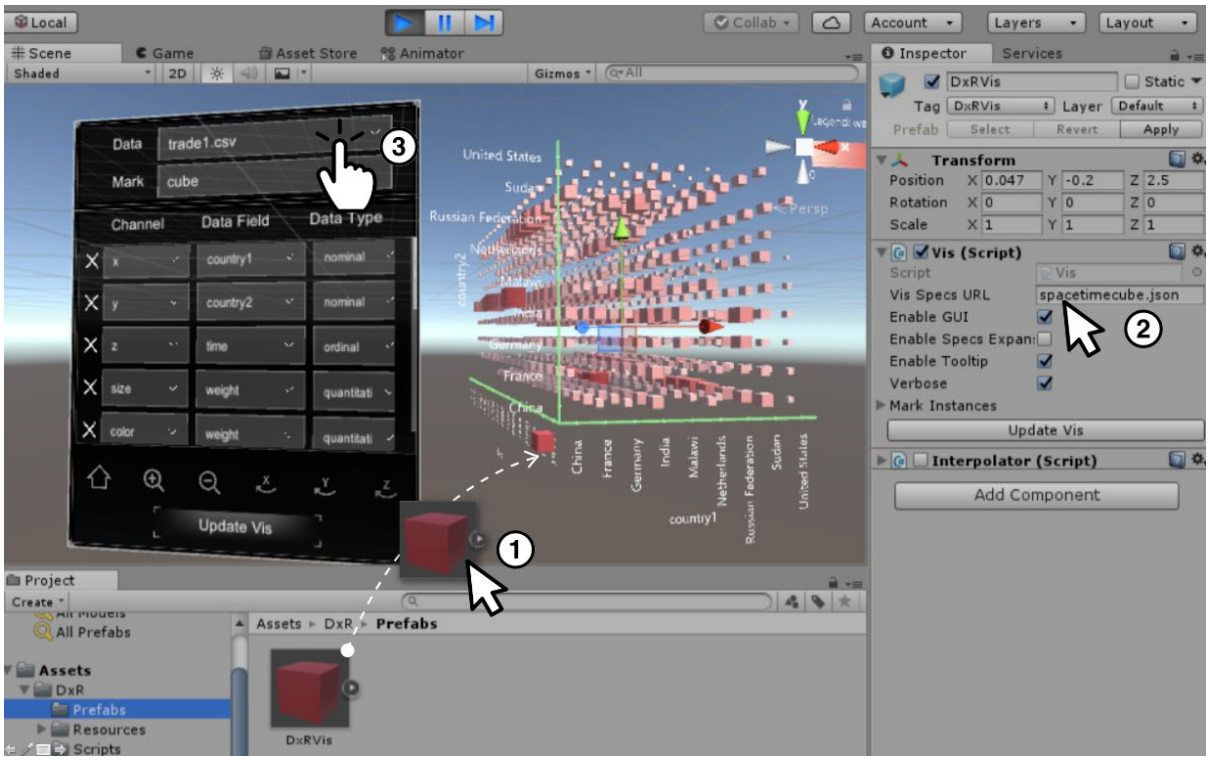


Abbildung 4 Ausschnitt des DXR-Toolkit in Unity (Sicat 2018: 3)

Unity ermöglicht umfangreiche Gestaltungsmöglichkeiten für zwei- und dreidimensionale Visualisierungen. Neben gängigen Diagrammtypen wie dem Säulen- und Balkendiagramm, lassen sich zweidimensionale Diagrammtypen in die räumliche Perspektive konvertieren. Dies erleichtert den Wechsel zwischen einer 2D- und 3D-Scene (Vgl. Sicat 2018: 1).

Abbildung 4 zeigt ein *Scatterplot*, welches mit dem DXR-Toolkit in der 3D-Scene von Unity erstellt wurde. Die dreidimensionale Szene ermöglicht eine erweiterte Perspektive, die mit einem zweidimensionalen Koordinatensystem nicht realisierbar ist. Vorgefertigte Unity-Scripte liefern eine einfache Umsetzung einer Datenrepräsentation. Wie in Abbildung 4 zu erkennen, können die von DXR bereitgestellten Prefabs in die Unity-Scene reingezogen werden. Anschließen liefert das erstellte *GameObject* konfigurierbare Diagrammtypen. Zu den Diagrammtypen werden passende Benutzeroberflächen mitgeliefert, mit denen der Nutzer in der Lage ist, die Diagramme während der Programmausführung zu modellieren (Vgl. Sicat 2018: 1-2). Die Engine bietet zudem eine einfache Integration eines VR- oder AR-Headsets, welches die Projektion zu einer immersiven Datenrepräsentation erweitert. Die Immersion beschreibt das Eintauchen in ein Medium. Im Effekt blendet der Betrachter die reale Umgebung aus und ist mit allen

Sinnen auf das Medium fokussiert. Wie stark die Immersion beim jeweiligen Betrachter eintritt, wird durch verschiedene Forschungsmethoden in Präsenz gemessen. Sie hängt von den individuellen Eigenschaften ab und wird durch stark stimulierende Projektionen wie beispielsweise einem *Head-Mounted Display* begünstigt (Vgl. Pasch 2009: 1-2).

Je nach Ausgangsformat bietet die Engine zahlreiche Interaktionsmöglichkeiten, in denen diverse Steuerelemente und Peripheriegeräte zum Einsatz kommen. Neben Eingabegeräten wie Maus und Tastatur bei Desktop-Anwendungen, werden bei AR-Systemen beispielsweise durch das Gestikulieren der Hand Events ausgelöst. Der Entwickler ist in der Lage, auf diese Events zurückzugreifen und diese in seine Programmabfolge einfließen zu lassen. In VR-Systemen sind die Steuerelemente überwiegend zwei Hand-Controller, welche sowohl die Bewegungen der Controller als auch weitere Eingaben über die Taster registrieren können. Diese weitreichenden Technologien erweitern die Repräsentation und Interaktion der Visualisierung, die mit zunehmender Komplexität fortgeschrittene Programmierkenntnisse erfordern. Bei der Abfrage dynamischer Datensätze ist zu berücksichtigen, dass die Daten im Wertebereich der jeweiligen Kategorie liegen, um keine unerwünschten Effekte in der Visualisierung auszulösen. Die Interaktionsmöglichkeiten der Engine liefern unterschiedliche Ansätze, um die Datensätze dynamisch anzupassen (Vgl. Donalek 2014: 610-611).

Aus Sicht eines Daten-Wissenschaftlers gilt es zu prüfen, ob eine dreidimensionale Darstellung sowie eine hohe Präsenz der Datensätze beim jeweiligen Anwendungsfall sinnvoll ist. Im Vordergrund steht die Aussage des Autors und die Darstellung eines Sachverhalts. Mit vorgefertigten Lösungen wie dem DXR-Toolkit werden viele Parameter für die Datenvisualisierung mitgeliefert, die eine einfache Umsetzung in Unity ermöglichen. Größe, Farbgestaltung, Rotation und weitere Eigenschaften sind mit wenigen Skripten definierbar. Vorgefertigte Benutzeroberflächen bieten während der Programmausführung den Zugriff auf die direkte Manipulation und Gestaltung der jeweiligen Visualisierung. Auf welche Weise der Betrachter mit der Benutzeroberfläche interagiert, hängt vom jeweiligen Ausgabegerät und der Absicht des Autors ab (Vgl. Sicat 2018: 9).

Die Engine bietet ein umfangreiches Werkzeug, mit dem die Einbindung neuster Technologien und gestalterische Freiheiten möglich sind. Somit sind sowohl konventionelle als auch innovative Umsetzungen einer Datenrepräsentation möglich. Aus diesen Erkenntnissen konnten Rückschlüsse auf das Spektrum der Gestaltungsmöglichkeiten in der Unity-Engine gezogen werden. Die Engine beschränkt sich dabei nicht auf die Erstellung zweidimensionaler Darstellungen, sondern bietet weitreichende Hilfsmittel für ein immersives Erlebnis (Vgl. Sicat 2018: 9).

2.5 Zusammenfassung

In diesem Kapitel wurden die Grundlagen für die thematische Auseinandersetzung dieser Masterthesis festgelegt. Zu Beginn wurde die Definition der Eigenschaften digitaler Audio-Inhalte untersucht. Hierfür wurde zunächst der geschichtliche und marktwirtschaftliche Aspekt der Analyse und die Klassifizierung von Audio-Inhalten erläutert und zudem die Vor- und Nachteile personalisierter Musikempfehlungen benannt. Dazu liefern die Audio-Features der Spotify-API einen Ansatz für die Suchanfrage korrelierender Musiktitel.

Im zweiten Abschnitt dieses Kapitels wurden die drei Formen der musikalischen Rekonstruktion genannt. Die erste Form ist die Reproduktion musikalischer Werke und gliedert sich in den Rubriken *Sampling, Remix* und *Mashup*. Die zweite Form ist die Interpretation eines Musikstücks und wird als *Coversong* bezeichnet. Die Definition einer *Coverversion* verschwimmt mit der Grenze der Imitation und Interpretation. Diese Form hängt von der individuellen kreativen Mitwirkung des Interpreten zusammen. Die letzte Form musikalischer Rekonstruktionen ist die *Interpolation* und definiert sich durch das Umgestalten oder Verfälschen eines musikalischen Werkes. Aus diesen Erkenntnissen wurde eine allgemeine Definition der musikalischen Rekonstruktion hergeleitet.

In dem Kapitel der Datenvisualisierung wurden die Grundlagen der Datenklassifizierung und Data-Storytelling beschrieben. Bei der Klassifizierung von Datensätzen gilt es zu beachten: Je mehr Variablen in eine Visualisierung einbezogen werden, desto komplexer wird sie. Daher ist eine geeignete Visualisierungstechnik notwendig, um große Datensätze zielgerichtet interpretieren zu können. Data Storytelling hat das Ziel, Sachinformationen prägnant und effektiv zu vermitteln. Durch geeignete

Visualisierungstechniken kann die Aufmerksamkeit des Betrachters auf das Wesentliche gelenkt werden, um eine klare Aussage zu vermitteln.

Das letzte Kapitel behandelt die Datenvisualisierung mit einem computergestützten System. Dazu liefern zahlreiche Frameworks und Bibliotheken die Möglichkeit eine interaktive Datenvisualisierung umzusetzen. Diese Hilfsmittel ermöglichen dem Autor die visuelle und technische Aufbereitung der Daten. Im Vergleich einer webspezifischen zu einer nativen Datenvisualisierung sind webbasierte Anwendungen in der Regel zugänglicher. Jedoch bieten native Anwendungen mit ihren systemspezifischen Funktionen und erhöhter Rechenleistung eine erweiterte Bandbreite an Gestaltungsmöglichkeiten. Die Unity-Engine mit ihren nativen Funktionen bietet daher die besten Voraussetzungen die Datenvisualisierung umzusetzen.

Im nächsten Kapitel werden korrelierende Visualisierungen zu der genannten Thematik dieser Arbeit herangezogen, um Anregungen für die eigene Datenvisualisierung zu finden. Dieses Kapitel dient zum grundlegenden Verständnis der musikalischen Rekonstruktionen und interaktiven Datenvisualisierung.

3. Visualisierung der Spotify Audio-Features

Die bisherigen Kapitel liefern die Basis für die thematische Auseinandersetzung der Audio-Features der Spotify-API. Zudem wurden die Grundlagen interaktiver Datenvisualisierungen anhand der Datenklassifizierung und die Schlüsselelemente des Data-Storytelling nähergebracht. In diesem Kapitel werden vergleichbare Referenzen hinzugezogen, um Ansätze für die eigene Visualisierung zu ermitteln. Die Referenzen werden unter den Gesichtspunkten des Datenmappings, Clustering und der Darstellung von Hierarchien untersucht und die daraus resultierenden Erkenntnisse zusammengefasst.

Das Datenmapping ist der erste Schritt bei der Wahl und Einordnung eines geeigneten Datenmodells. Im ersten Kapitel werden die Faktoren des Datenmappings anhand der Datenvisualisierung „Klangspektrum“ beleuchtet. Dazu werden die verschiedenen Gestaltungsmöglichkeiten und die Vorgehensweise der Datenaufbereitung erläutert. Im darauffolgenden Kapitel wird das Rangieren von Datensätzen mit der Datenvisualisierung von *Music-Circles* beschrieben. In dieser Methodik wird untersucht, wie die Kennzahlen in Relation zueinander gebracht und nach einem spezifischen Muster rangiert werden. Das letzte Kapitel behandelt die Mustererkennung in Abhängigkeit zum Ausgangsmaterial. Diese Methodik wird mittels der Datenvisualisierung *MusicViz* beschrieben. In diesem Ansatz werden die Datentypen in Bezug zueinander gebracht und die kategorischen Gemeinsamkeiten zusammengefasst. Die genannten Referenzen bilden den Ausgangspunkt für die eigene Konzeptionierung. In Abschnitt 4.4 werden alle Erkenntnisse zusammengetragen und mit der eigenen Visualisierung gegenübergestellt. In dieser Gegenüberstellung wird die Aufgabe und Abgrenzung der Visualisierung definiert.

3.1 Datenmapping

Das Projekt *Klangspektrum* von Michael Schwarz wurde 2017 veröffentlicht und bietet Spotify-Nutzern eine Datenvisualisierung zur individuellen Analyse der eigenen Musikgewohnheiten. Mit Hilfe der bereitgestellten Musikeigenschaften der Spotify-API wird der persönliche Musikverlauf analysiert und grafisch aufbereitet. Zu dieser Visualisierung hat Schwarz seine Argumentationen und Vorgehensweisen in seiner Bachelor-Thesis festgehalten. Im folgenden Abschnitt wird diese Dokumentation mit der Visualisierung untersucht und der eigenen Vorgehensweise gegenübergestellt. Die Idee

der Anwendung ist es, den Hintergrundprozess des Spotify-Algorithmus für den Nutzer transparenter zu gestalten. Die Visualisierung ermöglicht, komplexe Vorgänge und Verknüpfungen der Spotify-Nutzeranalyse in einer grafischen Darstellung zusammenzufassen. Die Zielgruppe der Anwendung richtet sich an Spotify-Nutzer, die einen Einblick in ihre eigenen Musik-Tendenzen erlangen wollen (Vgl. Schwarz 2017: 7-9).

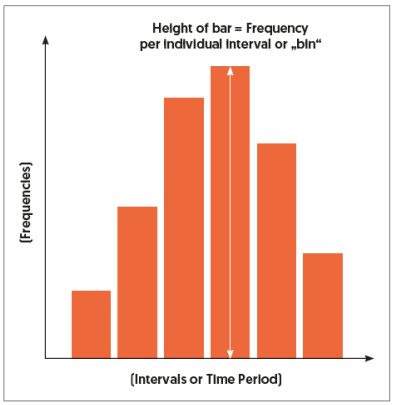


Abbildung 5 Histogramm (Ribecca (o. J.))

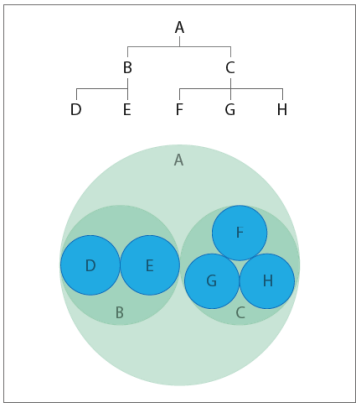


Abbildung 6 Treemap (Ribecca (o. J.))

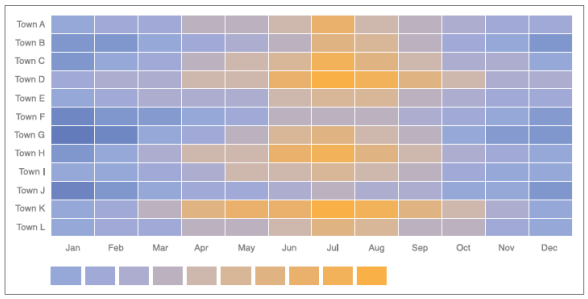


Abbildung 7 Heatmap (Ribecca (o. J.))

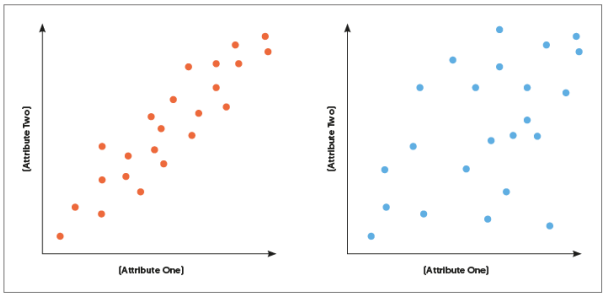


Abbildung 8 Streudiagramm (Ribecca (o. J.))

Schwarz bezieht sich bei der Umsetzung auf typische Diagrammtypen. Dazu nennt der Autor vier Arten von Diagrammen, die seiner Visualisierung am nächsten kommen. Im folgendem werden diese Diagrammtypen kurz beschrieben. Das *Histogramm* bietet die Möglichkeit, eine Häufigkeitsverteilung in Relation einer zugeteilten Klasse zu veranschaulichen (s. Abb. 5). Zweck dieser Darstellung ist, eine Aussage über die Gesamtverteilung zu treffen. Das *Heatmap*-Diagramm veranschaulicht auf intuitive Weise große Datenmengen (s. Abb. 7). Mit ihr lassen sich beispielsweise markante Werte in zeitlichen Abständen erfassen. *Treemap*-Diagramme dienen zur Demonstration hierarchischer Schemata (s. Abb. 6). In dieser Form werden die Daten nach ihrer Zugehörigkeit und Hierarchie definiert. Diese Darstellung wird überwiegend als rechteckige oder kreisrunde Form abgebildet. Ein *Streudiagramm* präsentiert eine statistische Perspektive der Datensätze (s. Abb. 8). Die Wertepaare werden in einem kartesischen Koordinatensystem eingetragen und ergeben in der Gesamtverteilung eine

lineare Struktur. Auf Grundlage dieser Diagrammtypen untermauert Schwarz seine Argumentation für die designtechnische Konzeptionierung der Visualisierung „Klangspektrum“.

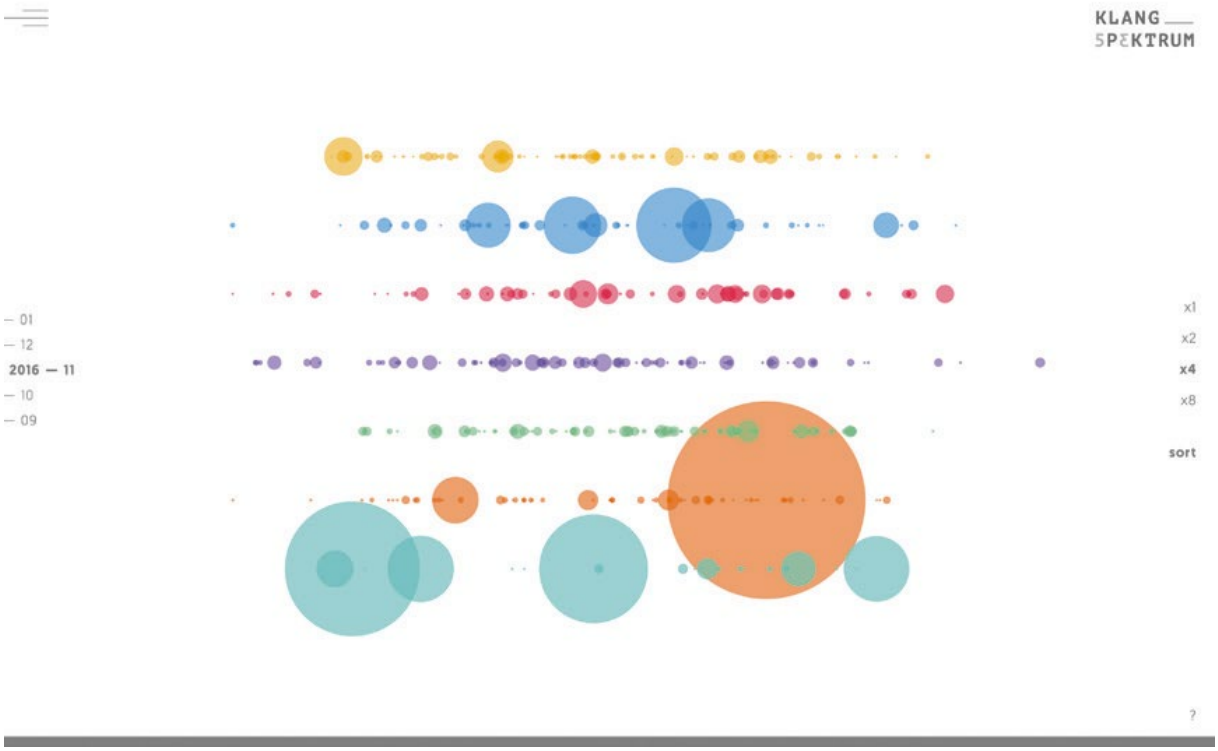


Abbildung 9 Visualisierung Klangspektrum (Schwarz 2017: 36)

Abbildung 9 zeigt eine Momentaufnahme der zeitlich modulierten Visualisierung von Schwarz. In dieser Darstellung sind die Musikeigenschaften der Spotify-API in Abhängigkeit des zeitlichen Verlaufs dargestellt. Basierend auf der Auswertung gehörter Musiktitel, bildet die Visualisierung durch die Animation der Wertepaare einen dynamischen Zeitstahl. Die zeitlichen Intervalle sind auf der linken Seite durch den angegebenen Monat wählbar. Auf der Y-Achse sind die Audio-Eigenschaften in kategorisierte Strecken eingeteilt und unterscheiden sich farblich voneinander. Die Wertepaare der Audio-Eigenschaften sind in ihren Wertebereichen von 0 bis 100 eingeteilt. Wie stark die Audio-Eigenschaften der gehörten Musiktitel zu einem bestimmten Zeitpunkt ausgeprägt sind, misst sich an der Größe der Kreise. Zu dieser Visualisierung wird dem Nutzer eine Legende mitgeliefert, in dem die einzelnen Eigenschaften erklärt werden (s. Abb. 10). Auf der rechten Seite der Anwendung liefern die jeweiligen Schaltflächen eine individuelle Skalierung der Visualisierung. Je nach Skalierung resultiert eine detaillierte oder umfassende Gesamtdarstellung bei der Darstellung der einzelnen Kreise.

Abbildung 10 Legende Audio-Features (Schwarz 2017: 38)

Aus den zuvor beschriebenen Diagrammen und der Visualisierung von Schwarz lassen sich folgende Gemeinsamkeiten herleiten: Zum einen nutzt Schwarz das Konzept des *Treemap*-Diagramms zur Darstellung hierarchischer Strukturen. Entscheidend für eine gestaffelte Zuordnung sind die Größenverhältnisse der Wertepunkte. Die Visualisierung orientiert sich bei der Erfassung größerer Datenmengen an die *Heatmap*. Mit dem Unterschied zur *Heatmap* verwendet Schwarz keine farbliche Kennzeichnung zur Einordnung der Werte, sondern zur Trennung der einzelnen Kategorien.

Bei der Vorgehensweise eines Streudiagramms übernimmt der Autor die ordinalskalierten Merkmale, um statistische Zusammenhänge darzustellen. Durch die Größenordnung der gestaffelten Wertepaare in der Visualisierung ergeben sich überlappende Punktewolken. Aus diesen Schemata resultieren lineare Strukturen, die vom Betrachter erfasst werden. In der Gegenüberstellung zum Histogramm übernimmt Schwarz das Prinzip der Häufigkeitsverteilung, indem er die einzelnen Kategorien in einer jeweiligen Strecke darstellt. Durch die Größenverhältnisse der Kreise zueinander ergeben sich wechselseitige Zusammenhänge, in denen die Verteilung der klassifizierten Daten konkretisiert werden. Im Unterschied zum Histogramm nutzt Schwarz für die zeitliche Unterteilung das Prinzip der Bewegung und Animation. Mit diesem Schema verknüpft Schwarz die Historie auf einer weiteren Ebene. Aus diesen Erkenntnissen lässt sich schlussfolgern, wie Schwarz das Konzept erprobter Diagramme in seine eigene Visualisierung einbindet.

Die Vorgehensweise von Schwarz bietet innovative Ansätze für das Datenmapping der eigenen Visualisierung. Zudem können wichtige Aspekte der Visualisierungstechniken übernommen werden. Die Visualisierung von Schwarz zeigt auf kreative und informative Weise, wie mehrere Dimensionen in einer Darstellung untergebracht werden. Die Besonderheit der Visualisierung besteht in der zeitlichen Repräsentation der Datensätze, welche in Form der Animation und Bewegung ausgedrückt wird. Da diese Visualisierung in eine Anwendung eingebettet ist, können die Möglichkeiten eines computergestützten Systems für eine dynamische Berechnung der Daten genutzt werden. Daraus ergeben sich viele Interaktions- und Animationsmöglichkeiten, die in die Visualisierung miteinfließen. Ein weiterer Vorteil einer virtuellen Visualisierung ist die individuelle Auswertung der Datensätze, die zur Generierung der Visualisierung dynamisch umgestaltet werden.

3.2 Darstellung von Rangfolgen

Das Bilden einer Rangfolge von Objekten in einer hierarchischen Struktur ermöglicht die Vergleichbarkeit unterschiedlicher Größen. Die dabei angewandte Methodik besteht darin, die Kennzahlen in Relation zueinander zu setzen und nach einem beliebigen Muster zu rangieren. Die Bewertung umfangreicher Informationen wird somit durch die Rangfolge der Ordnungszahlen erkennbar. Die Vorgehensweise, Musik nach ihren Metadaten wie Album, Künstler und weiteren Informationen zu sortieren, erleichtert Nutzern das Erkunden neuer Musik. Mit dem Rangieren von Musiktiteln hinsichtlich der kategorischen Einordnung ist möglich, musikalische Werke nach ihren Eigenschaften zu vergleichen und abzuwägen. Eine geeignete Visualisierung unterstützt den Informationsfluss zwischen Nutzer und Datensatz. In der Musikindustrie erfüllt das Ranking populärer Musiktitel die Funktion eines Verkaufsarguments und einer Empfehlung (Vgl. Guedes 2017: 1).

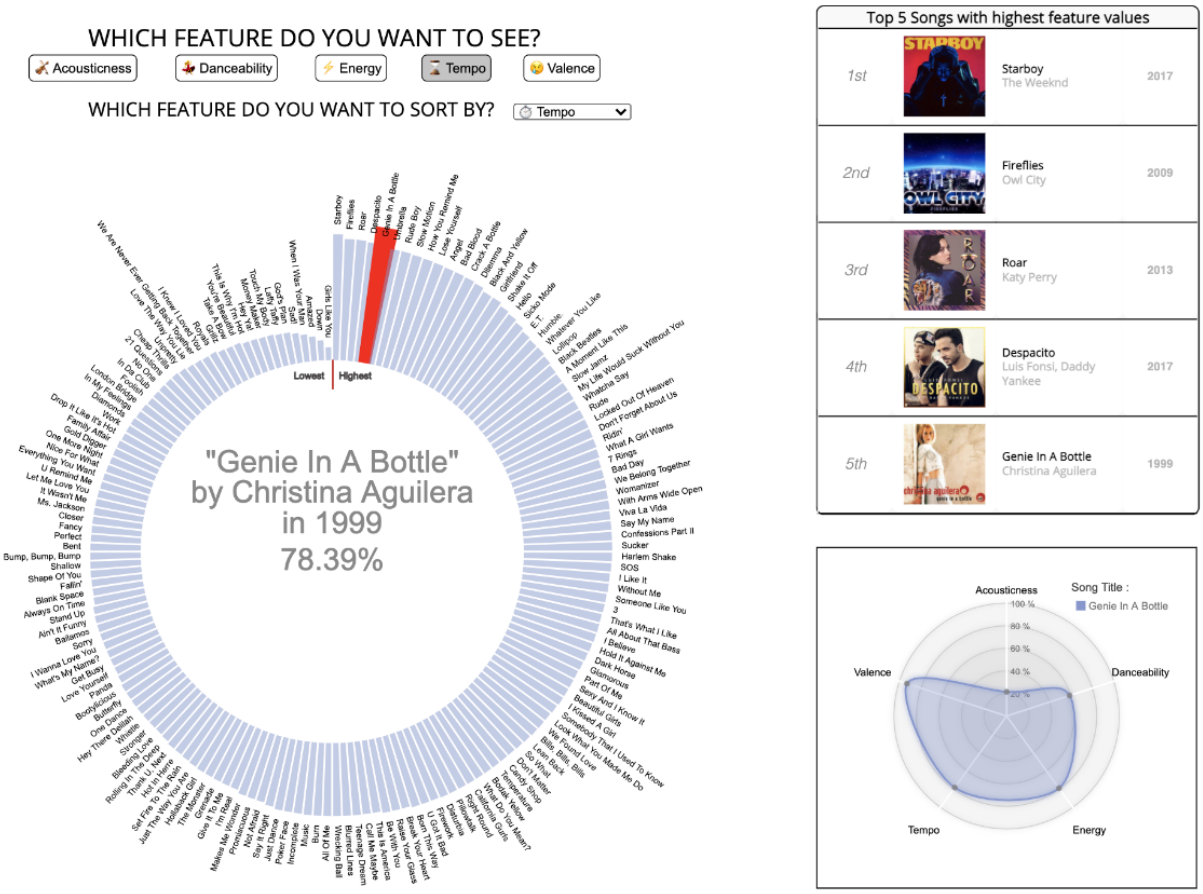


Abbildung 11 Music-Circle (Kim 2021: 3)

Um hierarchische Strukturen für den Nutzer nachvollziehbar zu machen, hat das Team von Music-Circles an einer interaktiven Visualisierung geforscht, welches die Ähnlichkeit populärer Songs veranschaulicht (s. Abb. 11). Als Ausgangsmaterial der Visualisierung dienen die Top 100 Billboard Charts im Zeitraum von 1999 bis 2019. Dazu werden die Audio-Features der einzelnen Songs verglichen und nach einem Ranking-System sortiert. Der Nutzer kann dabei einstellen, nach welcher Kategorie die Daten sortiert werden sollen. Somit ist der Nutzer in der Lage, die Trends der Audio-Features nach ihren präferierten Eigenschaften zu erforschen (Vgl. Kim 2021: 1).

Die Anwendung *Music-Circles* besteht aus zwei Diagrammen, mehreren Schaltflächen und einer Tabelle mit den fünf Songs mit den höchsten Werten in der jeweiligen Kategorie. Die Schaltflächen dienen als Interaktionsschnittstelle, in der die Auswahl der zu untersuchenden Kategorie und der Faktor der Sortierung festgelegt wird. Die kategorische Einteilung der Spotify Audio-Features ist ähnlich zur Visualisierung von Schwarz. Das größere Diagramm besteht aus einem *Circular-Barplot*, welches 168 Songs nach ihrem Ranking auflistet. Als weitere Interaktionsmöglichkeit ist der Nutzer in der

Lage, mit dem Mauszeiger über die gelisteten Songs zu fahren, um an detailliertere Informationen zu gelangen. Diese Informationen werden in die Mitte des Diagramms projiziert. Die Höhe der Säulen repräsentieren, wie stark der Song in der ausgewählten Kategorie ausgeprägt ist. Diese Anordnung ermöglicht eine einfache Gegenüberstellung aller Songs. Das *Radar-Chart* (s. Abb. 11) liefert eine grafische Darstellung aller Kategorien für den jeweiligen Song (Vgl. Kim 2021: 2-3).

Music-Circle liefert einen Ansatz, wie Audio-Features der Spotify-API dazu genutzt werden können, um ein Verständnis vergleichbarer Songs zu erlangen. Zudem bietet die Anwendung eine unterhaltende Funktion, um das Ranking von Audio-Inhalten zu erkunden. Ziel dieser Anwendung ist es, eine Veränderung beim Benutzer auszulösen, in dem das Bewusstsein für die Audio-Eigenschaften gestärkt wird (Vgl. Kim 2021: 4). Diese Anwendung übermittelt eine Grundlage für die eigene Vorgehensweise bei der Mustererkennung von Audio-Inhalten. Durch das Ordnen hierarchischer Strukturen wird ein Verfahren angewendet, welches die Ähnlichkeiten nach Ihren Größen hin untersucht. Durch die Untersuchung klassifizierter Datensätze lässt sich eine große Anzahl an Musiktiteln vergleichen. Durch das Mapping der Daten auf ein *Circular-Barplot* ist ein umfangreicher Überblick auf einer kleinen Fläche möglich. Im Gegensatz zu einer zeilenorientierten Tabelle liefert die kreisförmige Anordnung eine platzsparende Lösung. Das ergänzende Diagramm bietet zudem eine detailliertere Informationsquelle bei der Untersuchung einzelner Songs.

3.3 Clustering

Das Clustern von Daten beschreibt das Verfahren bei der Aufbereitung und Analyse von strukturellen Ähnlichkeiten. Das Ziel dieser Methodik besteht darin, Zusammenhänge aus Datenbeständen zu erfassen und zu klassifizieren. Die Aufgabe in der Informatik liegt in der Entwicklung einer Handlungsvorschrift, die eine Musterkennung in Abhängigkeit zum Ausgangsmaterial ermöglicht. Anwendungsbeispiele sind der Segmentierungsprozess in einem bestimmten Markt oder die multimediale Verarbeitung von Bildmaterial (Vgl. Myatt 2009: 67).

Die Relevanz von Clustering ist in der derzeitigen Musikindustrie bei der Vermarktung und Vertreibung von Musik ein wichtiger Aspekt. Die Möglichkeit Musikdaten zu analysieren und aufzubereiten, liefert Streaming-Plattformen einen Wettbewerbsvorteil. Eine anwenderfreundliche Nutzung wird durch die Optimierung des Navigationsmechanismus begünstigt. Realisiert wird dieser Vorgang durch die Generierung personalisierter Musikempfehlungen und Anzeige kohärenter Wiedergabelisten. Dieser Prozess steigert die Bedeutsamkeit des Streamingdienstes im Marktsegment (Vgl. Kim 2021: 1).

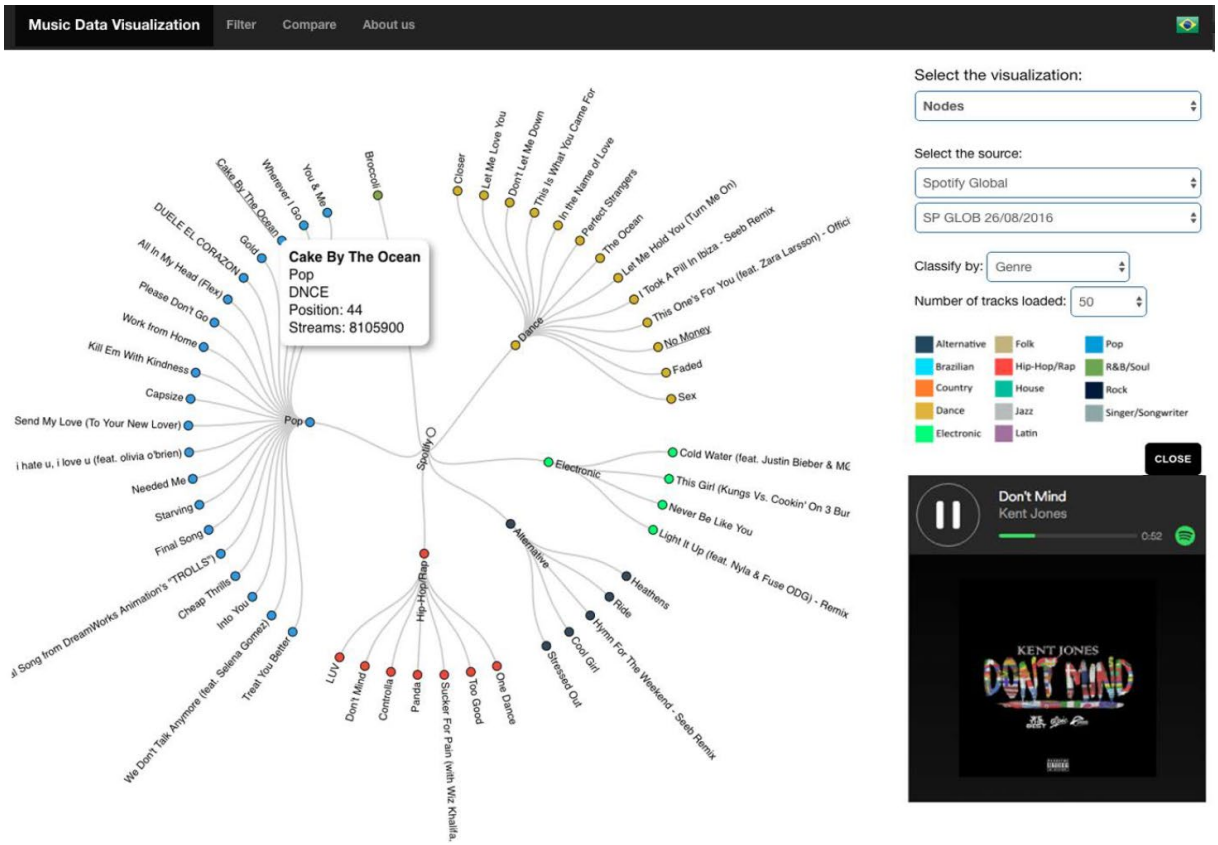


Abbildung 12 Music Rankings (Guedes 2017: 3)

Die Visualisierung *MusicViz* veranschaulicht die Spotify- und Billboard-Charts in Abhängigkeit verschiedener Zeiträume und Klassifizierungen. Die Anzahl der gelisteten Titel und die Kategorisierung sind in der interaktiven Visualisierung beliebig auswählbar. Der Nutzer ist in der Lage, die gleichen Datensätze und Klassifizierungen in vier verschiedenen Diagrammtypen darzustellen. Der Diagrammtyp *Nodes-Link Tree* liefert einen knoten-basierten Graphen, welcher ausgehend von der gewählten Klassifizierungsart die Musiktitel kategorisch in Unterknoten zusammenfasst (s. Abb. 12).

Das Ranking findet in diesem Szenario im Uhrzeigersinn im jeweiligen Unterknoten statt und ist nach der Anzahl des Streamings absteigend sortiert. Durch das Überschweben über einen einzelnen Musiktitel mittels des Mauszeigers lassen sich detaillierte Informationen anzeigen. Diese Funktion reduziert den Informationsgehalt der gesamten Visualisierung. Mit dem Spotify-Player lassen sich die einzelnen Songs wiedergeben. Dies wird mit einer URL-Anfrage ausgehend von der Spotify-API realisiert (Vgl. Guedes 2017: 4-5).

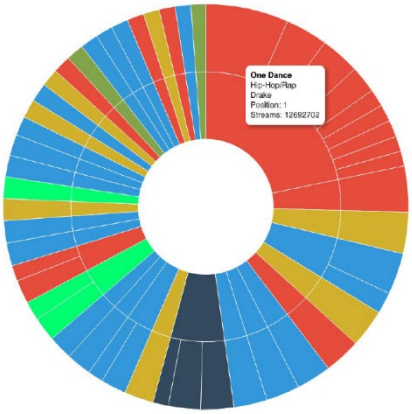


Abbildung 13 Donut Chart (Guedes 2017: 4)

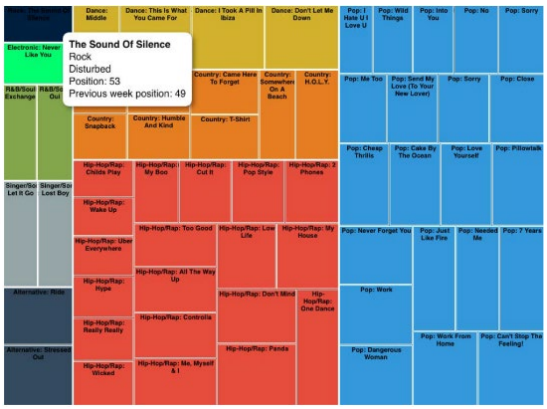


Abbildung 14 Treemap (Guedes 2017: 4)

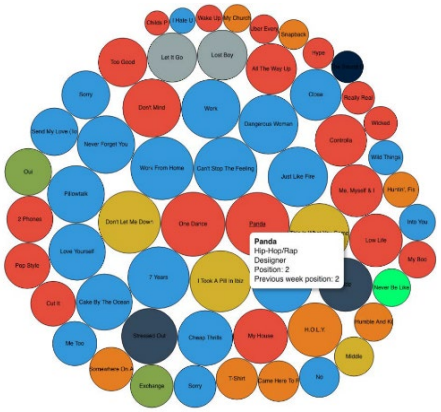


Abbildung 15 Bubble Chart (Guedes 2017: 4)

Das Donut-Diagramm zeigt eine kreisförmige Anordnung der Spotify Top 50 (s. Abb. 13). Aus dieser Auflistung geht hervor, welche Künstler mit welchem Anteil eine Platzierung in den Top 50 erlangt haben. In dieser Darstellung ist es aus platztechnischen Gründen nicht möglich, die einzelnen Teilabschnitte mit dem Titel zu etikettieren. Daher ist die Funktion mit der Maus an detaillierte Informationen zu gelangen, eine wichtige Eigenschaft. Das *Treemap*-Diagramm hingegen ist quadratisch angeordnet und listet alle Top 60 Billboard-Charts in Abhängigkeit des Musikgenre auf (s. Abb. 14). Die Größe eines Quadrats definiert die Positionierung in den Charts. Die vierte Variante repräsentiert die Top 60 der Billboard-Charts innerhalb einer Bubble Chart (s. Abb. 15). Geordnet nach der Position von innen nach außen, gibt die Größe der Blase ebenfalls die Platzierung an. Die farbliche Zuordnung der drei genannten Diagramme bezieht sich gleichermaßen auf die jeweiligen Künstler (Vgl. Guedes 2017: 3-5).

Folgende Gemeinsamkeiten und Gegensätze lassen sich zwischen den vier Varianten herleiten: Die Hierarchien der jeweiligen Musiktitel sind bis auf das *Node-Link Tree* durch die Größe definiert. Durch die Farbgestaltung der Datensätze findet eine Kategorisierung statt. Aus den Größenverhältnissen in der Gesamtverteilung wird deutlich, welchen Anteil die ausgewählte Kategorie einnimmt.

Die Anwendung *MusicViz* liefert einen Ansatz, um das Clustering von Musiktiteln zu repräsentieren. Die Klassifizierung der Daten wird mittels Farbgestaltung und Knotenpunkten realisiert. Diese Erkenntnisse liefern Motive für die Gliederung und Strukturierung für die eigene Visualisierung. Die Darstellung der hierarchischen Anordnung der Daten wird wie bei der Visualisierung von Schwarz auch durch Größenverhältnisse ausgedrückt. Interaktionsmechanismen wie das Hover-Event der Maus reduzieren den Informationsgehalt und ermöglichen das Einblenden detaillierter Informationen.

3.4 Zusammenfassung

Dieses Kapitel bildet die Grundlage für die Auseinandersetzung mit den Audio-Features der Spotify-API und eine Einführung für die interaktive Datenvisualisierung. Zudem bezieht sich das Kapitel auf die Suche nach Ansätzen für die eigene Visualisierung. Dies wird unter den Gesichtspunkten des Datenmappings, Clustering und Darstellung von Rangfolgen untersucht.

Die Visualisierung von Schwarz stellt innovative Ansätze für das Datenmapping bereit und vermittelt wichtige Aspekte der Visualisierungstechniken, die in eigene Visualisierungen integriert werden können. Die Stärke dieser Visualisierung liegt in der zeitlichen Darstellung der Datensätze durch Animation und Bewegung. Da die Visualisierung in eine Anwendung eingebettet ist, können die Vorteile eines computergestützten Systems für eine dynamische Berechnung der Daten genutzt werden.

Die Bildung einer Rangfolge von Objekten in einer hierarchischen Struktur ermöglicht den Vergleich unterschiedlicher Größen. Dabei werden Kennzahlen in Relation zueinander gesetzt und nach einem beliebigen Muster sortiert. Der Ansatz von Music-Circle nutzt die Audio-Features der Spotify-API, um vergleichbare Songs zu identifizieren und ein besseres Verständnis für diese zu erlangen.

Das Clustern von Daten liefert eine Analyse-Methode zur Erkennung von strukturellen Ähnlichkeiten in Datenbanken. Die Anwendung *MusicViz* bietet einen Ansatz, um das Clustering von Musiktiteln zu visualisieren. Die Daten werden dabei mit Farbgestaltung und Knotenpunkten klassifiziert.

Im nächsten Kapitel werden die konzeptionellen Ansätze für die Umsetzung der Datenvisualisierung diskutiert. In diesem Kapitel wurden nach Ansätzen für die eigene Visualisierung gesucht. In Kapitel 4.4 werden diese Ansätze mit den eigenen verglichen.

4. Die Konzeptionierung der Visualisierung

In den folgenden Kapiteln wird die konzeptionelle Ausarbeitung der Anwendung *Comperify* beschrieben. Dabei werden die Strukturen und Abläufe der Anwendung festgelegt sowie die Diagrammtypen für die Visualisierung begründet und die gestalterischen Aspekte der Anwendung erläutert. Diese konzeptionelle Ausarbeitung bildet die Grundlage für den Implementierungsprozess und definiert die Planung, die für die Umsetzung der Anwendung erforderlich ist.

Zu Beginn dieses Kapitels wird das Menükonzept erarbeitet, das die Ziele der Benutzererfahrung widerspiegelt. Als Ansatz wird das Trichterprinzip im journalistischen Sinn mit den eigenen Designprinzipien verglichen. Im darauffolgenden Abschnitt erfolgt die Beschreibung der geplanten Benutzeroberfläche anhand der erstellten Wireframes und Flowcharts. Dieser Teil der Konzeptionierung dient als Vorlage für die programmiertechnische Umsetzung und liefert einen Überblick über die einzelnen Funktionalitäten der Anwendung. Darauffolgend werden die verwendeten Diagrammtypen beschrieben und begründet, die für die Visualisierung der Datensätze von Spotify zum Einsatz kommen. Abschließend gilt es, die beschriebenen Referenzen aus Kapitel 3 mit der eigenen Visualisierung zu vergleichen.

4.1 Menüführung

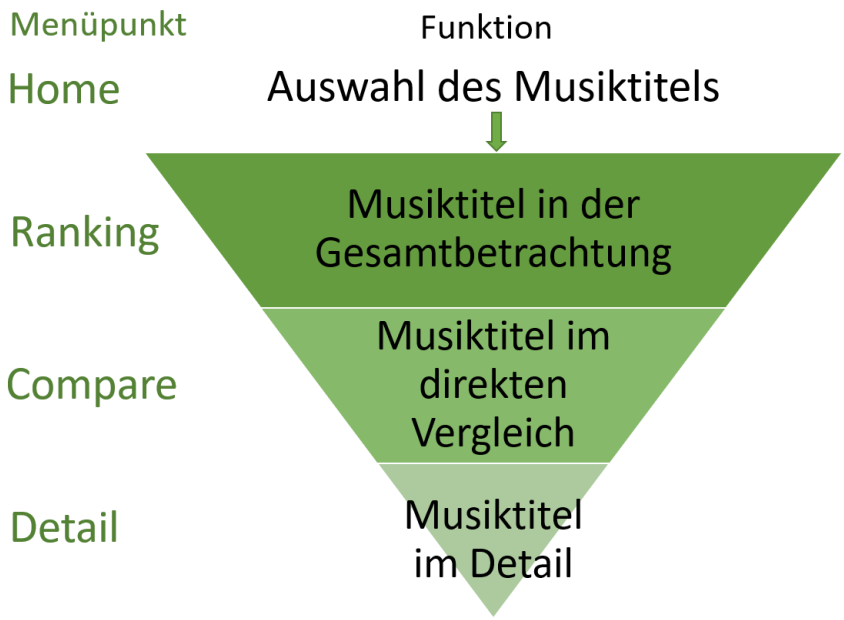


Abbildung 16 Trichterprinzip Comparify

Das Ziel der Benutzererfahrung besteht darin, den Nutzer von der Gesamtbetrachtung bis hin zu den Details der Songanalyse durchzuführen. Verglichen mit dem Ansatz des Trichterprinzips sollen die Eingaben des Nutzers vom Gesamtkontext bis hin zu einer Detailansicht gefiltert werden. Im journalistischen Kontext wird dieses Prinzip als Kommunikationsmodell verwendet und beschreibt den strukturellen Aufbau einer Nachricht. Dabei beginnt eine Nachricht mit einer Hauptaussage und folgt mit detaillierteren, meist sekundären Informationen (Vgl. Mast 2018: 345-346). Dieses Prinzip lässt sich auf das Designprinzip für Benutzeroberflächen übertragen. Durch die interaktive Informationsabfrage zu Beginn der Anwendung durchläuft der Nutzer eine individuelle Erkundung verwandter Musiktitel, bis hin zu einer detaillierten Veranschaulichung seiner Eingaben (Vgl. Joshi 2004: 1-3). Dieses Modell bietet einen konzeptionellen Ansatz für die Menüführung der Anwendung. Abbildung 16 veranschaulicht das angewendete Trichterprinzip für Comparify. Auf der linken Seite sind alle Menüpunkte aufgeführt, auf der rechten Seite die jeweiligen Funktionen der einzelnen Menüpunkte. Diese Menüpunkte finden sich auf der Navigationsleiste der Anwendung wieder (s. Abb. 17).

Abbildung 17 Comparify-Menüführung

Zu Beginn der Menüabfolge trifft der Nutzer eine Auswahl eines Musiktitels. Anschließend erhält er auf der Seite *Ranking* einen Eindruck, wie der ausgewählte Musiktitel in der Gesamtbetrachtung einzuordnen ist. Dazu werden alle Audio-Features der fünf stärksten Übereinstimmungen in der Visualisierung gegenübergestellt. Im Menüpunkt *Compare* erfolgt ein direkter Vergleich mit dem Song mit der höchsten Übereinstimmung. In dieser Gegenüberstellung erfährt der Nutzer einen detaillierten Einblick, wie sich der Musiktitel in Relation zu einem anderen Musiktitel verhält. Als letzte Funktion der Menüführung wird dem Nutzer eine Übersicht aller Eigenschaften des zu untersuchenden Musiktitels dargestellt. Auf dieser Ebene werden die Details und Funktionsweisen der einzelnen Audio-Features vermittelt. Der Nutzer durchläuft durch die hier skizzierte Struktur einzelne Stationen, in denen er die unterschiedlichen Merkmale und Einordnung musikalischer Rekonstruktionen erlernt.

4.2 Benutzeroberfläche

Der erste Schritt bei der Erstellung einer geeigneten Konzeptionierung besteht darin, ein Wireframe zu erstellen. Die Aufgabe eines Wireframes besteht darin, die einzelnen Eingabemöglichkeiten zu sammeln und in einem Layout zu strukturieren. Das Wireframe ist der erste Entwurf, in dem die Erfahrungen des Endnutzers im Vordergrund stehen. Anders als in einem Vorführmodell, werden keine visuellen Entscheidungen getroffen. Demnach wird im Vorfeld der Aufbau der Anwendung geplant und auf die Anwenderfreundlichkeit geprüft, bevor der eigentliche Prozess der Produktion beginnt. Dieser Schritt der Konzeptionierung verbindet die Planung und Umsetzung zwischen Designer und Programmierer. Der Fokus bei der Erstellung eines Wireframes liegt darauf, den Ablauf und die Funktionsweise der Anwendung für den Benutzer nachvollziehbar zu gestalten. Die Funktion der Farbgestaltung bei einem Wireframe liegt primär den Entwicklern und Designern auf Besonderheiten, Funktionen oder Kategorisierungen hinzuweisen. Um systematische Abläufe der Anwendung darzustellen, kann als Ergänzung ein Flussdiagramm hinzugezogen werden. Dies hilft im Entwicklungsprozess, einen Überblick über die einzelnen UI-Komponenten zu verschaffen und verschiedene Szenarien für den Programmcode durchzuplanen (Vgl. Yang 2016: 565-566). Für die weitere Durchführung der zu erstellenden Anwendung wurden Wireframes angefertigt (s. Anhang Abb. 44 bis 46). Im Folgendem werden Ausschnitte aus den Wireframes dargestellt und erläutert.

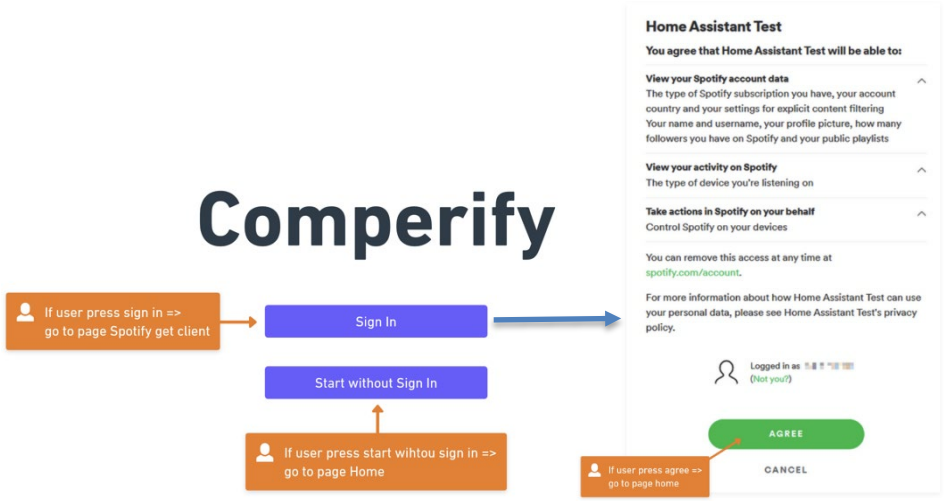


Abbildung 18 Ausschnitt Wireframe: Login-Seite von Comparify (links), Login-Seite von Spotify (rechts)

Zu Beginn der Anwendung wird mit der jeweiligen Login-Methode ein Client angefordert, welcher für die zukünftigen API-Abfragen benötigt wird. Der Benutzer kann sich dabei entscheiden, ob er sich mit seinem eigenen Account einloggen und somit die

Vorteile des integrierten Musikplayers nutzen oder ohne Account fortzufahren will (s. Abb. 18). In dem Szenario einer Account-Verifizierung wird der Benutzer zum Login-Portal von Spotify weitergeleitet. Mit der Verifizierung erteilt der Nutzer die Freigabe seiner Benutzerdaten in Form von *Scopes* (developer.spotify.com 2022: Authorization Scopes). Zusätzlich wird jedem Benutzer die Möglichkeit geboten, die Anwendung unabhängig eines Spotify-Accounts zu nutzen (s. Abb. 18 Button: Start without Sign In). Ist die Abfrage erfolgreich, wird in beiden Fällen ein Client mit den jeweiligen Berechtigungen an die Anwendung übermittelt.

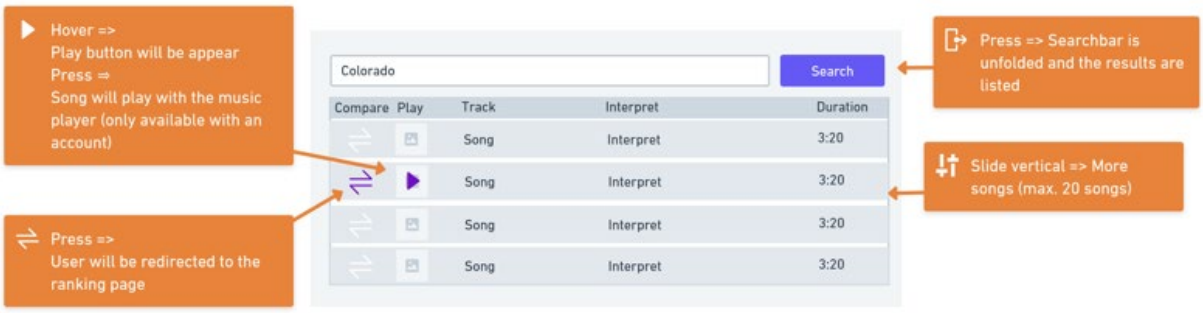


Abbildung 19 Ausschnitt Wireframe: Suchleiste

Konnte ein Client erfolgreich angefordert werden, wird der Nutzer zur Home-Seite weitergeleitet. In der Suchleiste kann der Nutzer im Eingabefeld nach einem beliebigen Track suchen (s. Abb. 19). Die Resultate der Suchanfrage werden im unteren Fenster aufgelistet. Entscheidet sich der Nutzer für einen Musiktitel und drückt mit der Maus auf den Button *Compare*, wird der Nutzer zur nächsten Seite *Ranking* weitergeleitet. Ist der Nutzer mit einem Account eingeloggt, kann dieser mittels des *Play*-Button einen Musiktitel über den integrierten Player abspielen (s. Abb. 20). Der Benutzer-Status wird auf der rechten oberen Seite der Anwendung eingeblendet (s. Abb. 21 Account-Status).

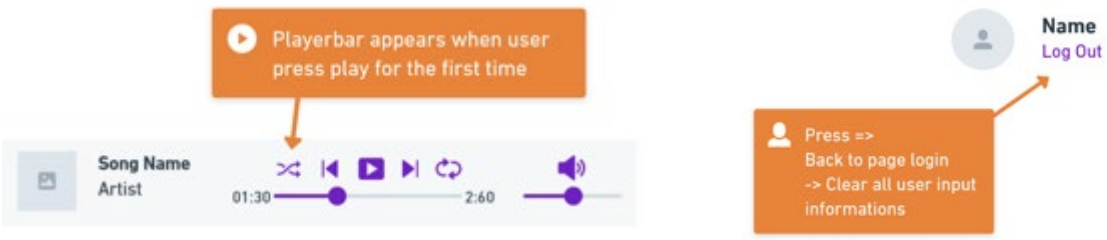


Abbildung 20 Ausschnitt Wireframe: Musikplayer (links), Account-Status (rechts)

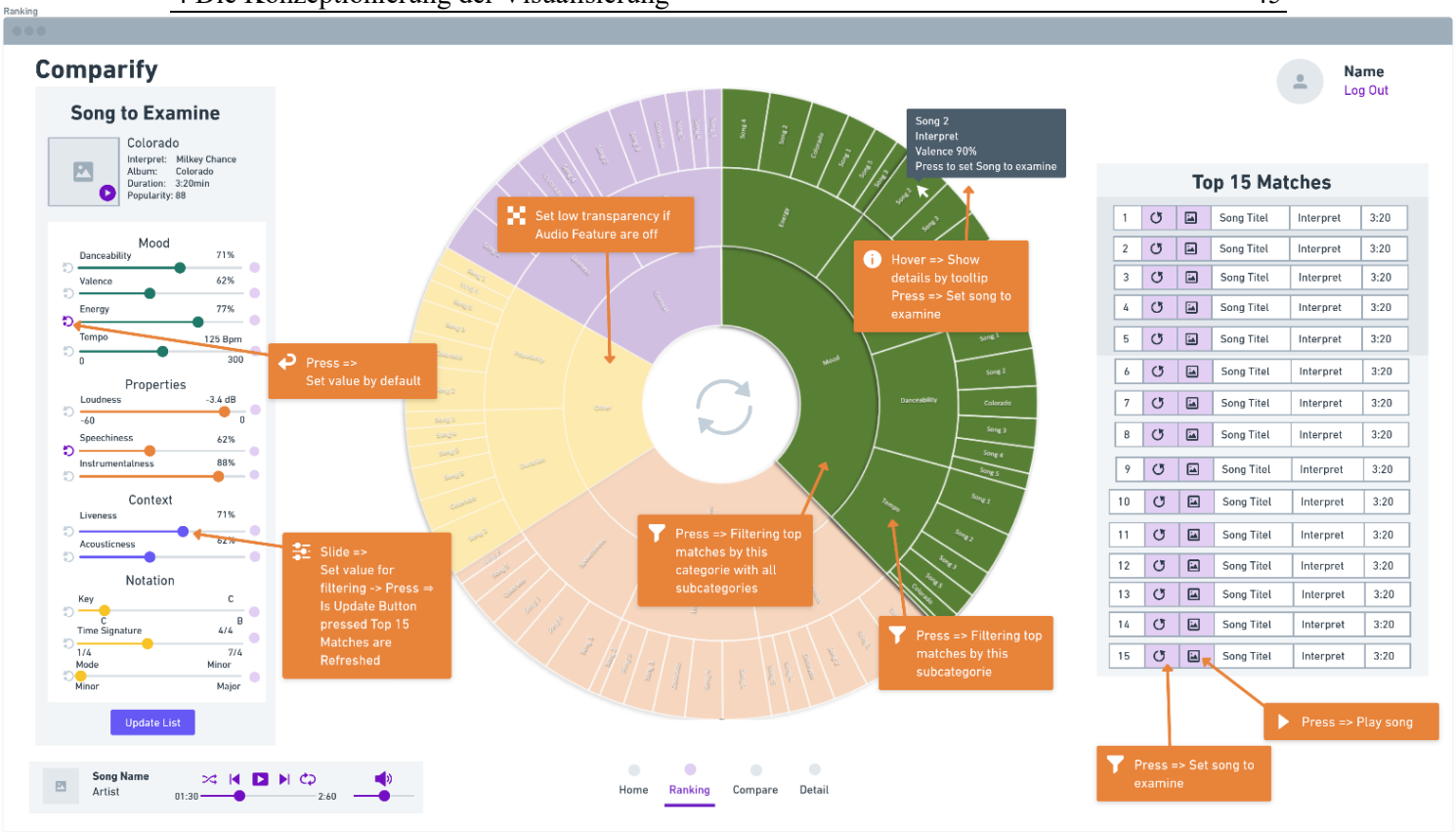


Abbildung 21 Wireframe: Ranking

Abbildung 21 zeigt das Wireframe für die Seite *Ranking*. Auf der linken Fläche der Abbildung werden die Informationen des ausgewählten Musiktitels angezeigt. Im oberen Infofeld sind allgemeine Informationen wie der Titelname, Interpret, Album, Titellänge, Popularität und Albumcover eingeblendet. Wie auch in der Liste der Suchleiste auf der Hauptseite hat der Nutzer die Möglichkeit, mit dem Click auf das Musikcover den Titel mit dem integrierten Player abzuspielen. In der darunterliegenden Fläche sind alle Audio-Features des jeweiligen Musiktitels aufgelistet. Ausgehend von den Parametern des ausgewählten Musiktitels ist der Nutzer in der Lage, die einzelnen Features zu modifizieren und die Suche nach ähnlichen Titeln zu spezifizieren. Wurde eine Modifizierung angewendet, kann die Liste übereinstimmender Musiktitel aktualisiert werden. Diese Liste ist auf 15 Stück limitiert und nach ihrer höchsten Übereinstimmung sortiert. Die fünf stärksten Übereinstimmungen fließen in das Diagramm ein. Die Limitierung der Liste garantiert eine übersichtliche Visualisierung. Ein weiteres Feature der Anwendung bilden die *UI-Tweens*, mit denen kurze Ladezeiten überbrückt und temporäre UI-Elemente ausgeblendet werden können. Diese kurzen Animationen erzeugen weiche Übergänge, indem dieses die jeweiligen Objekte durch ihre Transformationen über einen bestimmten Zeitraum modulieren. Dieses Feature wird im Implementierungsprozess näher erläutert.

Das dargestellte *Sunburst*-Diagramm im Wireframe ist ein Platzhalter für das in der Anwendung tatsächlich umgesetzte Diagramm. Die skizzierten Funktionen in der Abbildung werden auf das eigentliche Diagramm übertragen. Ein weiteres Mittel für eine übersichtliche Visualisierung bietet die Hover-Funktion mittels des Mauszeigers. Mit dieser Funktion können nähere Details über einen Musiktitel und den jeweiligen Datenpunkt dargestellt werden. Ausgelagerte Parameter, die der Nutzer nicht in die Selektierung der gefilterten Musiktitel einfließen lassen will, werden in den Diagrammen transparent dargestellt.

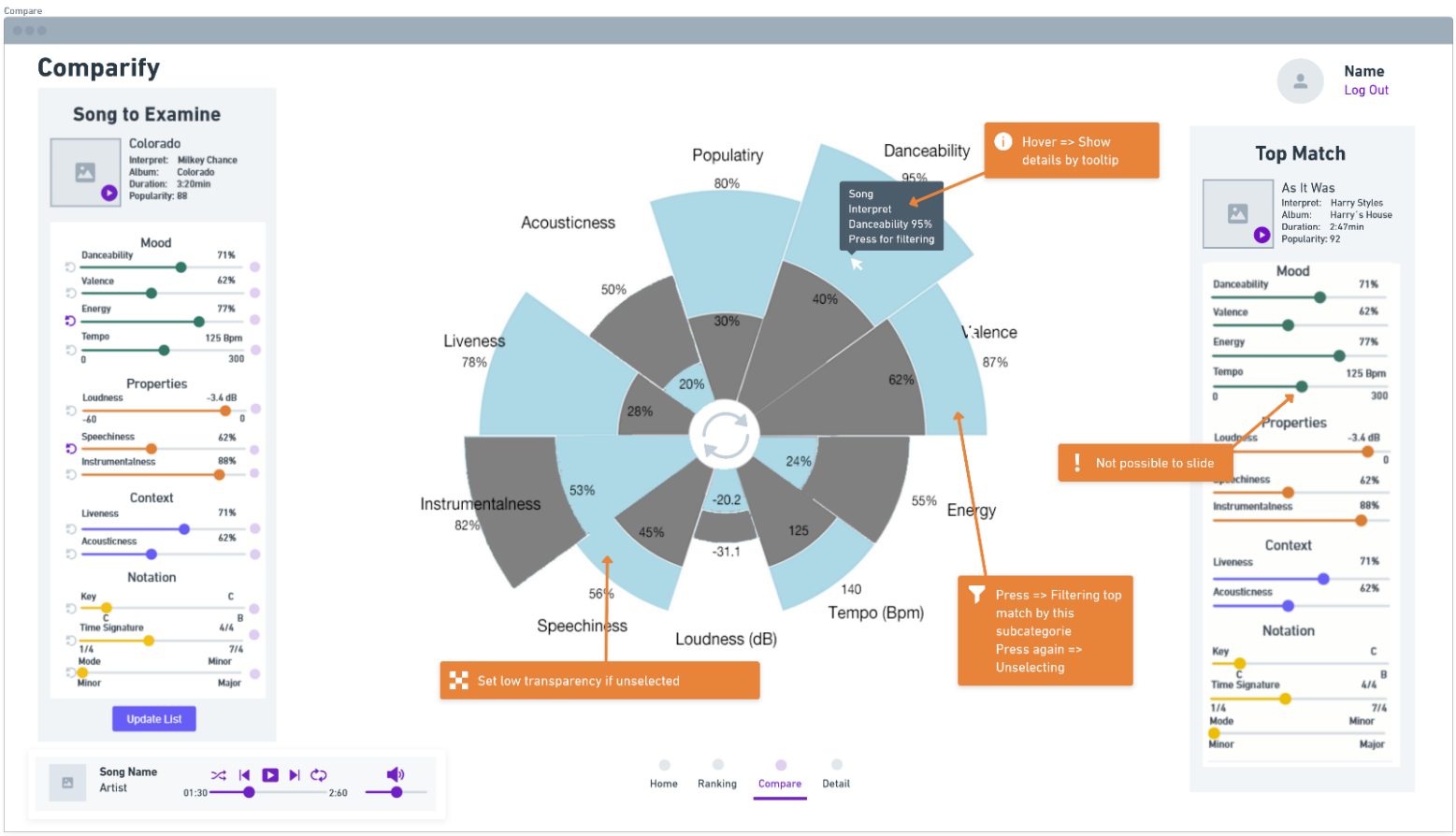


Abbildung 22 Wireframe: Compare

In Abb. 22 ist das Wireframe für die Seite *Compare* dargestellt. Wie in Abschnitt *4.2.2 Menüführung* beschrieben, besteht die Funktion dieser Seite darin, einen direkten Vergleich zwischen dem ausgewählten und dem ähnlichsten Musiktitel zu ziehen. Das Panel auf der linken Seite bleibt auf der Seite *Ranking* bestehen. Auf der rechten Seite wird ein ähnliches Panel mit dem korrelierenden Musiktitel angezeigt. Mit der Einschränkung können die Schieberegler nicht verstellt oder ausgeschaltet werden, da eine Justierung der Werte nicht notwendig ist. Wie auch beim vorherigen Wireframe der Seite *Ranking*, ist das dargestellte Diagramm ein Platzhalter für das eigentliche

Diagramm. Die Interaktionen mit dem Diagramm sind ähnlich wie mit dem vorherigen. In diesem Szenario wird die Hover-Funktion ebenfalls für die Darstellung der Details der einzelnen Datenpunkte genutzt. Inaktive Parameter, die aus der Filterung ausgelagert werden, sind gleichfalls im Diagramm transparent dargestellt und können per Mausklick selektiert werden.

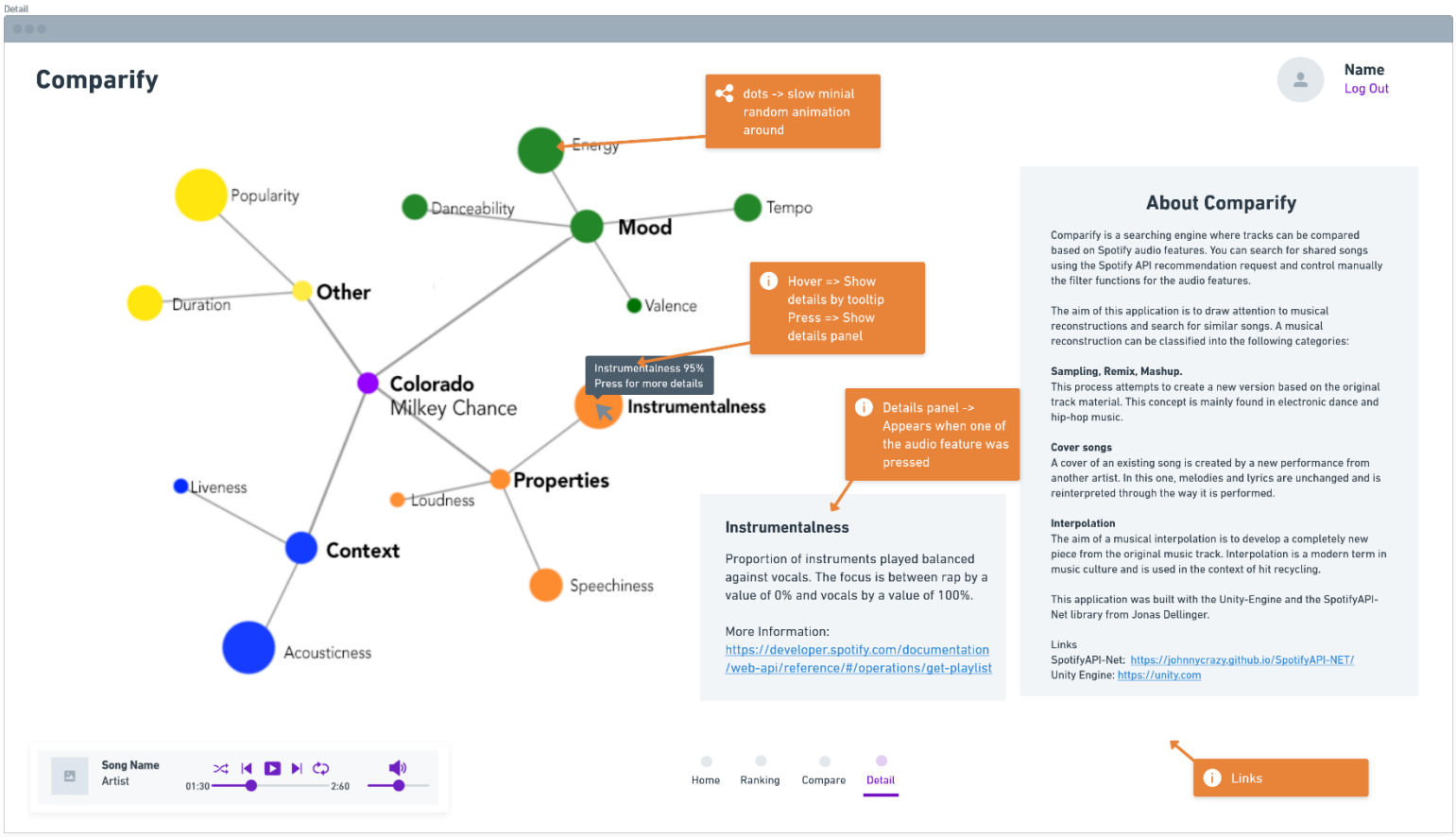


Abbildung 23 Wireframe: Detail

Auf der letzten Seite *Detail* werden alle ausgewerteten Audio-Features des zu analysierenden Musiktitels in einem Netzwerk-Diagramm zusammengefasst (s. Abb. 23). Diese Seite dient zur Aufklärung der musikalischen Rekonstruktionen und Bedeutung der einzelnen Audio-Features. Die Größe der Kreise im Netzdiagramm repräsentiert die relative Ausprägung der einzelnen Features. Detailliertere Informationen über die jeweiligen Features erhält der Nutzer über die Hover- und Press-Funktion. Das rechte Infofeld dient ebenfalls als Quellenangabe und Verknüpfung weiterer Referenzen. In diesem Panel werden die drei Formen der musikalischen Rekonstruktion erklärt.

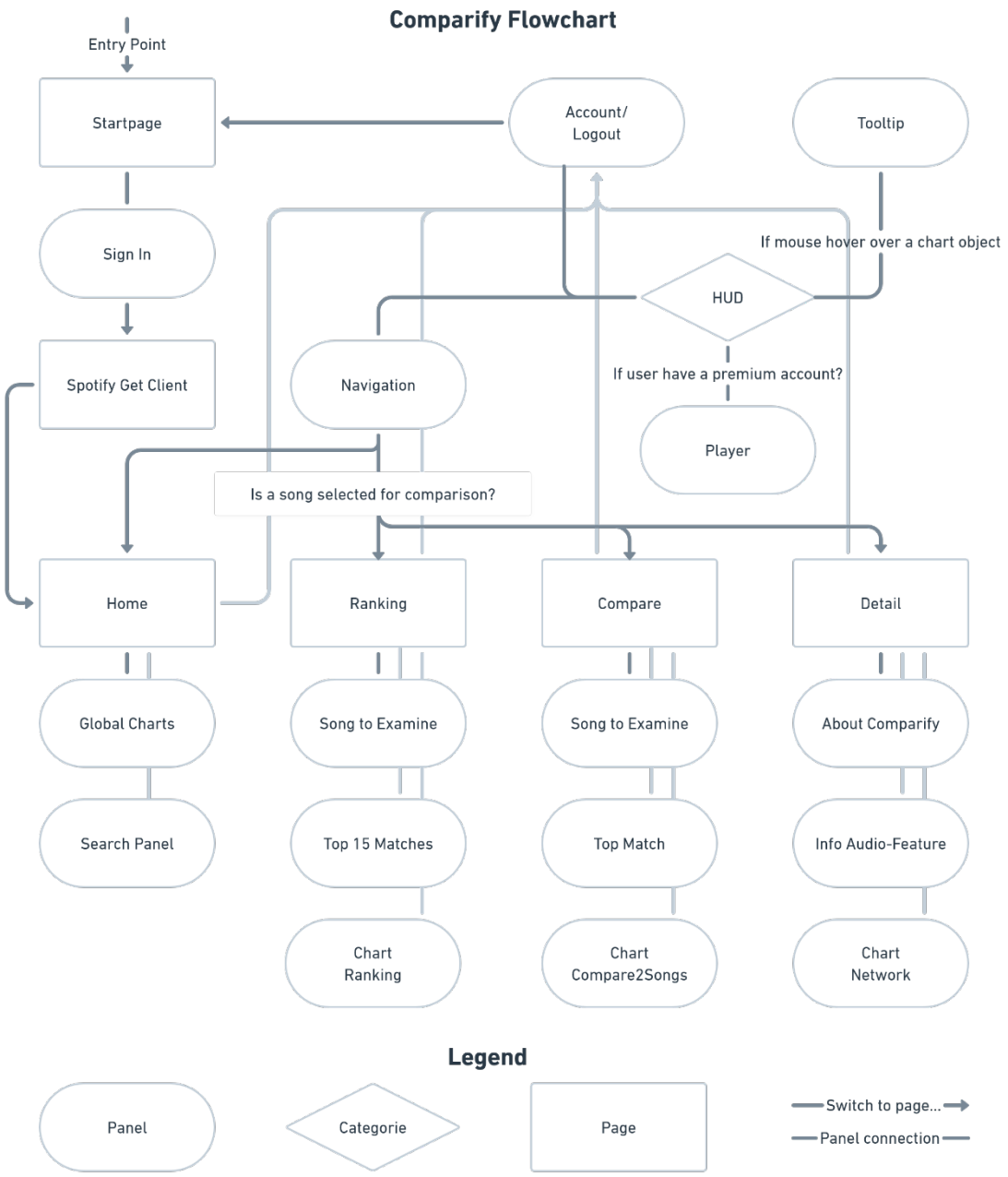


Abbildung 24 Comperify: Flowchart

In Abbildung 24 ist der Flowchart der Anwendung dargestellt. Dieser Flowchart veranschaulicht alle Verbindungen, die zwischen den Seiten und Panels vorliegen. Die Legende deutet auf die vorhandenen Symbole und Verbindungen hin und etikettiert diese ihrer Assoziation zu. Dabei umfasst die Ellipse alle Panels, die auf den unterschiedlichen Seiten eingeblendet werden. Zur Einteilung der Kategorien wird die geometrische Form einer Raute verwendet. Die Rechtecke symbolisieren die einzelnen Seiten. Die Pfeile veranschaulichen die Verknüpfungen und Assoziationen der jeweiligen Elemente. Diese Abbildung kann dazu genutzt werden, um Programmabläufe und Designentscheidungen besser einordnen zu können. Zudem beschreibt der Flowchart alle Prozessschritte, die der Anwender bei der Benutzung der Anwendung durchläuft.

4.3 Diagrammtypen

In diesem Abschnitt erfolgt die Beschreibung der konzeptionellen Ausarbeitung der Diagramme, die in der Anwendung *Comperify* zur Visualisierung der Spotify-Datensätze verwendet werden. Dazu werden die verschiedenen Funktionsweisen der Diagrammtypen mit den Anforderungen der Visualisierung verglichen. Die Zusammenfassung im darauffolgenden Kapitel subsummiert alle beschriebenen Diagrammtypen und stellt sie den Referenzen aus Kapitel 3 gegenüber.

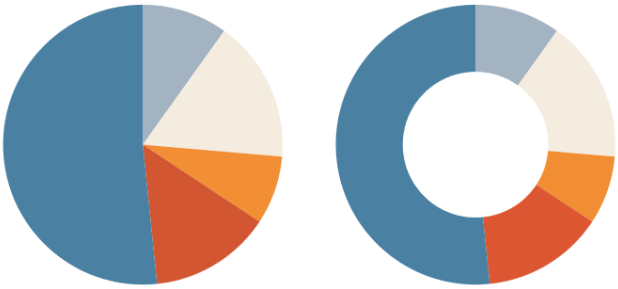


Abbildung 25: Pie-Chart (links) und Donut-Chart (rechts), (Vgl. Cai 2017: 2)

Insgesamt setzt sich die Anwendung aus drei verschiedenen Visualisierungen zusammen, die sich thematisch an der Menüführung orientiert. Nach erfolgreicher Auswahl eines Musiktitels auf der Hauptseite, wird der Nutzer auf die Seite *Ranking* weitergeleitet. Die Funktion dieser Seite besteht darin, dem Nutzer seinen ausgewählten Musiktitel im Gesamtkontext zu korrelierenden Musiktiteln in Bezug der Audio-Features einordnen zu können. Die Anforderungen an die Visualisierung besteht demnach darin, eine Vielzahl an Musiktiteln auf der Ebene der Hierarchie und dem Faktor einzelner Kategorien zu vergleichen. Ein *Donut-Chart* veranschaulicht den prozentualen Anteil mehrerer Segmente in einer Grafik (s. Abb. 25 rechte Seite). Mit diesem Diagrammtyp lassen sich die Größenverhältnisse der einzelnen Musiktitel in den jeweiligen Kategorien vergleichen. Der *Donut-Chart* ist ähnlich zu einem *Pie-Chart* (s. Abb. 25 linke Seite) mit dem Unterschied, dass die Mitte des *Donuts-Chart* nicht ausgefüllt ist und daher mehr Fläche für weitere Informationen in der Visualisierung bietet.

Um die Rangfolge zu strukturieren, hilft der Lösungsansatz, die Musiktitel nach ihren prozentualen Anteilen auf- oder absteigend zu sortieren (Vgl. Skau 2016: 2). Der Einsatz von Farben soll die Hierarchie der Musiktitel verdeutlichen, indem die Sättigung der Farbe mit dem prozentualen Anteil des jeweiligen Musiktitels korreliert. Diese Sortierung der Daten steigert zudem die Übersichtlichkeit bei einer hohen Anzahl an Musiktiteln. Um das *Donut-Chart* kategorisch nach Audio-Features zu gliedern und übersichtlich zu

gestalten, werden die vier Gruppen (*Mood, Properties, Notation, Context*) in mehrere Charts aufgeteilt und die jeweiligen Kategorien in einem Donut-Chart unterteilt. Das *Baumdiagramm* auf der linken Seite der Abb. 25 veranschaulicht die Einteilung der Gruppen mit den jeweiligen Audio-Features. Auf der rechten Seite der Abbildung sind die *Donut-Charts* skizziert. Diese Skizze dient als Vorlage für den weiteren Implementierungsprozess und veranschaulicht den strukturellen und farblichen Aufbau der ersten Visualisierung. Die genaue Umsetzung der Visualisierung wird in Kapitel 5 näher erläutert.

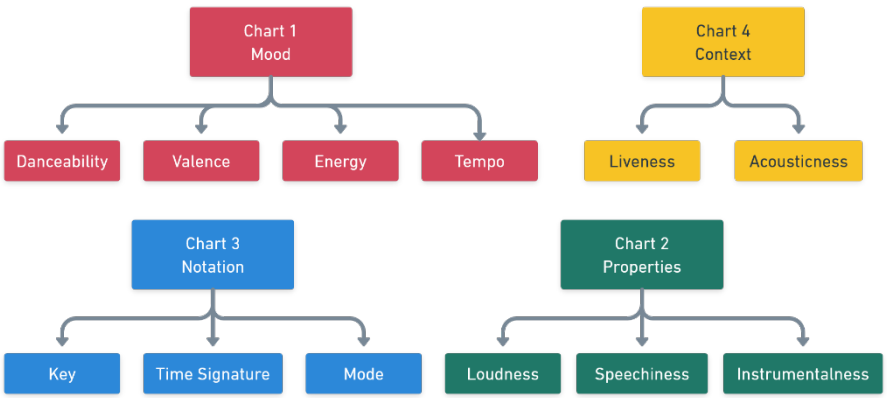


Abbildung 26 Farbliche Einteilung der Gruppen und Kagegorien

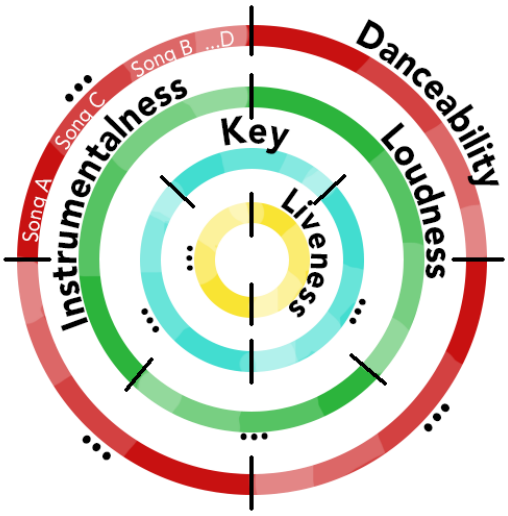


Abbildung 27 Skizze Visualisierung: Ranking

Die zweite Visualisierung fokussiert sich auf den direkten Vergleich zwischen dem ausgewählten und dem dazu korrelierenden Song. Das Ziel dieser Visualisierung ist es, die einzelnen Audio-Features gegenüberzustellen. Um die Datenpunkte vergleichen zu können, bietet das *Nightingale-Rose-Chart* die beste Voraussetzung (s. Abb. 29). Dieses Diagramm ist in mehrere Kategorien unterteilt und stellt eine abgewandelte Form eines Balkendiagramms dar. Durch seine radiale Ausrichtung ist es möglich, eine hohe Anzahl an Kategorien in einer kompakten Form darzustellen. Daher eignet sich dieser Diagrammtyp besonders für die Veranschaulichung der 12 verschiedenen Audio-Features. Wie auch bei einem Balkendiagramm, wird die Ausprägung der einzelnen Datenpunkte durch die Höhe bestimmt. Die Farbgestaltung wird bei diesem Diagrammtypen dazu verwendet, um die einzelnen Vergleichsproben zu unterteilen. Zudem können die einzelnen Segmente über den Höhepunkten der Balken mit einer Beschriftung der jeweiligen Kategorie ergänzt werden (Vgl. Shabdin 2020: 4). Abbildung 26 zeigt die skizzierte Abwandlung des *Nightingale-Rose-Charts* für die Anwendung Comperify. In dieser Abwandlung ist ein Teil der Mitte für die Beschriftung der einzelnen Spalten mit den Kategorien vorgesehen. Diese Anordnung hat den Vorteil, den Platz über

den jeweiligen Spalten für den Namen und Wert der jeweiligen Vergleichsproben nutzen zu können. Die Farbeinteilung dient zur Gruppierung der einzelnen Kategorien. Um die Vergleichsproben innerhalb einer Spalte unterscheiden zu können, wird eine unterschiedliche Farbsättigung verwendet.

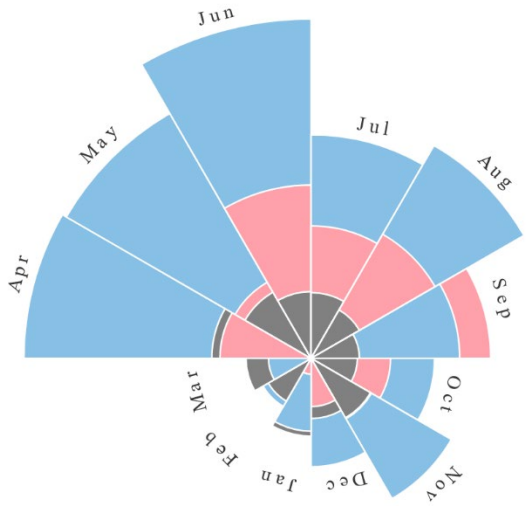


Abbildung 29 Nightingale Rose (Vgl. Ribecca (o. J.))

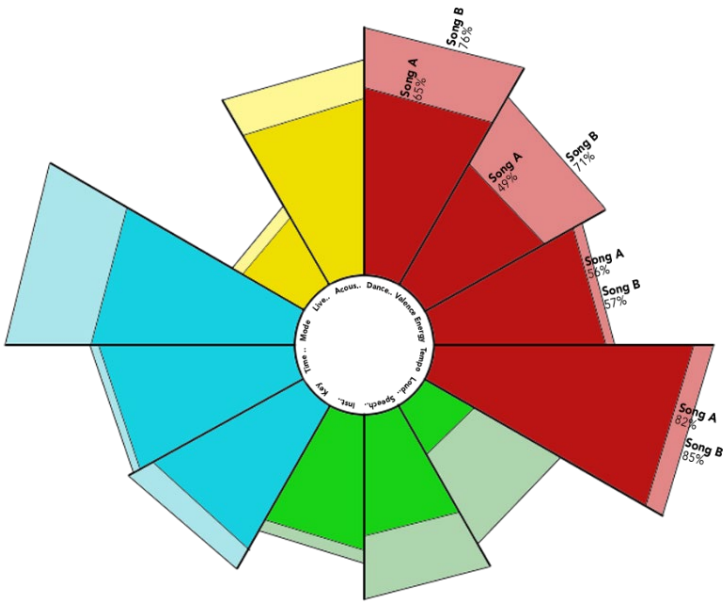


Abbildung 28 Skizze Visualisierung: Compare

Die dritte Visualisierung dient der Darstellung und Beschreibung der Eigenschaften der einzelnen Audio-Features. Dies erfolgt durch die Zusammenfassung der Audio-Features, in der die Verknüpfungen innerhalb eines *Netzwerkdiagramms* abgebildet werden (s. Abb. 30). In der Mitte des *Netzwerkdiagramms* steht der zu untersuchende Musiktitel und gliedert sich in den jeweiligen Gruppen. Diese Knotenpunkte sind wiederum in den einzelnen Kategorien unterteilt. Wie stark ein Audio-Feature für den jeweiligen Song ausgeprägt ist, wird durch die Größe der Knotenpunkte bestimmt. Wie auch bei den vorherigen Visualisierungen, dient die Farbeinteilung der Zuordnung der jeweiligen Gruppe. Die Farbsättigung der einzelnen Knotenpunkte bekräftigen zudem die Ausprägungen der einzelnen Datenwerte. Um die einzelnen Audio-Features bei diesem Diagrammtypen zueinander besser vergleichen zu können, werden die einzelnen Datenwerte unter dem jeweiligen Namen mit angegeben. Die Eigenschaften eines selektierten Audio-Features werden neben dem *Netzwerkdiagramm* durch eine Infobox beschrieben. Diese Infobox liefert eine detaillierte Beschreibung und Funktionsweise des jeweiligen Audio-Features. Diese gestalterischen Mittel ermöglichen es dem Nutzer, die Bedeutung und Einordnung der Audio-Features näher zu untersuchen.

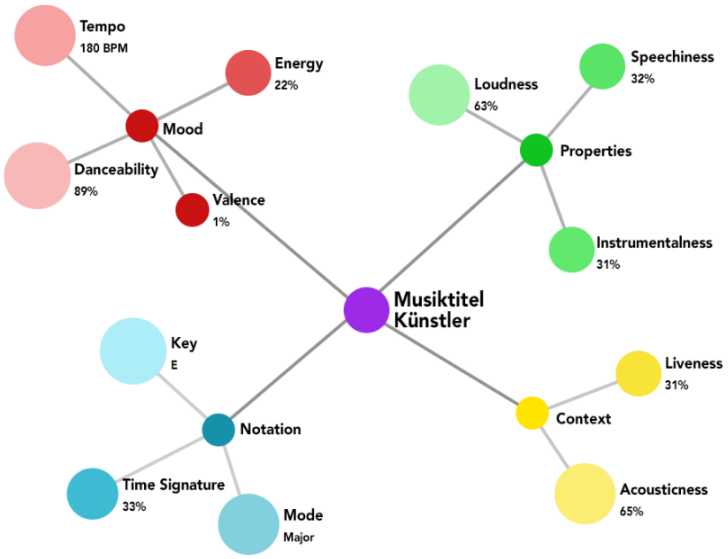


Abbildung 30 Skizze Visualisierung: Detail

Wie zu Beginn dieses Kapitels beschrieben, orientieren sich die drei Visualisierungen an den konzeptionellen Ansätzen der Menüführung. Um einen einheitlichen konzeptionellen Ansatz zu erzeugen, weisen alle drei Diagramme verschiedene Gemeinsamkeiten auf. Zum einem wird die Gruppeneinteilung der einzelnen Audio-Features durch die Zuordnung der Farbe gekennzeichnet (s. Abb. 27). Zum anderen verstärkt die Farbsättigung bei der ersten und dritten Visualisierung die Ausprägung der Audio-Features. Im Gegensatz dazu wird bei der zweiten Visualisierung die Farbsättigung dazu eingesetzt, um eine Unterscheidung der beiden Vergleichsproben vorzunehmen. Eine weitere Gemeinsamkeit liegt bei der Beschriftung der Diagramme. Dabei werden die Segmente nach ihrer jeweiligen Kategorie und alle Datenwerte mit dem jeweiligen Musiktitel und Wert gekennzeichnet. Im folgenden Abschnitt werden die Aufgaben der Visualisierungen genauer erläutert und mit den Visualisierungen aus Kapitel 3 verglichen.

4.4 Positionierung und Abgrenzung der Visualisierungen

Das Ziel der Anwendung *Comperify* besteht darin, dem Nutzer eine individuelle Erkundung verwandter Musiktitel sowie den direkten Vergleich und die Einordnung eines spezifischen Musiktitels zu ermöglichen. In Kapitel 3 wurden bereits Ansätze vergleichbarer Visualisierungen genannt. In diesem Abschnitt werden diese Ansätze mit der eigenen Visualisierung verglichen. Obwohl die Zielsetzungen der Visualisierungen unterschiedlich sind, lassen sich bei der Gegenüberstellung Besonderheiten und Unterschiede zur eigenen Visualisierung feststellen.

In der ersten genannten Visualisierung aus Kapitel 3 verwendet der Autor eine multidimensionale Darstellung, um seine Datensätze zu repräsentieren und auf unterschiedliche Weise zu mappen. Dabei nutzt er die Gestaltungsmittel Farbe, Bewegung, Aufteilung und Größe, um die Daten zu veranschaulichen. Der Autor erwähnt in seiner Ausarbeitung vier verschiedene Diagrammtypen, die jeweils eine Dimension seiner Visualisierung darstellen. Zum Beispiel bezieht er sich auf das Histogramm, das den zeitlichen Verlauf bestimmter Datenpunkte in einem Standbild veranschaulicht. Um die Historie seiner Daten dynamisch darzustellen, verwendet der Autor Animationen, die den zeitlichen Verlauf der Datenwerte in einem kurzen Zeitraffer darstellen. Der zeitliche Abstand wird mit einer Datumsanzeige auf der linken Seite übermittelt und kann manuell angepasst werden. In der Visualisierung von *Comperify* gibt es keine zeitliche Zuordnung der Daten, jedoch spielen Animationen in Form von *UI-Tweens* eine wichtige Rolle bei der Interaktion und Manipulation der Daten. Mit diesem Ansatz können kurze Ladezeiten beim Abruf der Daten überblendet und irrelevante Informationen ein- und ausgeblendet werden. Um die Datensätze manuell anzupassen, stehen dem Nutzer in der ersten und zweiten Visualisierung mehrere Schieberegler zur Verfügung. Mit ihnen kann der Nutzer die Suche nach ähnlichen Musiktiteln im Rahmen der Audio-Features beeinflussen und spezifizieren.

Im Gegensatz zur Visualisierung von Schwarz werden die einzelnen Ebenen nicht in einer Darstellung zusammengefasst, sondern in drei unterschiedliche Visualisierungen aufgeteilt. Dies hat den Vorteil, die musikalischen Eigenschaften unter mehreren spezifischen Blickwinkeln zu betrachten. Wie in Abschnitt 4.1 Menüführung beschrieben, übernehmen die Visualisierungen unterschiedliche Funktionen. Diese Menüführung soll dem Nutzer entsprechend dem Trichtersystem die verschiedenen Merkmale musikalischer Rekonstruktionen näherbringen. Zum einen fokussiert sich die erste Visualisierung auf die Einordnung des ausgewählten Musiktitels auf den Gesamtkontext. In der zweiten Visualisierung erfolgt eine direkte Gegenüberstellung des Musiktitels mit der größten Übereinstimmung. Als letztes werden die Audio-Features des ausgewählten Musiktitels zusammengefasst und im Detail betrachtet. Das Datenmapping von *Comperify* erfolgt somit im Gegenzug zu *Klangspektrum* auf mehreren unterteilten Ebenen. Dieser Ansatz gestaltet die Visualisierung übersichtlich und nachvollziehbar.

Für das Rangieren einer Rangfolge liefert die Visualisierung *Music-Circles* einen beispielhaften Ansatz. Um die Datensätze nach ihrer Rangfolge zu ordnen, werden die Datenätze nach ihrer Größe sortiert und radialförmig angeordnet (s. Abschnitt 3.2, Abb. 11). Zudem gibt die Höhe der einzelnen Balken die jeweilige Ausprägung an. Nach diesem Prinzip ist es möglich, eine Vielzahl an Datensätze auf kompakter Ebene zu präsentieren. Dieser Ansatz ist ein essenzielles Konzept für die erste Visualisierung in Comperify, da eine beliebig große Anzahl an Datenwerten abgebildet werden müssen. Im Gegensatz zu *Music-Circles* werden die einzelnen Datenpunkte kategorisch und absteigend sortiert (s. Abschnitt 4.3, Abb. 26). Die Ausprägungen der einzelnen Datenpunkte werden bei allen drei Visualisierungen ebenfalls durch die Höhe bzw. Größe definiert. Um diesen Effekt zu verstärken, wird die Farbsättigung der einzelnen Datenpunkte je nach Ausprägung stärker oder schwächer reguliert.

Die Visulisierung *MuicViz* liefert ein Beispiel einer Clusteranalyse. Wie in Abschnitt 3.3 erläutert, beschreibt das Clustering das Aufbereiten struktureller Ähnlichkeiten innerhalb eines Datensatzes. Um die Ähnlichkeiten und Gruppenzugehörigkeiten der Datensätze zu unterteilen, nutzen die Autoren der Visualisierung eine farbliche Unterscheidung. In der Hauptanwendung (s. Abschnitt 3.3, Abb. 9) wird die Klassifizierung des Musikgenres farblich unterschieden. Zudem findet beim *Netzwerkdiagramm* eine Unterteilung der Gruppen durch die einzelnen Knotenpunkte statt. Diese Einteilung wird auch in

Comperify auf vielen Ebenen angewendet. Zum einen werden die Gruppen *Mood, Context, Notation* und *Properties* in den Farben Rot, Grün, Blau und Gelb dargestellt (s. Abschnitt 4.3, Abb. 27). Dieses Konzept wird bei allen Visualisierungen in *Comperify* angewendet und bildet den strukturellen Leitfaden der Anwendung. Wie auch bei der Hauptanwendung von *MusicViz*, werden die Gruppen in der dritten Visualisierung von *Comperify* zudem durch ein Netzwerkdiagramm klassifiziert. Mit dem Unterschied liefert die Farbsättigung zusätzlich eine Aussage über die Rangordnung der Datensätze. In der ersten Visualisierung werden die Gruppen den jeweiligen *Kreisdiagrammen* zugeordnet und in den einzelnen Kategorien unterteilt. Mit diesen Ansätzen der Farbkodierung und Untergliederung können die Datensätze auf einheitliche Weise gekennzeichnet und klassifiziert werden.

In diesem Vergleich wurden die Visualisierungen anhand ihrer Merkmale wie Datenmapping, Darstellung von Rangfolgen und Clustering untersucht. Viele der genannten Ansätze wie beispielsweise die Klassifizierung mittels der Farbgestaltung in der Visualisierung *Music-Circle* finden sich in mehreren Referenzen wieder. In diesem Kapitel wurden die relevanten Aspekte für die konzeptionelle Umsetzung der Visualisierungen von *Comperify* genannt. Zudem verfolgten die aufgeführten Referenzen jeweils eine andere Zielsetzung. Jedoch beziehen sich alle Visualisierungen auf unterschiedliche Metadaten der Spotify-API. Die Aufgabe von *Comperify* besteht darin, nach ähnlichen Musiktiteln zu suchen und zu vergleichen. Dies wird durch den Ansatz einer dreiteiligen Visualisierung aus mehreren verschiedenen Perspektiven hervorgehoben. Die Anwendung konzentriert sich darauf, die Recherche musikalischer Rekonstruktionen für jeden zugänglich zu machen. Wie im Grundlagenkapitel beschrieben, setzt sich die musikalische Rekonstruktion aus den drei Themenbereichen der Interpolation, Reinterpretation durch ein Cover oder Wiederverwertung zusammen. Durch das angewendete Trichterprinzip in der Menüführung wird der Nutzer an diese Themenbereiche schrittweise herangeführt.

4.5 Zusammenfassung

Dieses Kapitel beschreibt die konzeptionelle Ausarbeitung von Comperify. Dazu wurden die Strukturen und Abläufe der Anwendung festgelegt und die Diagrammtypen für die Visualisierung begründet. Einleitend zum Kapitel wurden die Ansätze der Menüführung genannt. Ziel der Menüführung ist es, den Nutzer von einer Gesamtansicht bis hin zu den Details der Songanalyse zu führen. Im Gegensatz zum Trichterprinzip soll der Nutzer schrittweise durch die Gesamt- bis zur Detailansicht geführt werden.

Zu Beginn der Konzeptionierung wurden Wireframes erstellt. Diese Wireframes dienen zur Strukturierung des Layouts und fassen alle Eingabemöglichkeiten zusammen. Dabei bilden die Wireframes den ersten Entwurf, bei denen die Erfahrungen und die Führung des Endnutzers im Fokus stehen. Das Flowchart wurde erstellt, um alle Verbindungen zwischen den Seiten und Panels der Anwendung zu veranschaulichen.

Im darauffolgenden Kapitel wurden die verschiedenen Diagrammtypen in Hinblick der Anforderungen der Anwendung vorgestellt. Dabei besteht die Anwendung aus drei verschiedenen Visualisierungen, die sich am Prinzip der Menüführung orientieren. Der Lösungsansatz zur Strukturierung der Rangfolge der Musiktitel basiert auf den prozentualen Anteilen der Audio-Features. Um diese Rangfolge zu veranschaulichen, werden die Musiktitel nach ihren Anteilen absteigend sortiert. Durch den Einsatz von Farben wird die Rangfolge zusätzlich verdeutlicht, wobei die Sättigung der Farbe mit dem prozentualen Anteil des jeweiligen Musiktitels korreliert.

Im letzten Abschnitt des Kapitels wurden die Visualisierungen in Kapitel 3 mit den eigenen Visualisierungen vergleichen. Aus diesem Vergleich wurden die Positionierung und Abgrenzung der Visualisierungen abgeleitet. Im Unterschied zu den korrelierenden Visualisierungen, besteht die eigene Visualisierung aus drei Teilen. Durch das Trichterprinzip in der Menüführung wird der Nutzer schrittweise an die Themenbereiche der Visualisierung durchgeführt. Das Ziel der interaktiven Visualisierung besteht darin, die Recherche musikalischer Rekonstruktionen für jeden zugänglich zu machen. Die Suche nach Interpolationen, Reinterpretationen und Wiederverwertungen verschiedener Musiktitel sollen durch die Gegenüberstellung der Audio-Features ermöglicht werden.

.

5. Der Implementierungsprozess der Anwendung

In diesem Kapitel werden die Hintergrundprozesse für die Umsetzung von *Comperify* näher erläutert. Dazu wird Bezug auf die konzeptionellen Ansätze dieser Arbeit genommen und unter den technischen Aspekten betrachtet. Der Fokus dieses Kapitels liegt darauf, die Einbindung und Vorgänge bei der Verwendung der Spotify-API und die Umsetzung der Diagramme zu beschreiben. Der Prozess der Implementierung spiegelt die Ausführung der Anwendung wider und begründet die Vorgehensweise der technischen Ausführung dieser Arbeit.

Im ersten Abschnitt des Kapitels werden die unterschiedlichen Autorisierungsmodelle bei der Nutzung der Spotify-API-Anfragen vorgestellt. Diese Modelle liefern unterschiedliche Vor- und Nachteile, die bei der Umsetzung der Anwendung berücksichtigt werden müssen. Im darauffolgenden Abschnitt werden die angewendeten Filterprozesse bei der Suche korrelierender Songs erläutert und die relevanten API-Abfragen vorgestellt, die bei der Visualisierung der Datensätze eine Schlüsselfunktion übernehmen. Das dritte Kapitel erläutert die Umsetzung der in Abschnitt 4.3 beschriebenen Diagramme. Hier wird beschrieben, wie die gestalterischen Mittel in der Unity-Engine umgesetzt wurden. Im letzten Abschnitt des Kapitels werden die angewendeten Prozesse zusammengefasst und die relevanten Punkte zusammengetragen.

5.2 Autorisierungsvorgang in der Spotify-API

Eine API ist eine Programmierschnittstelle, in der Daten in Form von Quelltexten unabhängig von der Programmiersprache und dem System ausgetauscht werden können. Die Quelltexte werden oftmals im JSON-Format als ein Objekt verschachtelt und können mehrere Datentypen beinhalten. Der Interpreter bzw. das Computerprogramm entscheidet, inwiefern die JSON-Dateien verarbeitet und verstanden werden. APIs dienen oftmals für die Verwaltung von Datenbanken oder die Nutzung verschiedener Webdienste. Im Fall eines Webdienstes werden die Datenpakete mit einer URL-Adresse über die Internetverbindung angefragt. Diese Adresse beinhaltet den Namen der angefragten Datei und den Suchpfad als Umgebungsvariable (Vgl. Schröder 2018: 1-2).

Die Bereitstellung solcher Schnittstellen geht mit einer umfangreichen Dokumentation einher. Im Fall der Spotify-API ist die Dokumentation auf der Webseite „*Spotify for Developers*“ öffentlich zugänglich (Vgl. developer.spotify.com (o. J.): Web API). In diesem Abschnitt wird der Autorisierungsvorgang der Spotify-API genauer erläutert und die unterschiedlichen Autorisierungsmodelle vorgestellt. Anschließend wird festgelegt, welche Modelle für die Anwendung verwendet werden.

Um API-Anfragen an die Spotify-Datenbank stellen zu können, bieten sich unterschiedliche Vorgehensweisen an. Als erster Schritt muss ein Entwickler-Account auf der Developer Seite angelegt werden. Auf dem Dashboard des Entwickler-Portals kann anschließend eine Applikation angemeldet werden. In den Einstellungen der Applikation wird dem Entwickler eine Client ID übermittelt. Diese Client ID dient als Zugriffstoken für zukünftige Anfragen, das je nach Autorisierungsmodell unterschiedliche Zugangsberechtigungen besitzt. Dem Entwickler werden vier verschiedene Autorisierungsmodelle angeboten, die er in seiner Anwendung implementieren kann. Abbildung 31 listet alle Autorisierungsmodelle mit ihren jeweiligen Berechtigungsformen auf (Vgl. developer.spotify.com 2022: Authorization).

FLOW	ACCESS USER RESOURCES	REQUIRES SECRET KEY (SERVER-SIDE)	ACCESS TOKEN REFRESH
Authorization Code	Yes	Yes	Yes
Authorization Code With PKCE	Yes	No	Yes
Client Credentials	No	Yes	No
Implicit Grant	Yes	No	No

Abbildung 31 Authorization Flows (developer.spotify.com (o. J.): Authorization)

Das erste Modell eignet sich für Anwendungen, die langlaufend die Berechtigung des Benutzers erlauben und den Zugriffstoken von selbst stetig aktualisieren. In diesem Modell wird eine geheime Client-ID aus dem Entwickler-Portal benötigt, welche neben den notwendigen Zugangsberechtigungen weitere Benutzerinformationen über den Entwickler abfragt. Diese ID sollte von der Öffentlichkeit möglichst unzugänglich sein. Dieses Modell eignet sich daher für Backend-Dienste, welche nicht direkt in einem Browser oder einer mobilen Anwendung ausgeführt werden. Das zweite Modell ist eine Erweiterung zum ersten Modell und benötigt keine geheime Client ID. Diese Autorisierungsform ist bei der Umsetzung einer mobilen oder nativen Anwendung die richtige Wahl, da die Verwendung einer geheimen Client ID unsicher ist (Vgl. developer.spotify.com 2022: Authorization Code Flow).

Das Autorisierungsmodell *Client Credentials* besitzt keinen Zugriff auf Benutzerinformationen. Dieses Modell kann nur auf die öffentlichen Datenbanken von Spotify zugreifen und benötigt daher kein Benutzerkonto. Mit dieser Einschränkung können benutzerspezifische Funktionen, wie das Ansteuern des Musikplayers nicht genutzt werden. Mit diesem Modell ist die Anwendung für jeden Nutzer unabhängig eines Spotify-Accounts zugänglich (Vgl. developer.spotify.com 2022: Client Credential Flow).

Das Autorisierungsmodell *Implicit Grant* benötigt keinen serverseitigen Code. Zudem benötigt die Anwendung keine geheime Client-ID. Eine automatische Aktualisierung des Tokens erfolgt in dieser Form nicht und muss daher stetig aktualisiert werden. Dieses Modell eignet sich daher für Webanwendungen mit einer vollständigen Implementierung in JavaScript (Vgl. developer.spotify.com 2022: Implicit Grant Flow).

Die Anforderung der Funktionalitäten von *Comperify* besteht darin, mit Hilfe von Datensätzen der Spotify-API eine interaktive Visualisierung verwandter Musiktitel zu erzeugen. Zudem sollen möglichst vielen Nutzern der Zugang der Anwendung ermöglicht werden. Zum anderen liefern benutzerspezifische Funktionen, wie das Ansteuern eines Musikplayers, interessante Features. Diese Features führen zu einer Steigerung der Qualität der Anwendung. Der Nutzer soll zudem in der Lage sein, seine Suchanfragen zu spezifizieren, um nach musikalischen Rekonstruktionen zu recherchieren. *Comperify* soll als native Desktop-Anwendung entwickelt werden.

Unter diesen Voraussetzungen eignen sich die zwei Autorisierungsmodelle *Client Credentials* und *Authorization Code with PKCE* für die Einbindung der Spotify-API. Ein wichtiger Bestandteil der Anwendung ist der Zugriff der Datenbank von Spotify. Ohne diesen Zugriff können keine Visualisierungen umgesetzt und keine Suchanfragen ähnlicher Musiktitel durchgeführt werden. Beide Modelle eignen sich für die Umsetzung nativer Desktop-Anwendungen. Im Unterschied dazu erfordert das zweite Modell ein Spotify-Account, welcher bei erfolgreicher Verknüpfung auf benutzerspezifische Funktionen zurückgreifen kann. Mit diesem Zugriff kann der integrierte Musikplayer in der Anwendung angesteuert werden. Dieses Modell vermindert jedoch die Zugänglichkeit der Anwendung. Das erste Modell kann unabhängig eines Spotify-Accounts von jedem Nutzer verwendet werden. Auf benutzerspezifische Funktionen kann jedoch nicht zurückgegriffen werden. Aus diesen Erkenntnissen lässt sich schließen, dass die Integrierung beider Autorisierungsmodelle in die Anwendung die Anforderung am nächsten erfüllt. Im folgenden Abschnitt werden die Hintergrundprozesse der Anwendung näher erläutert.

5.3 Hintergrundprozesse und Suchalgorithmus

In diesem Kapitel werden die Hintergrundprozesse von *Comperify* beschrieben. Dazu werden die Funktionsweisen der einzelnen Komponenten in der Anwendung genannt und zudem die angewendete Methodik beim Suchalgorithmus genauer untersucht. Für die Kommunikation zwischen den Datenbanken der Spotify-API und den einzelnen Komponenten werden diverse API-Anfragen angewendet.

Um die API-Anfragen in der Programmiersprache C# zu vereinfachen, liefert die Spotify API-NET-Library von Jonas Dellinger eine gute Grundlage. Diese Library ist mit jedem System kompatibel, welches vom .NET-Framework unterstützt wird. Daher beschränkt sich die Implementierung dieser Library nicht nur auf die Unity-Engine, wie es in der praktischen Ausarbeitung der Fall ist. In der Anwendung werden die vorgefertigten Objekte der Library dazu genutzt, um die angefragten JSON-Dateien sinnvoll für die Weiterverarbeitung im Quellcode aufzubereiten. Zudem liefert die Library hilfreiche Funktionen, die bei der Sortierung und Verknüpfung der Datenpakete im Quellcode eine zentrale Rolle übernehmen (Vgl. Dellinger (o. J.): SpotifyAPI-NET). In den folgenden Abschnitten dieses Kapitels werden Ausschnitte des Quellcodes dargestellt, in denen die Implementierung der Library im Einzelnen veranschaulicht wird. Zudem werden die einzelnen Programmprozesse der Komponenten erläutert und kommentiert.

Wie oben geschildert, bilden die Zugriffstoken die Voraussetzung der jeweiligen API-Anfragen. Diese Tokens werden in der Anwendung mit den Autorisierungsmodellen *Client Credentials* und *Authorization Code with PKCE* angefordert. Wie in Abschnitt 2.2 beschrieben, entscheidet der Nutzer zu Beginn der Anwendung, ob die Anwendung mit seinem Spotify-Account verknüpft werden soll oder nicht (s. Abb. 18). Entscheidet der Nutzer sich für ein Modell, wird die jeweilige Methode aufgerufen und der Nutzer bei erfolgreicher Anmeldung zum Homescreen weitergeleitet.

Abbildung 32 veranschaulicht den Methodenaufruf für das Autorisierungsmodell *Client Credentials*. Zu Beginn der Methode wird eine Instanz mit dem Namen „SpotifyClientConfig“ erzeugt. Diese Instanz dient zur Konfiguration der entsprechenden Authentifikation. Je nach Wahl des Modells wird beim Aufruf des Konstruktors der Authentifikations-Klasse die Client-ID und ggf. auch die geheime Client-ID benötigt. In der Abbildung wurden die eigenen IDs durch die Begriffe „clientID“ und „clientSecret“

ersetzt. Die daraus resultierende Instanz mit dem Namen „config“ wird an die Instanz „SpotifyClient“ weitergeleitet. Diese Instanz liefert das Client-Objekt, mit welchem alle kommenden API-Aufrufe angefragt werden können. Daher wird dieses Objekt, wenn es erfolgreich instanziiert werden konnte, an das Client-Objekt der Klasse „spotifyService“ deklariert. Das deklarierte Objekt kann von jeder weiteren Klasse aus referenziert werden, da es sich bei der Service-Klasse um ein „SceneSingleton“ handelt (Vgl. hextantstudios.com 2022: Singleton in Unity). Zudem wird ein boolescher Wert für „withAccount“ mit in die Klasse übermittelt, welcher aussagt, ob auf die benutzerspezifischen Funktionen zugegriffen werden darf oder nicht. Sind diese Schritte erfolgt, wird der Nutzer mit dem Methodenaufruf „UpdateHUD“ auf die Hauptseite der Anwendung weitergeleitet.

```
private void OnSignIn()
{
    SpotifyClientConfig config = SpotifyClientConfig
        .CreateDefault()
        .WithAuthenticator(new ClientCredentialsAuthenticator("clientID", "clientSecret"));

    SpotifyClient client = new SpotifyClient(config);

    if (client != null)
    {
        spotifyService.client = client;
        spotifyService.withAccount = false;
        logic.UpdateHUD(3, true);
    }
    else
    {
        Debug.LogError("Client Credentials not found");
    }
```

Abbildung 32 Quellcode-Ausschnitt: Methodenaufruf "OnSignIn"

Die nächste Komponente, die implementiert wurde, ist die Suchanfrage mehrerer Tracks. In dieser Komponente gibt der Nutzer seine Suchanfrage über ein Eingabefeld ein. Anschließend werden die Resultate nach ihrer Übereinstimmung im darunterliegenden Fenster aufgelistet (s. Abb. 19, Abschnitt 4.2). Der API-Aufruf der Suchanfrage lautet:

```
https://api.spotify.com/v1/search?q=blind&type=track&limit=15&offset=0
```

Wie aus der Anfrage entnommen werden kann, benötigt der Aufruf die Texteingabe „q=blind“ und die entsprechende Gattung „type=track“. Zu der Anfrage kann optional ein Limit von maximal 50 Songs und ein Offset von maximal 1000 Songs angegeben werden.

In der Anwendung wurde dieser Aufruf mit der Spotify-C#-Library wie folgt angefordert und zugewiesen:

```
private async void OnPerformSearch()
{
    var client = SpotifyService.Instance.GetSpotifyClient();

    if (client != null && _searchField != null)
    {
        string query = _searchField.text;
        SearchRequest request = new SearchRequest(SearchRequest.Types.All, query);

        SearchResponse _lastSearchResponse = await client.Search.Item(request);

        List<FullTrack> _trackList = _lastSearchResponse.Tracks.Items;
    }
}
```

Abbildung 33 Quellcode-Ausschnitt: Methodenaufruf "OnPerformSearch"

Zunächst wurde wie zu Beginn beschrieben, der Client aus dem „SpotifySerivce“ aus dem Methodenaufruf mit einer lokalen Variablen deklariert. Bevor die Suchanfrage startet, wird geprüft, ob ein gültiger Client und String im Eingabefeld vorliegen. Wurden die Datenpakete erfolgreich angefragt und heruntergeladen, werden die Ergebnisse an die Liste „FullTrack“ übergeben. Aus dieser Liste werden anschließend einzelne *Prefabs* (Vgl. Seifert 2015: 373) erstellt und in der darunterliegenden Benutzeroberfläche aufgelistet (s. Abb. 33). Mit dem Button „Compare“ auf der linken Seite der einzelnen *Prefabs* kann der Nutzer einen Musiktitel auswählen. Mit diesem Musiktitel wird anschließend die Datenvisualisierung erstellt.

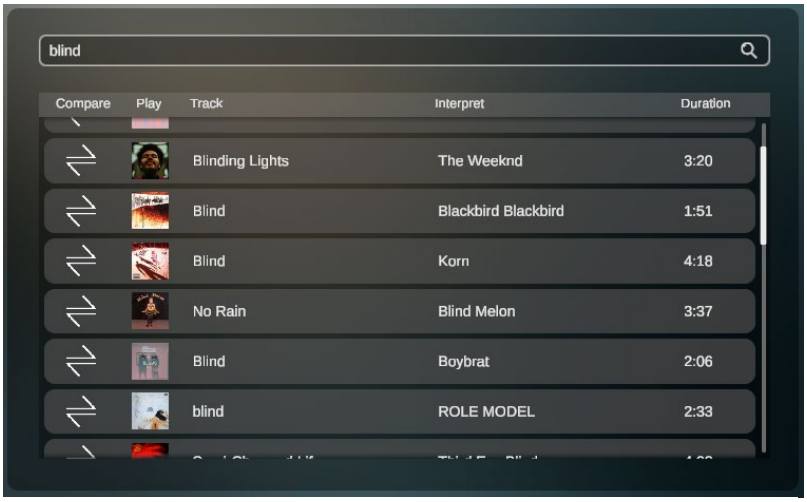


Abbildung 34 Comperify-Ausschnitt: Suchanfrage

Wurde sich für ein Musiktitel entschieden, wird der Nutzer zur nächsten Seite *Ranking* weitergeleitet. Der ausgewählte Musiktitel enthält Metainformationen, die aus der Klasse *FullTrack* stammen (Vgl. developer.spotify.com 2022: Get Track). Diese Informationen umfassen den Künstler, die Popularität und andere Details, die im Kopf des Musiktitel-

Panels auf der linken Seite dargestellt werden (s. Abb. 21, Abschnitt 4.2). Um die Audio-Features einer oder mehrerer Musiktitel abzufragen, wird folgende URL verwendet:

```
https://api.spotify.com/v1/audio-features?ids=0VjIjW4GlUZAMYd2vXMi3b
```

Bei dieser Anfrage verwendet man die Musiktitel-IDs aus der Metainformation der *FullTrack*-Klasse. Zurückgeliefert wird eine JSON-Datei, welche die entsprechenden Werte der Audio-Features enthält. Diese Dateien bilden die Grundlage für die Berechnung und Darstellung der entsprechenden Diagramme. Welche Funktion und Bedeutung jedes einzelne Audio-Feature besitzt, wurde in Abschnitt 2. 1 Tabelle 1 beschrieben. Die nachfolgende Abbildung zeigt den Rückgabewert der Anfrage:


```
{
  "audio_features": [
    {
      "danceability": 0.514,
      "energy": 0.73,
      "key": 1,
      "loudness": -5.934,
      "mode": 1,
      "speechiness": 0.0598,
      "acousticness": 0.00146,
      "instrumentalness": 0.0000954,
      "liveness": 0.0897,
      "valence": 0.334,
      "tempo": 171.005,
      "type": "audio_features",
      "id": "0VjIjW4GlUZAMYd2vXMi3b",
      "uri": "spotify:track:0VjIjW4GlUZAMYd2vXMi3b",
      "track_href":
"https://api.spotify.com/v1/tracks/0VjIjW4GlUZAMYd2vXMi3b",
      "analysis_url": "https://api.spotify.com/v1/audio-
analysis/0VjIjW4GlUZAMYd2vXMi3b",
      "duration_ms": 200040,
      "time_signature": 4
    }
  ]
}
```


Abbildung 35 JSON-Datei: Audio-Features

Recommendations bilden die Grundlage für die Suchanfrage korrelierender Musiktitel. Mit dieser Methode werden Nutzer-Empfehlungen weiterer Musiktitel generiert. Standardmäßig benötigt die Anfrage ein Musiktitel-, Musikgenre- oder Künstler-ID. In der jeweiligen Rubrik können bis zu vier weitere ID-Nummern angegeben werden. Diese nehmen Einfluss auf den Algorithmus. Bezeichnet werden diese Parameter auch als *Seeds*, da diese die Streuung der *Recommendation* beeinflussen. Bei diesem Algorithmus wird ein bestimmtes Analyseverfahren angewendet, aus dem die Playlisten generiert werden.

Ausgangspunkt des Analyseverfahrens sind die drei Faktoren Ähnlichkeit, Vertrautheit und Entdeckung. Eine ausgewogene Gewichtung dieser Komponenten steigert die personalisierte Empfehlungsqualität. Der Kompromiss zwischen einer entdeckungsorientierten und zentrierten Methodik liefert einen Ansatz für die Problemlösung eines nachhaltigen Musikvorschlags. Jeder Faktor des Analyseverfahrens verfolgt dabei ein eigenes Prinzip und ist unterschiedlich definiert. Die Ähnlichkeit definiert sich durch die Fähigkeit, Inhalte auf Basis der Vorlieben und dem Verhalten des Nutzers zu empfehlen. Dabei wird das Interessensprofil des Nutzers anhand seiner Verlaufsdaten ausgewertet und eine Häufigkeitsverteilung gehörter Songs erstellt. Die Vertrautheit bezieht sich auf den vorherigen Zugang zu den Inhalten. Besitzt der Nutzer eine positive Affinität zu den Inhalten, erhöht sich die Wertung in der Empfehlung. Dieser Faktor reduziert die Unsicherheiten des Nutzers, da die Erwartungen durch ein erhöhtes Verständnis vertrauensrelevanter Inhalte erfüllt werden. Die Entdeckung neuer Inhalte trägt maßgeblich dazu bei, die Entwicklung der Plattform durch die Empfehlung weniger bekannter Künstler nachhaltig zu verbessern. Der Nutzer erhält durch die Entdeckung neuer Inhalte eine Abwechslung zu den stagnierenden Empfehlungen. Um die genannten Faktoren ermitteln zu können, werden die Audio-Features der jeweiligen Musiktitel in Abhängigkeit zueinander gebracht und verglichen (Vgl. Mehrotra 2021: 3997-3998). *Recommendations* können mit folgender URL angefragt werden:

```
https://api.spotify.com/v1/recommendations?limit=10&market=DE&seed_tra
cks=0VjIjW4GlUZAMYd2vXMi3b&min_acousticness=0.7&target_danceability=0.
9&target_loudness=-3.1&max_tempo=100
```

Um die Streuung der *Recommendations* für die Suche musikalischer Rekonstruktionen zu spezifizieren, können die Wertebereiche der Audio-Features durch optionale Parameter eingegrenzt werden. Für das jeweilige Audio-Feature kann dabei das Minimum „min_acousticness=0", Maximum „max_tempo=100" oder der Zielwert „target_danceability=0.9" angegeben werden. Wie in Abschnitt 2.1 angesprochen, variieren die Wertebereiche je nach Kategorie. Beispielsweise wird das Tempo als ganzzahliger Wert ab Null aufwärts angegeben, während die *Danceability* durch Fließkommazahlen im Bereich zwischen 0 und 1 angegeben wird. Zu diesen Werten kann die Anzahl der Empfehlungen und die Spezifizierung der Länder eingetragen werden (Vgl. developer.spotify.com 2022: Get Recommendations).

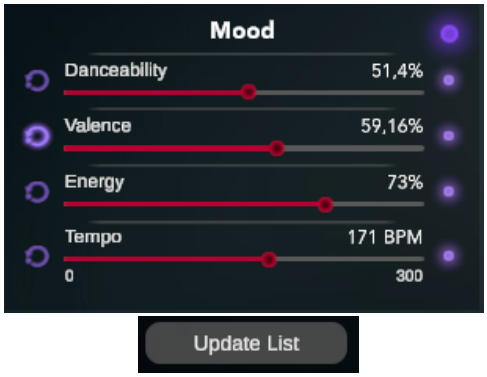


Abbildung 37 Ausschnitt aus der Anwendung: Musikpanel

```
_recommendationsRequest = new RecommendationsRequest()
{
    Limit = 16,
    SeedTracks = { _fullTrack.Id }
};
//[...]
_recommendationsRequest.Target.Add("danceability",
   featureList[0].ToString().Replace(",", "."));

_recommendationsRequest.Target.Add("valence",
    featureList[1].ToString().Replace(",", "."));
//[...]
_recommendationsResponse = await client.Browse.
    GetRecommendations(_recommendationsRequest);
```

Abbildung 36 Quellcode-Ausschnitt: Recommendations

Wie in Abschnitt 4.2 beschrieben, können die Parameter für die Suchanfrage vom Nutzer spezifiziert werden. Diese Modifizierung der Suchanfrage wird im linken Panel des ausgewählten Musiktitels auf der Seite „Ranking" vorgenommen. Abbildung 36 zeigt ein Ausschnitt des Panels, in dem die Audio-Features in der Kategorie „Mood" als *Slider* dargestellt werden. Wird die Seite „Ranking" zum ersten Mal aufgerufen, werden die Werte der Audio-Features des zu untersuchenden Musikstücks in den einzelnen *Slider* übernommen. Die genauen Werte der Audio-Features können auf der rechten Seite der *Slider* abgelesen werden. Von diesem Ausgangspunkt aus können die Audio-Features für die Spezifizierung der Suchanfrage verändert werden. Wurden Änderungen vorgenommen, leuchtet der Button auf der linken Seite auf. Mit diesem Button können die Werte auf den Ausgangspunkt zurückgesetzt werden. Auf der rechten Seite der *Slider* können die einzelnen Audio-Features oder auch die Gruppe „Mood" mit dem Button ausgeschaltet werden. Diese Werte werden in der Suchanfrage nicht berücksichtigt. Mit dem Button „Update List" können die Eingaben der Suchanfrage bestätigt und die Liste der Musikempfehlungen aktualisiert werden.

In der Abb. 37 ist der Funktionsaufruf der *Update*-Methode abgebildet. Mit der Klasse „RecommendationRequest“ kann die ID des entsprechenden Musiktitels und die Anzahl der Musikempfehlungen angegeben werden. Zu dieser Konfiguration können die Werte für die Spezifizierung mit dem Funktionsaufruf „Add“ hinzugefügt werden. In der Liste „FeatureList“ sind die entsprechenden Werte der Audio-Features des zu untersuchenden Songs hinterlegt und werden jeweils in dem Funktionsaufruf hinzugefügt. Die Funktionsaufrufe „ToString“ und „Replace“ formatieren jeweils die Werteangaben für die Konfiguration der *Recommendation*. Anschließend werden die Empfehlungen mithilfe der modifizierten Konfiguration angefragt und in die Liste „_recommendationsResponse“ gespeichert. Diese Liste bildet das Resultat der spezifizierten Suchanfrage und somit die Grundlage für die Datenvisualisierung.

In diesem Kapitel wurde der Hintergrundprozess und Suchalgorithmus der Anwendung, anhand der URL-Anfragen der Spotify-API und der Implementierung der C#-Library von Dellinger beschrieben. Dieses Kapitel bildet ein Teil des Implementierungsprozesses ab. Die Funktion der Musikempfehlungen der Spotify-API und die damit verbundenen Spezifizierung der Audio-Features, bietet die Grundlage für die Suche nach musikalischen Rekonstruktionen. Im folgenden Kapitel werden die Umsetzung und der Implementierungsprozess der interaktiven Visualisierungen, anhand der gestalterischen Mittel und Funktionen der Unity-Engine erläutert.

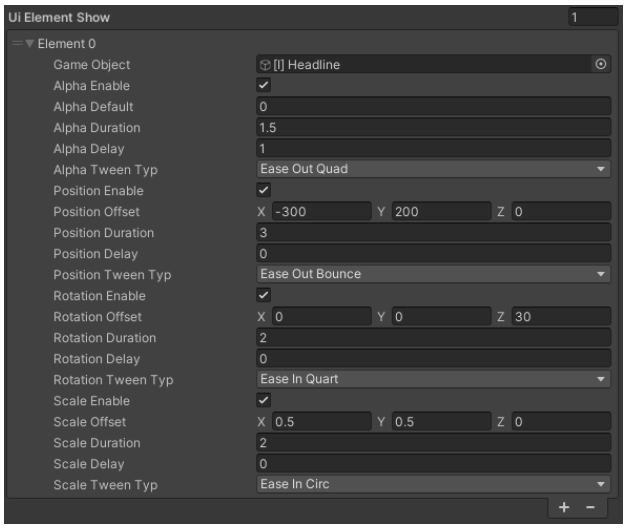
5.4 Die Umsetzung der Visualisierungen in der Unity-Engine

Dieser Abschnitt befasst sich mit der technischen Umsetzung der Visualisierungen. Als Fundament bietet die Unity-Engine viele Möglichkeiten, die Benutzeroberflächen und Visualisierungen zu gestalten. Das Unterkapitel liefert einen Einblick in die visuelle Ausführung der Anwendung. Während der vorherige Abschnitt auf die Hintergrundprozesse der Anwendung aufbaut, befasst sich dieser mit der Benutzeroberfläche und der Umsetzung der jeweiligen Diagramme.

Um die Übergänge und Transformationen einer Benutzeroberfläche ansprechender zu gestalten, werden *UI-Tweens* verwendet. Zudem überbrücken die *Tweens* im Fall von *Comperify* die Ladezeiten der jeweiligen URL-Anfrage an die Spotify-API. Unity liefert verschiedene Möglichkeiten, dieses gestalterische Mittel in die Anwendung einzubinden. Zum einen liefern die integrierten Animationstechnologien die Möglichkeit, *Tweens* ohne komplexen Code auf den Elementen der Benutzeroberfläche anzuwenden. Zum anderen bietet die Verwendung von Plugins wie *LeanTween* oder *DOTween* eine Vielzahl an Funktionen, um die Übergänge in ihrer Bewegung, Skalierung, Rotation und Farbänderung zu gestalten. Im Vergleich zu den integrierten Animationstechnologien der Unity-Engine, werden beim Plugin *LeanTween* keine *Keyframes* verwendet. In diesem Fall stehen dem Entwickler beim Plugin vorgefertigte Animationskurven zur Verfügung, welche die Transformationen in Abhängigkeit der Zeit fest definieren. Im Umkehrschluss lassen sich die Bewegungen mithilfe der Animationskurven vereinfachen. *Keyframes* hingegen liefern eine detailliertere Justierung der Animation (Vgl. Díaz 2015: 423-425).

Das *UI-Tweening* ist für das Überbrücken der Ladezeiten der URL-Anfragen ein zentraler Bestandteil der Anwendung. Daher wurde mit der Basis von *LeanTween* ein UI-Animationstool entwickelt, welches die Übergänge einzelner oder mehrerer Objekte in der Anwendung vereinfacht (s. Abb. 38). Mit diesem Tool können die *Tweens* in Abhängigkeit der Transparenz, Position, Rotation oder Skalierung für die jeweiligen UI-Elemente definiert werden. In der Anwendung sind die einzelnen UI-Elemente in Gruppen eingeteilt. Wie in Abschnitt 4.2 beschrieben, repräsentiert der Flowchart in Abb. 24 nicht nur den Verlauf der Anwendung, sondern dient auch zur Erfassung der UI-Gruppen (s. Abb. 24, *Panels*). Mit dem Animationstool können die Gruppen mit individuellen Einstellungen auf- und abgebaut werden. Dieser Ansatz vereinfacht nicht

nur die Umsetzung der Programmabläufe, sondern dient auch als gestalterisches Mittel bei der visuellen Ausarbeitung der Benutzeroberfläche.

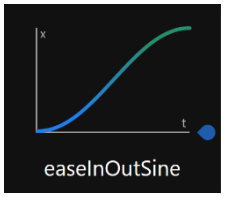



```
#region Position
public bool positionEnable;

[ConditionalField("positionEnable")]
public Vector3 positionOffset;

[ConditionalField("positionEnable")]
public float positionDuration, positionDelay;

[ConditionalField("positionEnable")]
public LeanTweenType positionTweenTyp =
    LeanTweenType.easeInOutSine;
#endregion
```



Sitnik (o. J.): Übergangsfunktionen

```
private void SetPosition()
{
    gameObject.transform.position = show ? _positionOpp : _position;
    LeanTween.move(gameObject, show ? _position : _positionOpp, positionDuration)
        .setDelay(positionDelay).setEase(positionTweenTyp).setOnComplete(onComplete);
}
```

Abbildung 38 Quellcode-Ausschnitt: Animationstool

Um die Einstellungen übersichtlich zu gestalten, können die einzelnen Transformationen im Inspector des Scripts mit dem jeweiligen *Toggle* ein- und ausgeblendet werden. Zudem werden die ausgeblendeten Transformationen bei der Ausführung der *Tweens* nicht mitberechnet und reduzieren somit die Rechenauslastung. Auf der rechten Seite der Abbildung sind die Parameter für die Animation der Position angegeben. Entnommen werden die Parameter über den Inspektor des Scripts (s. Abb. 38, rechte Seite). In dieser Form der Konfiguration definiert sich der *Tween* durch die Ausgansposition „positionOffset“, Animationsdauer „positionDuration“, Verzögerung „positionDelay“ und die Animationskurve „positionTweenType“. Das Tool *LeanTween* bietet dem Entwickler 32 verschiedene Animationskurven an (Vgl. Pixel (o.J.): LeanTweenType). Diese Kurven werden über dem *Enum* „LeanTweenType“ abgefragt und können im Inspektor im *Dropdown*-Menü ausgewählt werden. Wie oben beschrieben, liefert die Kurve einen vorgefertigten zeitlichen Verlauf der Animation. Somit lassen sich die Übergänge auf viele verschiedene Arten animieren. In der Abbildung ist die Animationskurve „easeInOutSine“ abgebildet (Vgl. Sitnik (o. J.): Übergangsfunktionen).

Im unteren Abschnitt der Abbildung ist der Funktionsaufruf für die Animation der Position dargestellt (s. Abb. 38). Zu Beginn der Funktion wird aus dem jeweiligen UI-Objekt die Ausgangsposition definiert. Dazu wird mit dem booleschen Wert „show“ zunächst ermittelt, ob das Objekt eingeblendet wird und somit auf seine *Offset*-Position „_positionOpp“ zugreift. Oder das Objekt wird ausgeblendet und seiner ursprünglichen Position „_positionOpp“ einnehmen soll. Mit dem Funktionsaufruf „move“ von der Klasse *LeanTween* wird anschließend der jeweilige *Tween* abgespielt. In diesem Aufruf werden auch die vorher festgelegten Variablen der Dauer, Verzögerung und Animationskurve festgelegt.

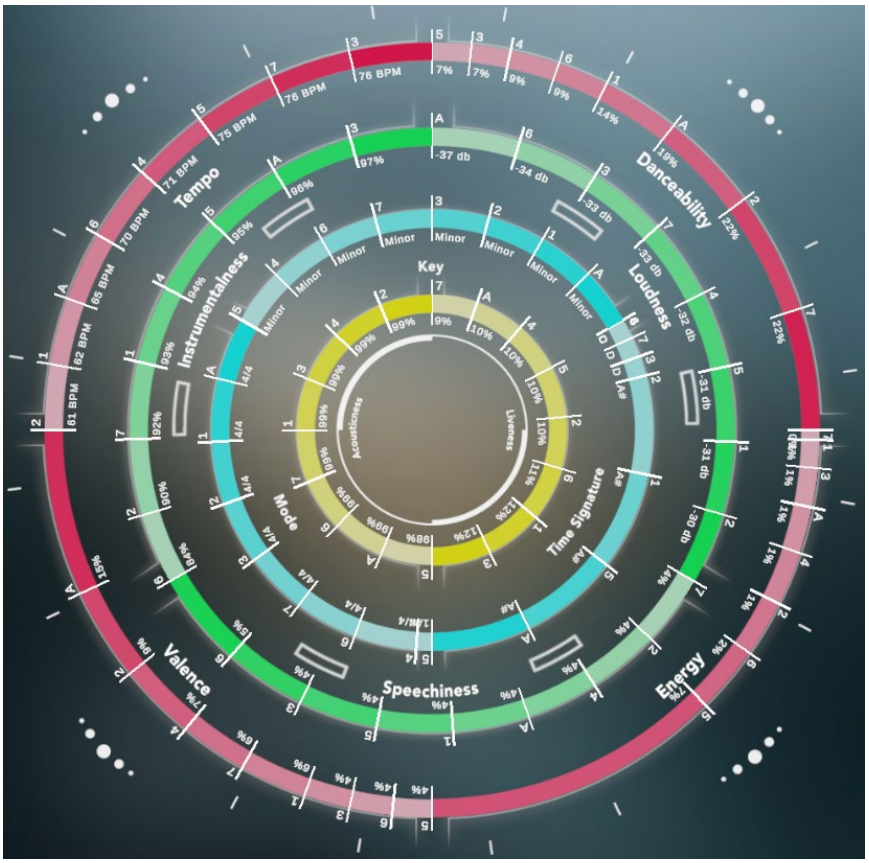


Abbildung 39 Ausschnitt aus der Anwendung: Dount-Diagramm

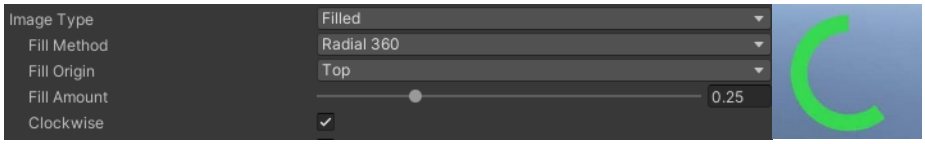


Abbildung 40 Image-Komponente: Fill-Methode

Wie in der Konzeptionierung beschrieben, dient das *Dount*-Diagramm zur Einordnung und Veranschaulichung der Audio-Features mehrerer Musiktitel (s. Abschnitt 4.2). Dazu können bis zu 15 Musiktitel dargestellt werden. Diese Visualisierung ist im Vergleich zu den anderen Visualisierungen hinsichtlich der Komplexität am höchsten. Je nach Kategorie sind die Datenpunkte in den jeweiligen Diagrammen in einzelne Segmente

eingeteilt. Zudem sind die Datenpunkte nach ihrer Ausprägung absteigend sortiert. Um die Datenpunkte in radialer Form auszurichten, bietet die Image-Komponente mit der *Fill*-Methode die notwendige Funktion (s. Abb. 39). Diese Funktion maskiert ein Teil des jeweiligen UI-Elements. Ausgehend von der „Fill-Method“ gibt der Parameter „Fill-Amount“ im Wertebereich zwischen 0 und 1 die Menge an, wie viel vom gesamten UI-Element aufgedeckt werden soll (s. Abb. 40). Dieser Parameter wird mit den Werten der Audio-Features für den jeweiligen Musiktitel gemappt. Mit diesem Ansatz können die einzelnen Datenpunkte in ihrer Länge dargestellt werden (Vgl. Sapio 2015: 50-52). Um eine detaillierte Angabe der Werte zu erhalten, werden die Daten der Audio-Features auf die Beschriftung unterhalb der Datenpunkte verknüpft.

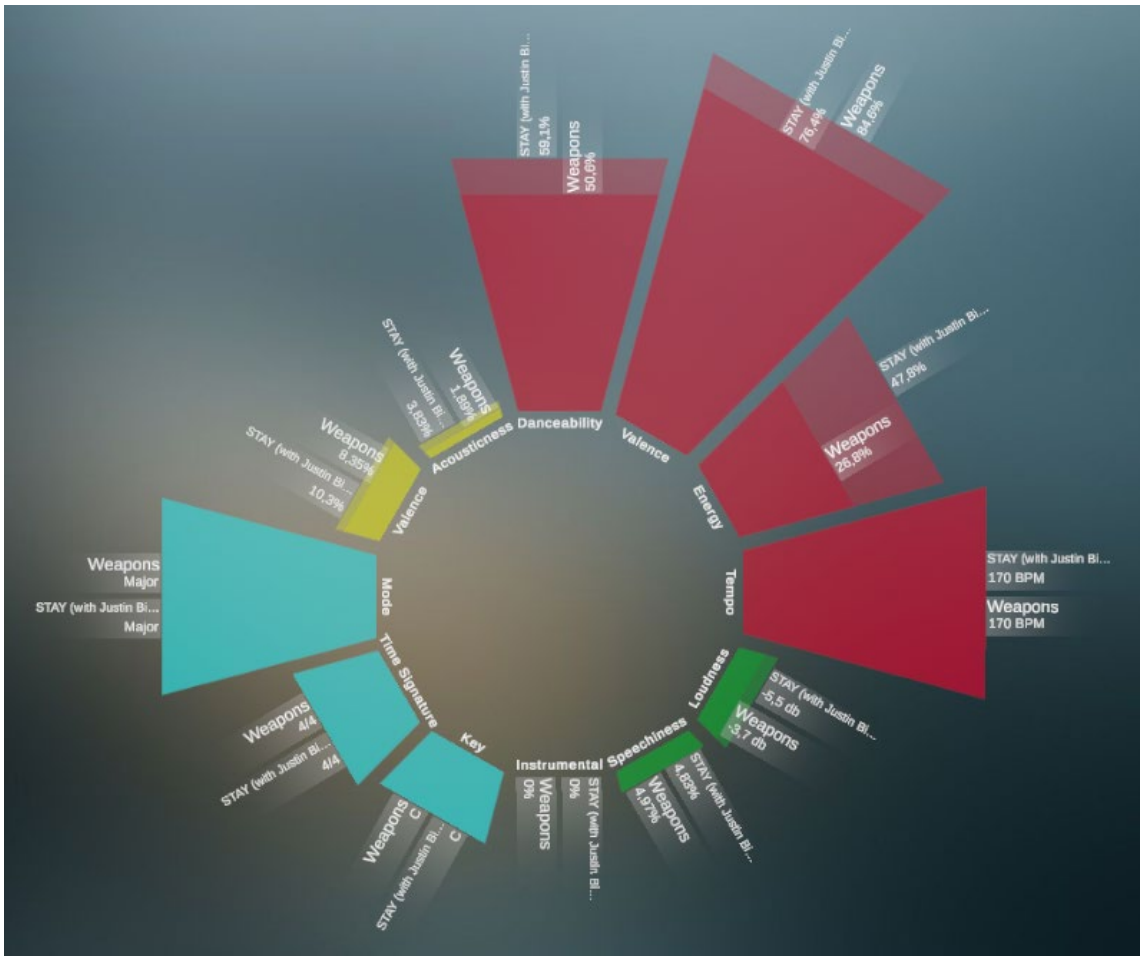


Abbildung 41 Ausschnitt aus der Anwendung: Nightingale-Diagramm

Wie in der Konzeptionierung der Arbeit beschrieben, liefert das *Nightingale*-Diagramm einen direkten Vergleich der Audio-Features zwischen dem zu untersuchenden und korrelierenden Musiktitel. Um die einzelnen Segmente des *Nightingale*-Diagramms darzustellen, werden - wie auch bei dem *Dount*-Diagramm - die *Image*-Features von Unity verwendet. Dazu wurden die 12 Segmente in Form eines Trapezes kreisförmig angeordnet (s. Abb. 41). Die farbliche Unterteilung richtet sich wie in der

Konzeptionierung angegeben, nach der kategorischen Einteilung der Audio-Features. Um die einzelnen Datenwerte der Audio-Features abzubilden, wurde die *Fill*-Methode in Vertikaler Ausrichtung verwendet (s. Abb. 40). Der Parameter „Fill Amount“ gibt die Höhe des jeweiligen Segments an. Im Quellcode werden diese Werte mit den Werten der Audio-Features gemappt. Zur Unterscheidung der beiden zu vergleichenden Musiktitel sind die Segmente leicht transparent dargestellt, wodurch die Überlagerung der beiden Vierecke deutlich wird. Die Beschriftung der Segmente oberhalb der Randwerte liefert eine detaillierte Angabe der Werte. Unterhalb der Segmente sind die Namen der einzelnen Audio-Features aufgelistet.

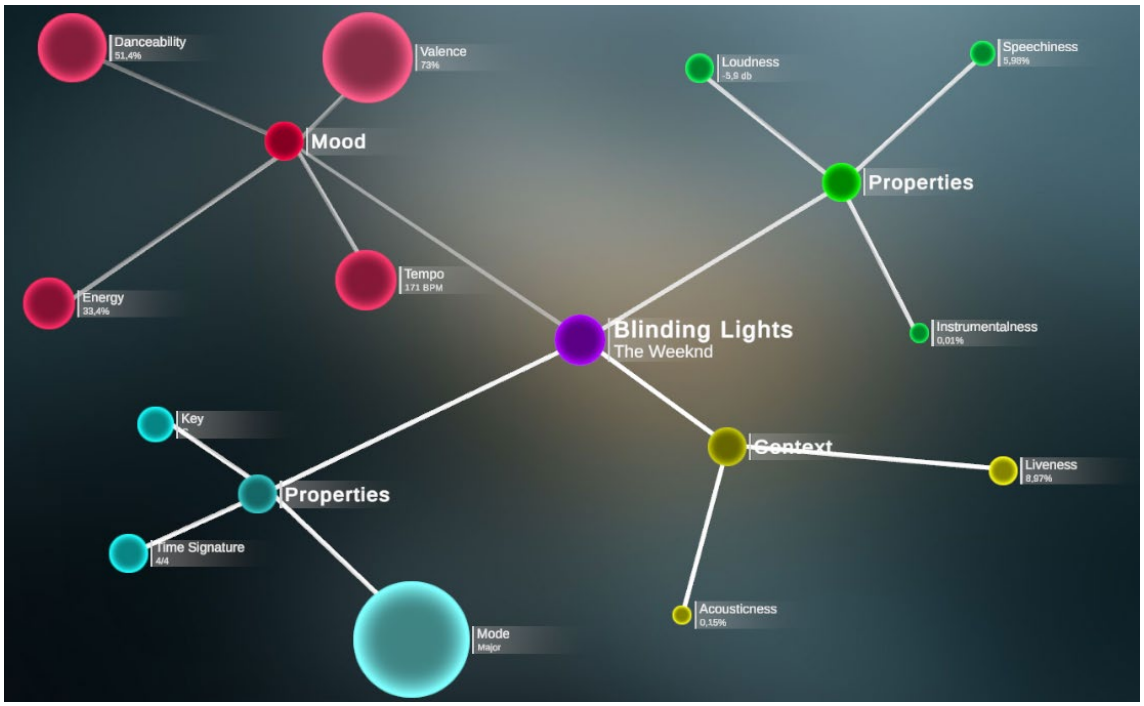


Abbildung 42 Ausschnitt aus der Anwendung: Netzwerkdiagramm

Das Netzwerkdiagramm bildet den letzten Abschnitt der Anwendung (s. Abb. 42). Wie in der Konzeptionierung dient diese Visualisierung zur Zusammenfassung und Beschreibung der einzelnen Audio-Features. Zudem veranschaulicht diese Visualisierung die Verbindung und Ausprägung der Audio-Features zueinander. Wie auch bei den anderen Visualisierungen, zeigt die Beschriftung den genauen Wert der jeweiligen Ausprägung. Diese Beschriftung ist seitlich zum Datenpunkt angeordnet. Um eine genaue Beschreibung für den jeweiligen Audio-Feature zu erhalten, kann der Nutzer die einzelnen Datenpunkte anklicken. Die entsprechende Beschreibung wird anschließend in der Infobox auf der rechten Seite der Anwendung aktualisiert. Diese Interaktion wird durch die Button-Komponente in den einzelnen Datenpunkten abgefragt. Sowohl der

Radius als auch die Farbsättigung der Datenpunkte wird durch den prozentualen Anteil der Audio-Features festgelegt.

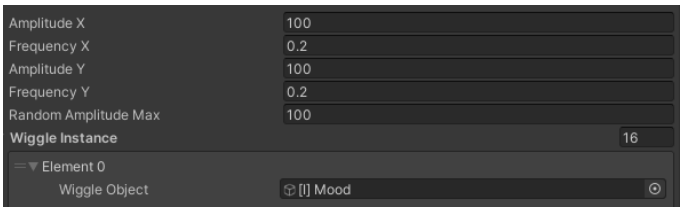


Abbildung 43 Wiggle-Komponente

Um diese Visualisierung interessanter zu gestalten, wurde eine *Wiggle*-Komponente entwickelt, in der die Positionen der einzelnen Datenpunkte moduliert werden (s. Abb. 43). In welcher Geschwindigkeit und in welcher Ausprägung diese Animationen berechnet werden, definiert sich durch die angegebene Frequenz und Amplitude. Damit die Datenpunkte eine unterschiedliche Bewegung aufweisen, wird die angegebene Amplitude zudem mit einem Zufallswert aufaddiert. Die einzelnen Verbindungen der Datenpunkte werden mit dem *Line-Renderer* erzeugt. Dazu werden die Positionen der einzelnen Datenpunkte in der *Update*-Methode im Quellcode ausgelesen und an den *Line-Renderer* übermittelt. Mit diesen Ansätzen wurde eine Visualisierung entwickelt, welche den Nutzer durch Bewegung und Interaktion animieren soll, weitere Aspekte über die einzelnen Audio-Features zu erfahren. Die Infobox „About Comperify“ informiert den Nutzer über musikalische Rekonstruktionen und liefert zudem ein Überblick über die Anwendung.

5.5 Zusammenfassung

Kapitel 5 erläutert die Hintergrundprozesse von Comperify unter Berücksichtigung der konzeptionellen Ansätze und technischen Aspekte. Es beschreibt die Einbindung der Spotify-API und die Umsetzung der Diagramme. Zudem wird der Implementierungsprozess der Anwendung reflektiert und die Vorgehensweise der technischen Ausführung begründet. Zu Beginn des Kapitels wurde der Autorisierungsvorgang der Spotify-API untersucht. Dazu bietet die API vier verschiedene Autorisierungsmodelle an. Diese Modelle besitzen unterschiedliche Zugangs-berechtigungen in Bezug der Benutzerinformationen und Datenbanken der Spotify-API. Nach dieser Untersuchung wurden die Vor- und Nachteile der Modelle mit den Anforderungen der Anwendung gegenübergestellt. Dabei bietet die Verwendung der Modelle „Client Credentials“ und „Authorization Code with PKCE“ die besten Voraussetzungen für den Autorisierungsvorgang der Anwendung.

Im zweiten Abschnitt dieses Kapitels wurden die Funktionsweisen und Hintergrundprozesse der einzelnen Komponenten beschrieben. Dazu wurde die Methodik des Suchalgorithmus hinsichtlich der *Recommendations* genauer untersucht. Recommendations bilden die Grundlage für Nutzer-Empfehlungen und somit die Suche nach musikalischen Rekonstruktionen. Ausgangspunkt des Analyseverfahrens ist der Seed-Track und die Werteangaben der jeweiligen Audio-Features. Neben der Erläuterung des Suchalgorithmus wurden diverse API-Anfragen genannt, die für die Kommunikation der Datenbanken der Spotify-API und den Komponenten zuständig sind.

Im letzten Abschnitt wurde die technische Umsetzung der Visualisierungen in *Comperify* erläutert. Dazu wurden die relevanten Features, wie beispielsweise die *Image*-Komponente und das UI-Animationstool genannt. Das Animationstool dient dazu, die Übergänge und Transformationen der Benutzeroberfläche ansprechender zu gestalten. Zudem überbrückt das Animationstool die Ladezeiten der jeweiligen URL-Anfrage. In der Image-Komponente wird die *Fill*-Methode angewandt, um die Mengen der einzelnen der Datenpunkte darzustellen. Die *Wiggle*-Komponente gestaltet zudem das Netzwerkdiagramm ansprechender, indem die Komponente die Positionen der einzelnen Datenpunkte moduliert. Mit der *Button*-Komponente können unterschiedliche Interaktionen ausgelöst werden. Dieses Kapitel spiegelt die technische Umsetzung der Anwendung wider und liefert den praktischen Teil dieser Arbeit. Im folgenden Kapitel werden die Ergebnisse und Erkenntnisse dieser Arbeit zusammengefasst und Bezug zur Fragestellung genommen.

6. Resümee

6.1 Fazit

Vorliegende Studie untersucht die Konzeptionierung und Implementierung einer interaktiven Datenvisualisierung. Diese Datenvisualisierung verfolgt das Ziel, die Recherche musikalischer Rekonstruktionen in Form einer Anwendung für jeden zugänglich zu machen. Das Motiv der Interaktion in der Anwendung ist die individuelle Erkundung und Erforschung dieser Thematik. Die Audio-Features der Spotify-API dienen dabei als Ausgangspunkt bei der Klassifizierung der musikalischen Rekonstruktionen. Wie der Name der Anwendung „Comperify“ vermuten lässt, soll der Zweck der Datenvisualisierung die Gegenüberstellung der klassifizierten Audio-Features sein. Die Methodik dieser Arbeit besteht darin, konzeptionelle und technische Ansätze für die Datenvisualisierung zu entwickeln und zu diskutieren. Um die Ziele dieser Untersuchung zu definieren, wurde zu Beginn folgende Fragestellung formuliert: „Können Rekonstruktionen musikalischer Werke mit geeigneten Audio-Features erkannt und visualisiert werden?“. In den folgenden Abschnitten werden die Ergebnisse dieser Arbeit zusammengefasst, um die zentrale Forschungsfrage zu beantworten.

Mit dem Ansatz der Music Emotion Recognition lassen sich Ähnlichkeiten zwischen Audio-Inhalten feststellen. Die Audio-Features der Spotify-API sind eine Form der Music Emotion Recognition und liefern einen Ansatz, Audio-Features zu klassifizieren. Die Rekonstruktion in der Musik definiert sich durch das Wiedergeben, Wiederherstellen oder Umgestalten bestehender musikalischer Werke. Diese Form gliedert sich in den Rubriken der Interpolation, Coverversion und Reproduktion. Die Datenvisualisierung bietet mit dem Fortschritt digitaler Systeme eine Vielzahl an Gestaltungsmöglichkeiten. Das Ziel von Data-Storytelling besteht darin, Sachinformationen effektiv zu vermitteln. Eine Visualisierung löst immer eine Veränderung beim Betrachter aus. Viele Visualisierungstools unterstützen eine dreidimensionale Darstellung der Datenvisualisierung. Diese räumliche Darstellung erweitert die Perspektive der Visualisierung, verfälscht jedoch unter Umständen die Vergleichbarkeit zwischen den Datensätzen. Webspezifische Anwendungen sind für den Betrachter zugänglicher als eine systemabhängige native Anwendung. Letztere erhöht mit ihren systemspezifischen Funktionen und Rechenleistung jedoch die Gestaltungsmöglichkeiten und immersiven Fluss der Datenvisualisierung. Die Unity-Engine bietet als Visualisierungstool ein breites Spektrum an Werkzeugen für die interaktive und visuelle Umsetzung einer Datenvisualisierung.

Das Trichterprinzip liefert einen konzeptionellen Ansatz für die Menüführung der Anwendung. Mit diesem Ansatz werden die Erfahrungen des Benutzers systematisch vom Gesamtkontext bis hin zu den Details der Thematik durchlaufen. Die Erstellung von Wireframes strukturiert die Eingabemöglichkeiten der Anwendung und liefert einen Entwurf, in dem die Erfahrungen des Nutzers berücksichtigt werden. Das Flowchart ist zu den Wireframes ein ergänzendes Mittel und fasst alle Programmabläufe zusammen. Die Datenvisualisierung ist in drei Abschnitte unterteilt. Durch diesen Ansatz wird der Nutzer schrittweise an die Themenbereiche der musikalischen Rekonstruktion herangeführt. Um die Datenbanken der Spotify-API für die Datenvisualisierung nutzen zu können, bietet Spotify vier Autorisierungsmodelle an. Das Autorisierungsmodell *Client Credentials* und *Authorization Code with PKCE* bietet in der Kombination die besten Voraussetzungen, sowohl die Anwendung für jeden zugänglich zu machen als auch die benutzerspezifischen Funktionen der Spotify API nutzen zu können. Die *Recommendations* der Spotify-API bilden die Grundlage der Suchanfrage korrelierender Musiktitel. Das Analyseverfahren der *Recommendations* bezieht sich auf die Faktoren der Ähnlichkeit, Vertrautheit und Entdeckung. Um Ladezeiten der URL-Anfragen zu überbrücken und die Datenvisualisierung interessanter zu gestalten, liefern *UI-Tweens* einen geeigneten Lösungsansatz. Die *Image*-Komponente in der Unity-Engine bietet mit der *Fill-Methode* eine passende Funktion, um die Datenpunkte der Diagramme entsprechend ihrer Werte zu skalieren.

Aus diesen Erkenntnissen lässt sich folgende Schlussfolgerung für die Fragestellung dieser Arbeit ziehen: Mit den Audio-Features sind musikalische Werke miteinander vergleichbar. Die Recherche nach musikalischen Rekonstruktionen können mit diesem Ansatz vorgenommen werden. Durch die Interaktionen und visuellen Aspekten der Datenvisualisierung werden diese Vergleiche für den Nutzer greifbar gemacht. Wie im Grundlagenteil dieser Arbeit beschrieben, löst jede Visualisierung eine Veränderung beim Betrachter aus. Mit der Datenvisualisierung *Comperify* erhält der Nutzer eine individuelle Erkundung verwandter Musiktitel. Die daraus erworbenen Erkenntnisse bilden die Veränderung, die beim Betrachter ausgelöst wird. Mit diesen Ergebnissen kann die Fragestellung, ob Rekonstruktionen musikalischer Werke mit geeigneten Audio-Features erkannt und visualisiert werden können, mit „Ja“ beantwortet werden.

6.2 Ausblick

Neben dieser Datenvisualisierung lassen sich weitere Lösungsansätze für die Visualisierung und Recherche musikalischer Rekonstruktionen finden. Wie im Grundlagenteil dieser Arbeit erörtert, erhöhen webspezifische Datenvisualisierungen die Zugänglichkeit solcher Darstellungen. Diese Form ist vom Betriebssystem unabhängig und kann von jedem browserfähigen System dargestellt werden. Die Umsetzung mit der JavaScript-Library D3.js liefert neben der nativen Anwendung einen weiteren Lösungsansatz, die Audio-Features der Spotify-API zu visualisieren. Für die Klassifizierung digitaler Audio-Inhalte lassen sich aus der *Music Emotion Recognition* und *Music Information Retrieval* zudem weitere Analyseverfahren außerhalb der *Recommendations* der Spotify-API finden. Das Musik- und Audio-Analyse Tool *Librosa* liefert beispielsweise die Möglichkeit, Audioinhalte nach ihren Chroma-Features zu analysieren und abzugleichen (Vgl. McFee 2015: 18). Zu den genannten Ansätzen lassen sich mit technologischem Fortschritt und gezielter Forschung viele weitere Lösungen finden, welche die Recherche musikalischer Rekonstruktionen für jeden zugänglich machen. Diese Arbeit liefert einen Ansatz, die Konzeption und Implementierung einer solchen interaktiven Datenvisualisierung zu verwirklichen.

Literaturverzeichnis

developer.spotify.com (o. J.): Authorization. Stockholm: Spotify AB. URL: https://developer.spotify.com/documentation/general/guides/authorization/ (Zugriff: 10.09.2022)

developer.spotify.com (o. J.): Authorization Code Flow. Stockholm: Spotify AB. URL: https://developer.spotify.com/documentation/general/guides/authorization/code-flow/ (Zugriff: 10.09.2022)

developer.spotify.com (o. J.): Authorization Scopes. Stockholm: Spotify AB. URL: https://developer.spotify.com/documentation/general/guides/authorization/scopes/ (Zugriff: 12.12.2022)

developer.spotify.com (o. J.): Client Credential Flow. Stockholm: Spotify AB. URL: https://developer.spotify.com/documentation/general/guides/authorization/client-credentials/ (Zugriff: 10.09.2022)

developer.spotify.com (o. J.): Get Track. Stockholm: Spotify AB. URL: https://developer.spotify.com/documentation/web-api/reference/#/operations/get-track (Zugriff: 10.09.2022)

developer.spotify.com (o. J.): Get Track´s Audio Features. Stockholm: Spotify AB. URL: https://developer.spotify.com/documentation/web-api/reference/#/operations/get-audio-features (Zugriff: 15.05.2022)

developer.spotify.com (o. J.): Get Recommendations. Stockholm: Spotify AB. URL: https://developer.spotify.com/documentation/web-api/reference/#/operations/get-recommendations (Zugriff: 10.09.2022)

developer.spotify.com (o. J.): Implicit Grant Flow. Stockholm: Spotify AB. URL: https://developer.spotify.com/documentation/general/guides/authorization/implicit-grant/ (Zugriff: 10.09.2022)

developer.spotify.com (o. J.): Web API. Stockholm: Spotify AB. URL: https://developer.spotify.com/documentation/web-api/ (Zugriff: 12.01.2022)

Dellinger, J. (o. J.): SpotifyAPI-NET. A Client for the Spotify Web API, written in C#/.NET. URL: https://johnnycrazy.github.io/SpotifyAPI-NET/ (Zugriff: 05.12.2022)

Díaz, D., García, J. M., García, F., & Pérez, J. (2015). Unity3D: A survey of game development tools. Journal of computer science and technology, 30(3), 421-432.

Cai, X./Efstathiou, K./Xie, X./ Wu, Y./Shi, Y/Yu, L. 2018: A Study of the Effect of Donut Chart Parameters on Proportion Estimation Accuracy. Published in: Computer Graphics Forum, 37(6), 300-312.

Donalek, C., Djorgovski, S.G., Cioc, A., Wang, A., Zhang, J., Lawler, E., Yeh, S., Mahabal, A., Graham, M., Drake, A. and Davidoff, S. 2014: Immersive and collaborative data visualization using virtual reality platforms. In 2014 IEEE International Conference on Big Data (Big Data) (pp. 609-614). IEEE.

Dudenredaktion (Hrsg.), (o.J.): Rekonstruktion, Duden online. URL: https://www.duden.de/rechtschreibung/Rekonstruktion (Zugriff: 09.06.2022)

Dykes B. 2016: Data storytelling: The essential data science skill everyone needs. Forbes Magazine.

hextantstudios.com 2022: Singleton in Unity. URL: https://hextantstudios.com/unity-singletons/ (Zugriff: 05.06.2023)

Frey, T. 2017: Grundlagen der Datenvisualisierung. Universität Freiburg

Gernitz, T. 2019: Der Einfluss von Musikstreamingplattformen auf das Hörverhalten: Über die zeitgenössische Musikentwicklung. GRIN Verlag.

Griffen, W., Catacora, D., Satterfield, S., Bullard, J., Terrill, J. 2015: Incorporation 3D.js Information Visualization into Immersive Virtual Environment. Gaithersburg: National Institute of Standards and Technology. In 2014 IEEE International Conference on Big Data (Big Data) (pp. 609-614). IEEE.

Gu, J., Zheng, Y., Mackin, S. 2018 June: Making sense: an innovative data visualization application utilized via mobile platform. In 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th

International Conference on Data Science and Systems (HPCC/SmartCity/DSS) (pp. 1105-1109). IEEE.

Guedes, L., Freitas, C. 2017: Exploring Music Rankings with Interactive Visualization. In Proceedings of the Symposium on Applied Computing (pp. 214-219). Porto Alegre: Universidade Federal do Rio Grande do Sul

Jacke, C. (et. al.) 2006: Kulturschutt: Über das Recycling von Theorien und Kulturen. transcript Verlag.

Jamro, M. 2018: C# Data Structures and Algorithms: Explore the possibilities of C# for developing a variety of efficient applications. Packt Publishing Ltd.

Joshi, A., Mathur, G. 2004: The inverted pyramid approach in user interface design for interactive information retrieval. In Proceedings of the EASY3 (CHI South India) Annual Conference.

Kim, S., Park, J., Seong, K., Cho, N., Min, J. and Hong, H. 2021: Music-Circles: Can Music Be Represented With Numbers? Süd Korea: Seoul National University. arXiv preprint arXiv:2102.13350.

Kunz, C. 2020: Musicalytics – Entdeckung von vielversprechenden Talenten auf Spotify durch Kombination von Informationsvisualisierung und Data Analytics. Mensch und Computer 2020-Usability Professionals.

Kromer, L. (et al.) 2016: Performance Comparison between Unity and D3.js for Cross-Platform Visualization on Mobile Devices. Austria: St. Poelten University of Applied Sciences. In FMT (pp. 47-52)

Liem, C.C., Hanjalic, A. 2006: Cover Song Retrieval: A Comparative Study of System Component Choices. Netherlands: Delft University of Technology. system, 3(1).

Magnus, P. D. 2022: A Philosophy of Cover Songs. Cambridge: Open Book Publisher

Mast, C. ed., 2018. *ABC des Journalismus: ein Handbuch* (Vol. 1). Herbert von Halem Verlag.

McFee, B., Raffel, C., Liang, D., Ellis, D.P., McVicar, M., Battenberg, E. and Nieto, O., 2015, July. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference* (Vol. 8, pp. 18-25).

Mehrotra, R. 2021: Algorithmic Balancing of Familiarity, Similarity, & Discovery in Music Recommendations. In Proceedings of the 30th ACM International Conference on Information & Knowledge Management (pp. 3996-4005).

Myatt, G. J./Johnson, W. P. 2021: Making Sense of Data 2. A Practical Guide to Data Visualization, Advanced Data Mining Methods and Applications. New Jersey: John Wiley & Sons

Nair, L. R./Shetty, S. D./Shetty, S. D. 2016: Interactive visual analytics on Big Data: Tableau vs D3. js. Journal of e-Learning and Knowledge Society, 12(4).

Navas, E. 2012: Remix Theorie. The Aesthetics of Sampling. Wien: Springer-Verlag

Neifer, T. (et al.) 2020: Data Storytelling als kritischer Erfolgsfaktor von Data Sience. Hochschule Bonn-Rhein-Sieg: Springer Verlag

Nazemi, K./Kaup, L./Burkhardt, D./Below, N. 2021: Datenvisualisierung. Praxishandbuch Forschungsdatenmanagement. Berlin: ResearchGate GmbH

Panda, R. (et al.) 2019: How does the Spotify API compare to the Music Emotion Recognition State-of-the-Art? Portugal: University of Coimbra. In Proceedings of the 18th Sound and Music Computing Conference (SMC 2021) (pp. 238-245). Axea sas/SMC Network.

Pasch, M./Bianchi-Berthouze, N./van Dijk, B./Nijholt, A. 2009: Immersion in Movement-Based Interaction. In International Conference on Intelligent Technologies for Interactive Entertainment (pp. 169-180). Springer, Berlin, Heidelberg.

Petschow, A. 2014: Die Coverversion und musikalischer Fortschritt? Eine Analyse der künstlerischen Bedeutung der Praktiken des Coverns in der populären Musik. Hamburg: Diplomica Verlag Gmbh

Pixel, Dented (o.J.): LeanTweenType. URL: https://dentedpixel.com/LeanTweenDocumentation/classes/LeanTweenType.html (Zugriff: 04.01.2023)

Sapio, F. 2015. Unity UI Cookbook. Packt Publishing Ltd.

Schröder, M./Hees, J./Bernardi, A./Ewert, D./Klotz, P./Stadtmüller, S. 2018: Simplified SPARQL REST API: CRUD on JSON Object Graphs via URI Paths. In *The Semantic Web: ESWC 2018 Satellite Events: ESWC 2018 Satellite Events, Heraklion, Crete, Greece, June 3-7, 2018, Revised Selected Papers 15* (pp. 40-45). Springer International Publishing.

Schürmer, A. 2018: Klingende Eklats. Skandale und Neue Musik. Bielefeld: Transcript Verlag

Seifert, C. 2015: *Spiele entwickeln mit Unity 5: 2D-und 3D-Games mit Unity und C# für Desktop, Web & Mobile*. Carl Hanser Verlag GmbH Co KG.

Serra, J./Gomez, E./Herrera, P. 2010: Audio cover song identification and similarity: background, approaches, evaluation, and beyond. Schweiz: In Advances in music information retrieval (pp. 307-332). Springer, Berlin, Heidelberg.

Severino, R. (o. J.): Histogram. Datavizcatalogue.com. URL: http://datavizcatalogue.com/methods/histogram.html (Zugriff: 10.09.2022).

Severino, R. (o. J.): Scatterplot. Datavizcatalogue.com. URL: http://datavizcatalogue.com/methods/scatterplot.html (Zugriff: 10.09.2022).

Severino, R. (o. J.): Heatmap. Datavizcatalogue.com. URL: http://datavizcatalogue.com/methods/heatmap.html (Zugriff: 10.09.2022).

Severino, R. (o. J.): Treemap. Datavizcatalogue.com. URL: http://datavizcatalogue.com/methods/treemap.html (Zugriff: 10.09.2022).

Severino, R. (o. J.): Nightingale Rose Chart. Datavizcatalogue.com. URL: https://datavizcatalogue.com/methods/nightingale_rose_chart.html (Zugriff: 22.01.2022).

Shabdin, N.I., Ya'acob, S. and Sjarif, N.N.A., 2020: Relationship types in visual analytics. In *Proceedings of the 2020 6th International Conference on Computer and Technology Applications* (pp. 1-6).

Sicat, R. (et al.) 2018: DXR: A Toolkit for Building Immersive Data Visualizations. Australien: Monash University. IEEE transactions on visualization and computer graphics, 25(1), pp.715-725.

Sitnik, A./Solovev I. (o. J.): Übergangsfunktionen. URL: https://easings.net/de (Zugriff: 04.01.2023)

Skau, D. and Kosara, R., 2016: Arcs, angles, or areas: Individual data encodings in pie and donut charts. In *Computer Graphics Forum* (Vol. 35, No. 3, pp. 121-130).

Yang, Q. (et. al.) 2016: Planning adaptive mobile experiences when wireframing. In Proceedings of the 2016 ACM Conference on Designing Interactive Systems (pp. 565-576).

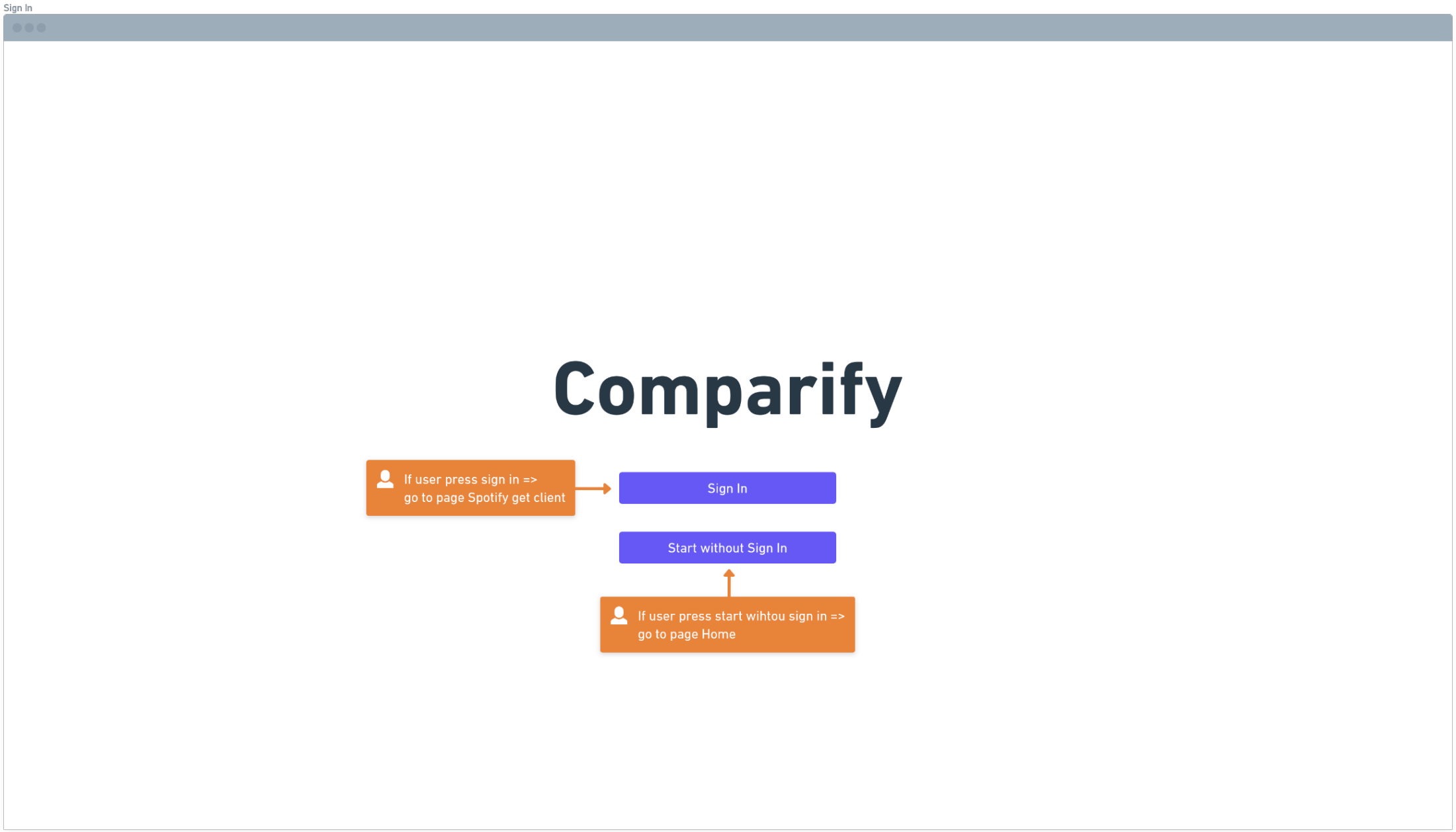


Abbildung 44 Wireframe: Login

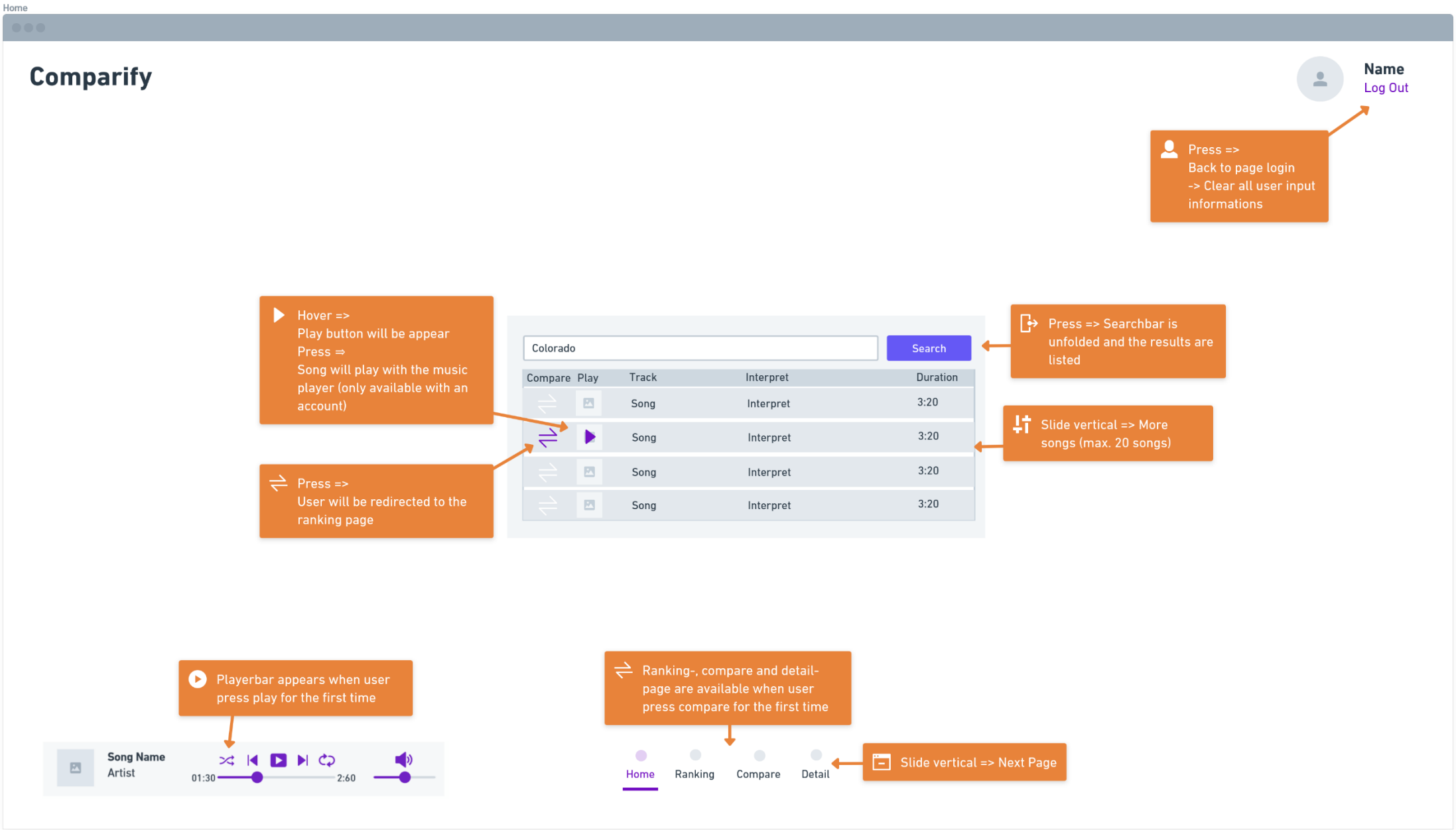


Abbildung 45 Wireframe: Home

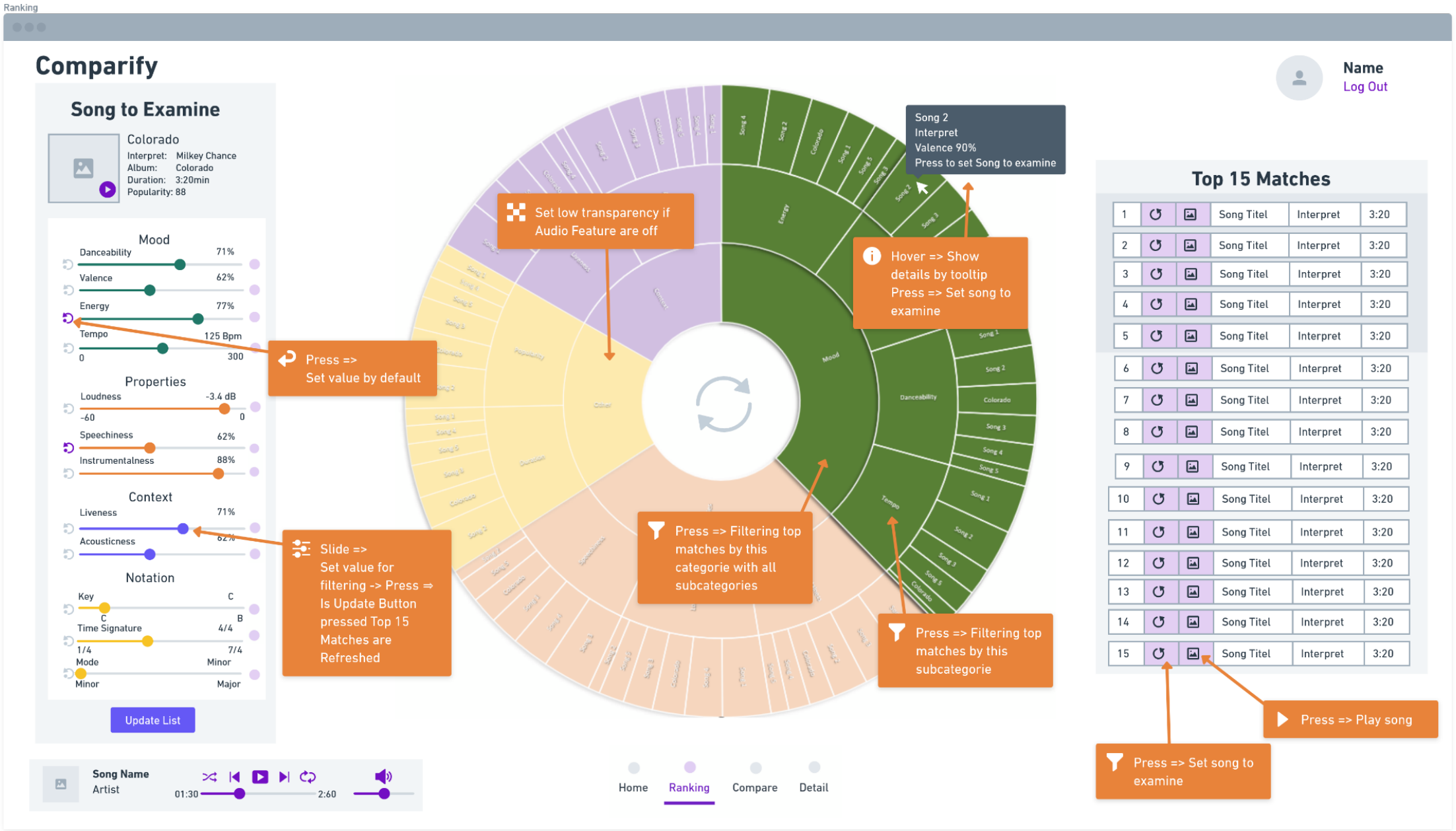


Abbildung 46 Wireframe: Ranking

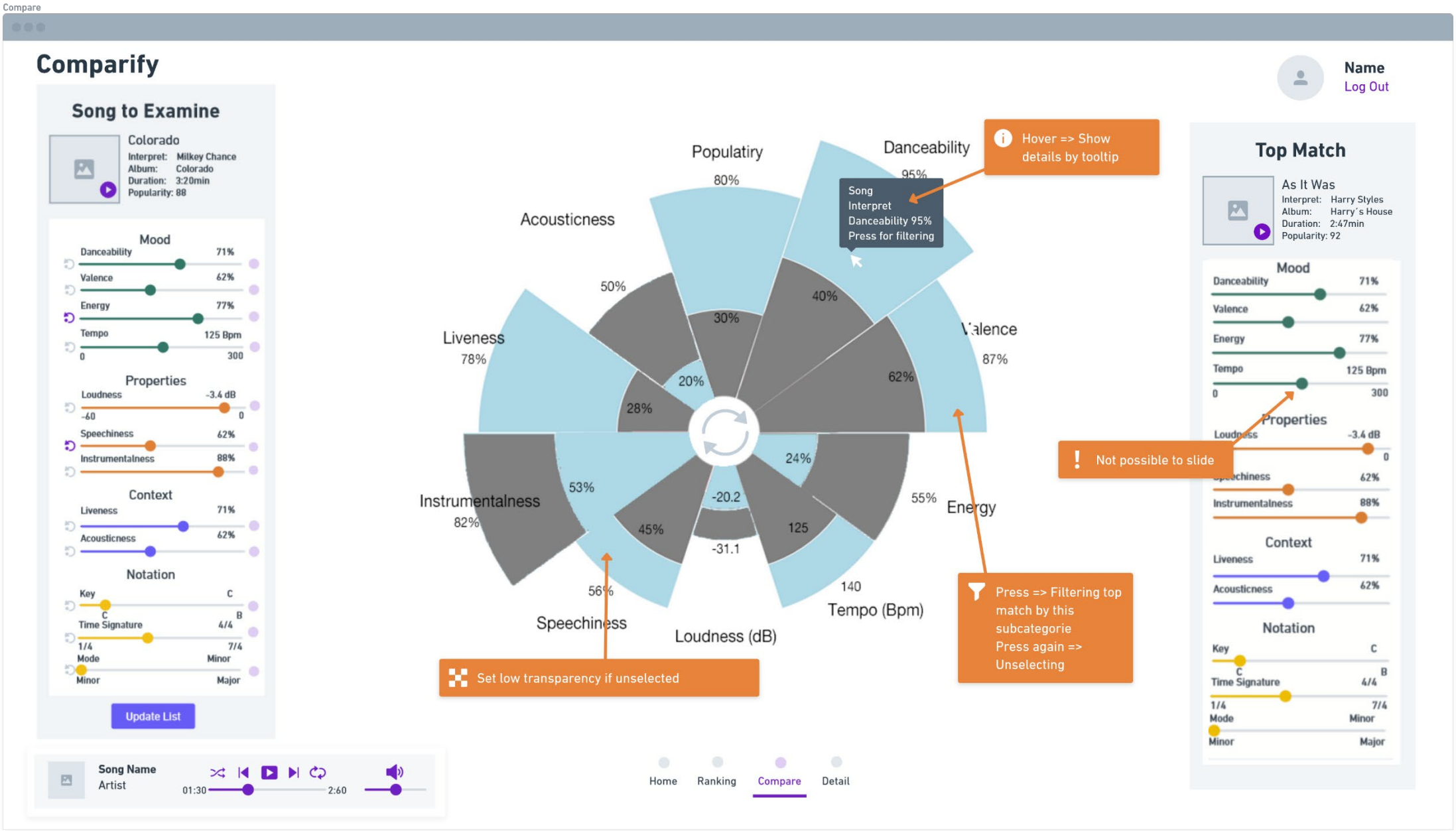


Abbildung 47 Wireframe: Compare

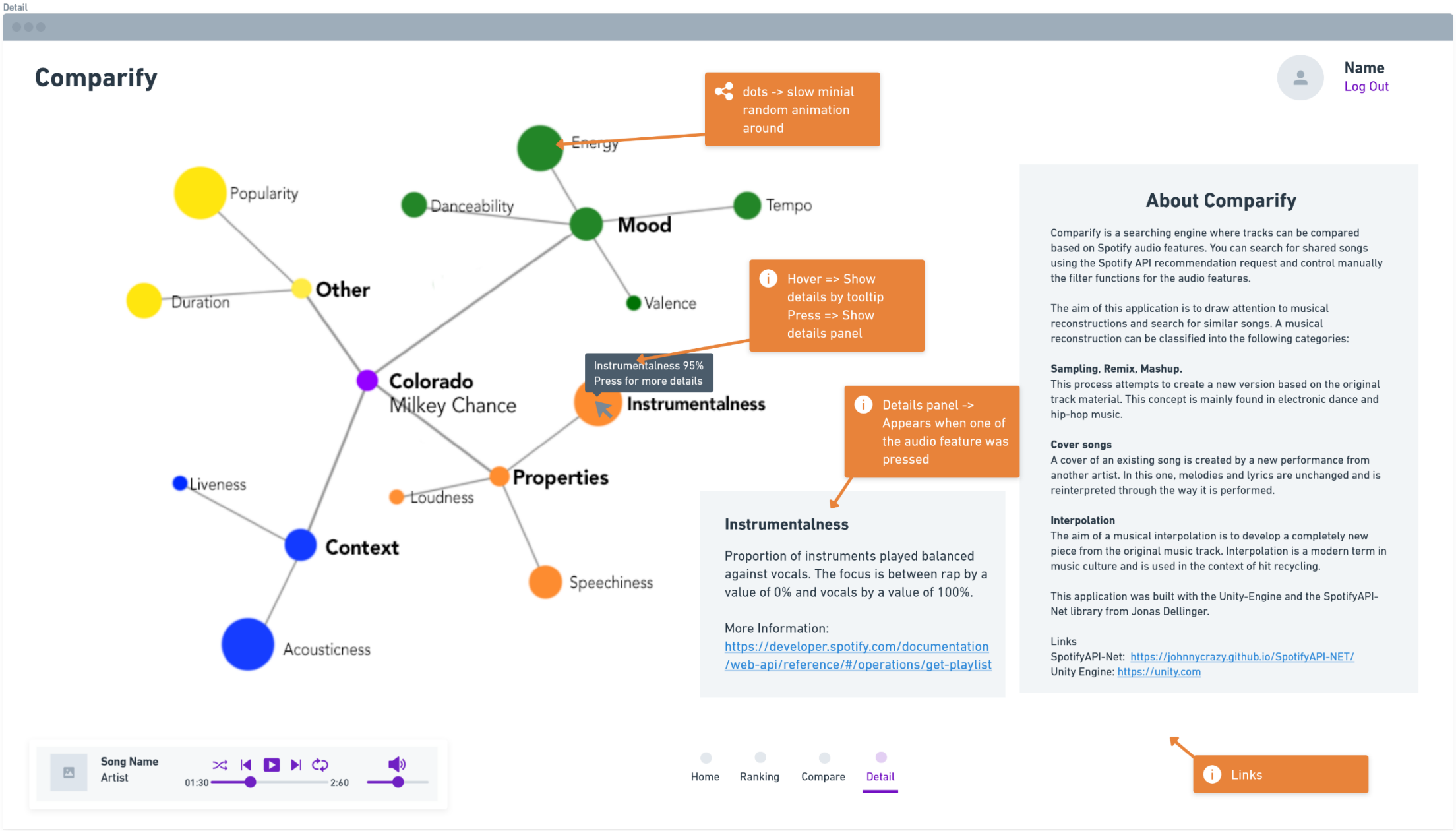


Abbildung 48 Wireframe: Detail

Eigenständigkeitserklärung

Hiermit versichere ich, Yannick Schmitz ehrenwörtlich, dass ich die vorliegende Arbeit mit dem Titel: „Konzeption und Implementierung einer interaktiven Datenvisualisierung zur Recherche von musikalischen Rekonstruktionen“ selbstständig und ohne fremde Hilfe verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken entnommen wurden, sind in jedem Fall unter Angabe der Quelle kenntlich gemacht. Die Arbeit ist noch nicht veröffentlicht oder in anderer Form als Prüfungsleistung vorgelegt worden.

_____

Bottrop, den 03.03.2023