Songify
Report – Project 1

Fontys Venlo University of Applied Sciences

By Jan Pinto Strohhäusl, Group DE-2, INF1-B

Rheurdt, 20.10.2021

Abstract

Dieser Bericht handelt von den Erarbeitungen der Gruppe DE-2 an „Songify“. Der Bericht wurde geschrieben, um den derzeitigen Stand der Gruppe an dem Projekt aufzuzeigen. Es werden die verschiedenen analytischen und designtechnischen Details der Gruppe aufgezeigt und ihre Zusammenhänge erklärt. Die Ergebnisse, die in diesem Bericht einfließen, entstanden aus verschiedensten Gruppenarbeits- und auch Einzelarbeitsphasen.

Die Gruppe erarbeitete die User von „Songify“, die verschiedenen Userrollen mit einem dazugehörigen Use-Case-Diagramm, verschiedenste Interaktionen, die ein User auf „Songify“ tätigen kann, ein Datenbankdesign im ER-Modell und im Rational Model mit den dazugehörigen Bedingungen und einem ersten Fensterdesign, wie die Website am Ende aussehen soll.

Inhaltsverzeichnis

1 Einleitung

Der vorliegende Bericht handelt von der Arbeit unserer Gruppe, der Gruppe DE-2, an dem Projekt 1. Der Bericht soll erklären, wie wir an „Songify“ gearbeitet haben.

Der Bericht existiert, da jetzt das Ende der Analyse- und Design-Phase erreicht wurde und wir uns in den nächsten Wochen der Implementation der Website widmen werden. Deshalb wird nun ein Bericht über die bisherigen Erfolge und Erarbeitungen aus unserer Gruppe erstellt. Dieser Bericht wird auch in eine Präsentation einfließen, die wir einzeln noch vorstellen werden.

Wie eben erwähnt, hat sich unsere Gruppe dazu entschlossen, das Thema „Songify“ zu nehmen. Dies hat den einfachen Grund, da wir uns selbst fordern wollten, um das Bestmöglichste aus uns herauszuholen. Deshalb haben wir „Songify“ genommen, da uns dies weitaus schwieriger, aber auch interessanter und spannender erschien als „Tickify“. Daher wird sich dieser Bericht um „Songify“ drehen, Inhalte zu „Tickify“ suchen Sie hier vergebens. In dem Bericht werde ich mich den Artefacts widmen, die wir erarbeitet haben, um eine Grundlage für die analytischen Komponenten für „Songify“ zu erstellen. Die Artefacts haben wir in Data-related-Artefacts, User-related-Artefacts und Task-related-Artefacts aufgeteilt, was uns eine bessere Aufteilung der Artefacts ermöglichte. Auch haben wir Datenbankschemas erstellt, die uns eine Implementation in SQL ermöglichen. Die richtige Implementation in eine Front-End Website werden wir erst in den nächsten Wochen vornehmen, da wir erst eine Grundlage für „Songify“ schaffen mussten.

Die Erarbeitung der Artefacts gestaltete sich einfach. Wir haben die Aufgaben, die wir jede Woche aufbekommen haben, in Gruppenarbeit gelöst, manchmal auch in Einzelarbeit. Da wir uns schon vor dem Projekt, nämlich in der Intro-Woche, kennengelernt hatten, war die Gruppenfindung einfach und zügig. Jeder wusste, was er bei den Aufgaben zu tun hatte, und falls es Probleme gab, konnten wir sie einfach ansprechen und lösen, gegebenenfalls mit Hilfe von Herrn Salz oder den Student Coaches, die während den Arbeitsphasen immer präsent sind und helfen. Da wir jede Woche die Aufgaben abgeben mussten, konnten wir sie immer wieder überarbeiten, weil Herr Salz uns Verbesserungsvorschläge zu den Aufgaben gab. Meist sind es Kleinigkeiten, die wir übersehen oder bei denen wir nicht genug nachgedacht haben, aber diese ließen sich schnell lösen.

Der Bericht ist in zwei Bereiche aufgeteilt: dem analytischen Teil und dem designtechnischen Teil.

In dem analytischen Teil werden die Artefacts in User-related-Artefacts, Task-related-Artefacts und Data-related-Artefacts aufgeteilt und näher beleuchtet. In den User-related-Artefacts wird es um die User-groups im Großen und Ganzen gehen. Die Task-related-Artefacts handeln von den Use-Cases und von den Scenarios. Als letztes kommen die Data-related-Artefacts mit den Constraints und dem ER-Modell, denn dies bietet einen fließenden Übergang zu dem designtechnischen Teil des Berichtes.

In dem designtechnischen Teil werden die Datenbankschemas aufgezeigt und erklärt. Außerdem widmen wir uns hier den ersten Fensterdesigns, die wir uns für „Songify“ überlegten. Es ist vielleicht noch nicht so, wie es dann im Endprodukt aussehen wird, aber danach werden wir uns orientieren. Auch werden hier das Moodboard vorgestellt, welches wir gemeinsam erstellten.

Am Ende des Berichtes steht eine Zusammenfassung der gesamten Arbeit an „Songify“ und eine Bewertung meinerseits. Auch wird hier ein Ausblick auf die kommenden Arbeiten gegeben, da wir uns gerade an der Hälfte zum fertigen Produkt befinden. Dazu wird hier eine persönliche Beurteilung bezüglich der Gruppen und deren Arbeitsmoral und -atmosphäre gefällt.

2 Analyse

Dieses Kapitel wird sich mit den analytischen Aufgaben von Songify beschäftigen. Die designtechnischen Inhalte finden Sie im nächsten Kapitel.

2.1 User-related-Artefacts

Zuerst widmen wir uns den User-related-Artefacts, um eine Grundlage zu schaffen, welche Personen „Songify“ nutzen sollen, welche Beziehungen sie zu der Website haben und welche Forderungen sie an „Songify“ haben.

2.1.1 User-Groups

Bei den User-Groups geht es um die Benutzer, von denen wir glauben, dass diese „Songify“ nutzen können. Im Folgenden werden diese und ihre User-Wünsche beschrieben.

Wir haben die Nutzerrollen in drei Kategorien eingeteilt: Benutzer, Künstler und Provider. Die Benutzer nutzen die Website, bedeutet diese Nutzergruppe kann Musik hören, Playlisten erstellen, nach Musik suchen, ein Song kaufen bzw. ein Abo abschließen etc. Die Künstler können das Gleiche machen wie die Benutzer, jedoch können sie auch Musik / Podcasts hochladen und die Einnahmen sehen, die sie pro Song machen. Die Provider sind die Personen, die die Website erstellen und betreiben.

Bei den Benutzern gibt es auch verschiedene Unterkategorien, welche die Website nutzen können. Wir haben die Benutzer je nach Alter kategorisiert, um ein möglichst breites Feld an Benutzer abzudecken. Wir bieten die Website für Kinder, Jugendliche, Erwachsene, Ältere Personen und auch Familien an. Die Personas, also die Hintergründe zu den Benutzergruppen, beinhalten neben Namen und Hintergrund der Person auch die Fähigkeiten der Person, was die Nutzung des Computers angeht. Diese Fähigkeiten bestehen aus „Erfahrung mit Computern“, „Visuelle Fähigkeiten“, „Onlineshopping- / Kauferfahrungen“ und „Bedienanspruch“.

Für den Bereich „Kinder“ haben wir uns den Persona „Isabelle Schmidt“ ausgedacht. Sie ist 9 Jahre alt und besucht die Grundschule. Da sie gerade ein neues Smartphone von ihren Eltern bekommen hat, tastet sie sich langsam heran. Mit Freunden chatten klappt schon, daher testet sie sich ans Musik hören. Daher ist sie auf „Songify“ gekommen. Mit Songify möchte sie alleine oder mit ihren Freundinnen Musik hören, oder sich auch mal einen Podcast ihres Lieblingsyoutubers anhören. Kinder wie sie vermeiden kompliziert geschriebene und kleingedruckte Texte und bevorzugt eher einfachere, größere Texte mit Bildern.

Das Gleiche gilt für den Bereich „Ältere Personen“. Hier lautet unser Persona „Gertrude Wollseif“. Sie ist eine 81-jährige Rentnerin und hat vor Kurzen einen Rechner von ihrem Sohn bereitgestellt bekommen. Da sie ziemlich alt ist, möchte sie immer auf dem neusten Stand bleiben, auch was Musik angeht, weshalb sie auf Songify gestoßen ist. Wie bei Kindern sollten die Texte und Symbole für ältere Leute, die im Allgemeinen unerfahren mit Computern sind, groß und einfach zu verstehen sein.

Bei „Jugendliche“ und „Erwachsene“ sieht es schon anders aus. „Max Mustermann“, der Persona aus dem Bereich „Jugendliche“, ist ein typischer 17-Jähriger, der einen hohen Bedienungsanspruch hat. Dasselbe gilt für „Torsten Eisen“, ein 41-Jähirger Kaufmann, der eine hohe Computer Erfahrung

aufweist, da er täglich mit ihnen zu tun hat, jedoch keinen besonderen Bedienungsanspruch hat. Beide möchten Songify nutzen, um sich mit Musik entspannen zu können. Da Jugendliche und Erwachsene sehr häufig im Internet unterwegs sind und dadurch geübt mit Webseiten umgehen können, können hier die Texte in normaler Schrift geschrieben sein und müssen nicht zwangsläufig einfach geschrieben sein. Das steht natürlich zum Kontrast mit den „Kinder“ und „Ältere Personen“, die es eher einfacher haben möchten. Später wird erklärt, wie wir dies lösen möchten (3.2 Fensterdesign). Außerdem wird in diesen Personengruppen Wert auf Individualisierung gelegt, da sie gerne Webseiten oder Applikationen nach ihren Wünschen umändern möchten.

Da wir „Songify“ auch für Familien anbieten, gestaltete sich das Schreiben des Personas für „Familie“ schwieriger, da hier mehrere Personen aufeinandertreffen, die unterschiedliche Erfahrungen mit Computer und deren Bedienung haben. Wir haben uns dann überlegt, dass wir einzelne kurze Personas zu den jeweiligen Familienmitgliedern entwerfen und die Fähigkeiten am Ende zusammenführen. Daraus folgte, dass wir eine Familie namens „Familie Schuhmacher“ erstellten, die aus 46-Jährige Vater, 47-Jähriger Mutter, 16-Jährige Tochter und 3-Jährigen Sohn besteht. Alle würden „Songify“ nutzen, um sich mit Musik, Podcasts oder auch Hörbücher zu entspannen. Es resultierte, dass die Familie mittlere Erfahrung mit Computern, hohe Erfahrung im Bereich „Onlineshopping / Kauferfahrung“ und einen geringen Bedienungsanspruch haben. Daher, dass wir auch für Familien die Website anbieten, haben wir uns überlegt, dass wir ein Familien-Abonnement erstellen, damit es nicht zu teuer für die einzelnen Familienmitglieder ist.

„DJ Moodymann“ ist unser Persona für den Bereich „Künstler“. Mit „DJ Moodymann“ haben wir alle Forderungen, die ein Musikproduzent an eine Musik-Website haben könnte, zusammengefasst. Er ist, wie andere Musiker, darauf aus, seine Musik so einfach wie möglich an seine Hörer zu bringen. Er nutzt das Internet gelegentlich, in letzter Zeit aber vermehrt, da die Pandemie keine Konzerte ermöglichte. Daher hat er den Anspruch an einer Musik-Website wie „Songify“, dass diese von allen Leuten einfach zu benutzen ist und dass man die Einnahmen sehen kann, die er mit seinen Songs machte, um zu erkennen, welche Songs von ihm erfolgreicher sind als andere Songs von ihm.

Die „Provider“ sind, wie erwähnt, die Personen, die die Website erstellen und verwalten. In dem Persona ist der „Provider“ eine einzelne Person namens „Sebastian Kurz“. Da er eine Website betreibt, hat er exzellente Erfahrung mit dem Umgang von Computer. Da er auch seine Website benutzerfreundlich gestalten will, hat er einen hohen Anspruch an sich selbst, diese Website so nutzerfreundlich wie möglich zu gestalten.

2.2 Task-related-Artefacts

In den Task-related-Artefacts geht es um allgemeine und spezifizierte Aufgaben, die User an „Songify“ stellen. Die Aufgaben an „Songify“ werden im Näheren beschrieben.

2.2.1 Use-Case-Diagramm

In diesem Abschnitt wenden wir uns dem Use-Case-Diagramm zu. Das Use-Case-Diagramm zeigt die Forderungen der einzelnen Nutzergruppen und deren Überschneidungen mit anderen Nutzergruppen.

Auf der nächsten Seite können Sie in Abbildung 1 das von uns erstellte Use-Case-Diagramm sehen.

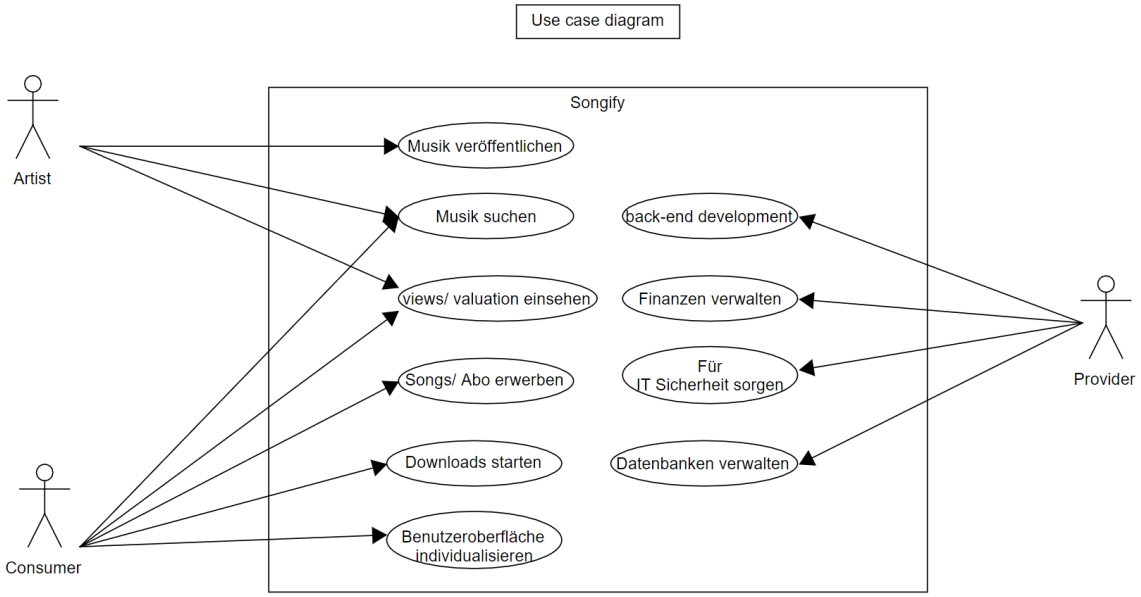


Abb. 1: Use-Case-Diagramm von „Songify"

Auf der linken Seite sind die Kunden von „Songify" zu sehen, während rechts der Provider ist. In dem Kasten, welcher „Songify" darstellt, sind verschiedene Blasen mit verschiedenen Inhalten zu sehen. In den Blasen stehen die Szenarien, die auf „Songify" passieren können, wenn sie aufgerufen werden. Man kann in dem Use-Case-Diagramm erkennen, welche User-Role auf welche Szenarien zugreifen kann. Der „Customer" kann auf alle Szenarien zugreifen, mit denen er Musik hören kann. Der „Artist" kann auf alle Szenarien zugreifen, die mit dem Verkauf seiner Musik zu tun haben. Zuletzt kann der „Provider" auf alle Szenarien zugreifen, die mit der Implementation der Website zu tun haben. Man erkennt, dass der „Artist" und der „Costumer" bei zwei Szenarien den gleichen Zugriff haben. Für den „Customer" ist es interessant, zu sehen, wie oft ein Song gehört wurde, um zu erkennen, wie gut der Song ist und wie erfolgreich der „Artist" ist. Der „Artist" kann sowie der „Customer" nach Musik suchen, um eigene oder auch andere Musik zu hören und um zu vergleichen, ob seine Musik wettbewerbsfähig ist. Wenn ein „Customer" sich ein Abo kauft, wird es auf seinem Account vermerkt, und der Nutzer kann sich jede Musik ohne jegliche Begrenzung und ohne Werbung anhören. Näheres zu den Szenarios steht im nächsten Abschnitt.

2.2.2 Scenarios

In diesem Abschnitt werden die Szenarien aus dem Use-Case-Diagramm aus dem vorherigen Abschnitt näher beleuchtet. Hier wird auch auf mögliche Umsetzungen der Szenarien eingegangen.

Damit eine Website vernünftig funktioniert, müssen vor der Implementationsphase verschiedene Aufgaben durchgegangen werden, die vom User an die Website gestellt werden können. Diese Aufgaben nennt man Szenarien. Der „Provider" muss vor der Implementationsphase diese Szenarien durchgehen, damit jede Interaktion zwischen dem Nutzer und der Website funktioniert und damit es nicht passiert, dass Interaktionen ins Nichts führen.

Wir haben, wie im Use-Case-Diagramm die Szenarien zu erkennen ist, die Szenarios in die User-groups aufgeteilt. Wie auch schon erwähnt, haben der „Artist" und der „Customer" auf zwei Szenarios den

gleichen Zugriff. Im Folgenden wird darauf eingegangen, wie die Szenarien funktionieren und es werden ein paar Beispiele gezeigt.

Wir haben uns überlegt, wie eine Aktion, beispielsweise wenn ein „Artist“ einen Song hochladen möchte, aussehen könnte. Dabei müssen wir erstmal sicherstellen, dass der „Artist“ einen Knopf hat, auf den er klicken kann, um weiter zum Interface geleitet zu werden, wo die Songs hochgeladen werden können. Wenn der „Artist“ dann auf diesen Knopf klickt, wird er dann zu einer Seite weitergeleitet, auf der er dann den Song hochladen kann. Außerdem werden dem „Artist“ noch andere Optionen angezeigt, mit denen der Song beschrieben wird, wie den Namen des Songs, das Genre, eine kurze Beschreibung (Produzent, Musiklabel, …), den Songtext und ein Foto für den Song. Wenn alles fertig eingestellt ist, kann der „Artist“ auf „Upload“ klicken, um den Song auf „Songify“ zu veröffentlichen. Wenn der Song hochgeladen wurde, wird vom System ein Link generiert, mit dem der „Artist“ oder auch „Customer“ den Song teilen können, sodass der Song an Bekanntheit gewinnen kann.

Es gibt auch andere Szenarien, die weniger kompliziert sind. Nehmen wir das Beispiel „Abo / Song erwerben“. Wenn ein „Customer“ ein Abo abschließt, dann wird sein Account als „Premium-Nutzer“ in der Datenbank abgespeichert. Um wiederum eine Bestellung abzuschließen, dann müssen die Songs, die der „Customer“ erwerben will, in einem Einkaufswagen gelegt werden. Erst danach können diese gekauft werden. Wenn dann diese Bestellung abgeschlossen wird, dann werden die Songs, die erworben wurden, dem Nutzer freigeschaltet, damit dieser die Songs hören kann.

Dies waren zwei Beispiele, wie die Szenarien umgesetzt werden könnten. Natürlich haben wir aus dem Use-Case-Diagramm jedes Szenario genau beschrieben, jedoch würde das hier den Rahmen sprengen, jedes einzelne zu erklären.

2.3 Data-related-Artefacts

In diesem Abschnitt werden wir uns den letzten Teil der Artefacts widmen, den Data-related-Artefacts. Hier wird auf das entstandene ER-Modell sowie den Constraints eingegangen.

2.3.1 ER-Modell

Dieser Abschnitt handelt von dem ER-Modell von „Songify“, welches wir entwickelten. Das ER-Modell bildet die Grundlage für das Relational Model, um dann später die Datenbank für „Songify“ zu erstellen.

Dieses ER-Modell wurde mit Absprache von Herrn Salz, Frau Holz und den Student Coaches mehrmals überarbeitet, sodass wir nun ein fertiges ER-Modell haben. Vor dem Erstellen des ER-Modells haben wir ein Data-Dictionary erstellt, in welchem alle wichtigen Entitäten und deren Attribute stehen. Daraus haben wir dann ein ER-Modell gefertigt. Das ER-Modell sehen Sie auf der nächsten Seite in Abbildung 2.

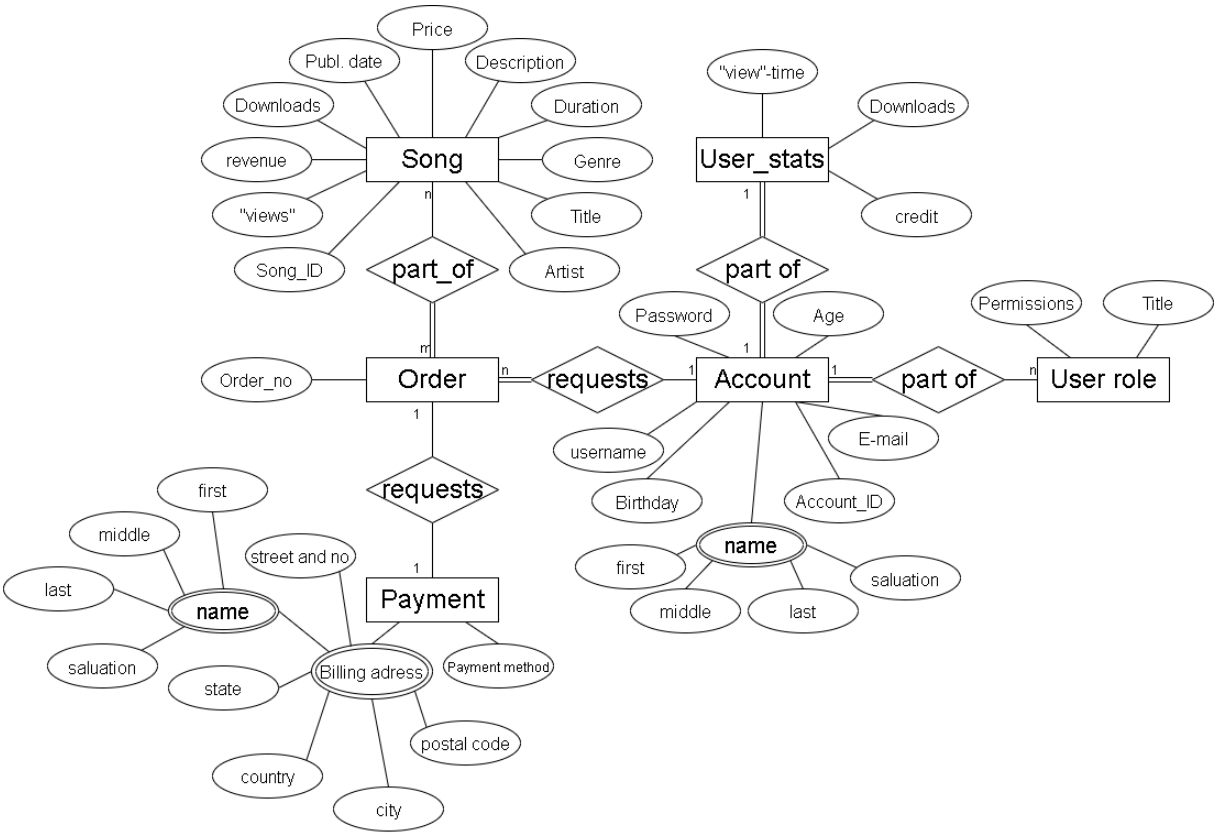


Abb. 2: ER-Modell zu „Songify“

Man kann das ER-Modell in drei Abschnitte eingliedern: „Account“, „Song“ und „Order“.

Der „Account“ ist, wie der Name schon sagt, der Account des Users. Hier werden alle Informationen zum Account abgespeichert, zum Beispiel Name, Geburtstag, Passwort etc. Außerdem haben wir noch zwei weitere Entitäten an dem Account angehängt, und zwar „User_stats“ und „User_role“. „User_stats“ werden automatisch mit jedem Account erstellt und verfolgen die Statistiken von jedem Account, also wie viele Songs der User heruntergeladen hat oder wie lange dieser insgesamt Musik gehört hat. Deshalb ist es eine 1-1 Beziehung zwischen „Account“ und „User_stats. Darüber hinaus gibt es eine weitere Entität, die „User_role“. Hier wird abgespeichert, welche User-Role der Account hat, also ob der Account ein „Customer“, ein „Artist“ oder ein „Provider“ ist. Außerdem werden in der Entität die Berechtigungen abgespeichert, die dem Account dann zur Verfügung stehen. Die Verbindung wurde extra so gewählt, da die „User_role“ auch ohne einem Account existieren kann, da dem Account beim Erstellen eine Rolle von bereits existierenden Rollen gegeben wird.

In dem Bereich „Song“ werden alle Attribute abgespeichert, die ein Song beinhalten kann. Wir haben uns endschieden, keine direkte Verbindung zwischen „Song“ und „Account“ zu erstellen, da wir den Song für den Account freischalten wollen, falls der Account ein Song kauft. Dies kann man über die ID des Songs machen. So kann man sich eine direkte Verbindung sparen.

Die einzige Verbindung, die „Song“ und „Account“ haben, ist die „Order“ Entität. Über diese Entität werden Käufe getätigt und die Songs werden dem Account dann freigeschaltet. „Order“ haben wir als schwache Entität in das ER-Modell eingetragen, da „Song“ und „Account“ auch ohne „Order“ existieren können und „Order“ nur auf Abruf existiert. Jede Bestellung hat eine zugehörige Bestellungsnummer, welche wir in „Order_no“ abspeichern. Außerdem wird bei jeder Bestellung die Bezahlmethode abgefragt, die in der „Payment“ Entität steht. Wir haben die „Payment“ Entität unabhängig vom

Account gemacht, da wir uns überlegten, dass es auch sein kann, dass es besser ist, wenn die Kaufmethode nicht zwischengespeichert ist. Wenn zum Beispiel ein Account von einem Kind geführt wird, dann muss es so immer seine Eltern fragen, bevor es sich einen weiteren Song kaufen kann, weil die Bankdaten bei jedem Kauf erneut eingetragen werden müssen. Das gibt die Sicherheit, dass es nicht zum Missbrauch von Bankdaten kommt, wenn jemand anderes die Bestellung für jemanden bezahlt.

2.3.2 Constraints

Attribute aus z. B. einem ER-Modell können ohne für sie vorgegebene Bedingungen, genannt „Constraints“, nicht funktionieren. Die wichtigsten Constraints für die Attribute werden hier beschrieben.

Im Allgemeinen beschreiben unsere Constraints, welche Attribute zu welchen Entitäten gehören. Dazu wurde auch aufgeschrieben, welches Format diese haben müssen und wie sie mit anderen Entitäten verbunden sind. Wir haben zum Beispiel als Constraint für „age“ aus „Account“: „A birthday must have the format dd-mm-yyyy“, weil es zu Unstimmigkeiten mit dem amerikanischen Format „mm-dd-yyyy“ kommen könnte. Auch haben wir bei „Order“ den Constraint „Every Order must have an order number”, damit die Bestellungen zugeordnet werden können. Mit dem Constraint „Every Order must contain at least one product” wird sichergestellt, dass wenn eine Order abgeschlossen wird, dass diese mindestens ein Song beinhaltet, da ansonsten keine Bestellung abgeschlossen werden kann, weil keine Songs gekauft werden.

Dies waren grundsätzliche Beispiele, was für wir Constraints erstellt haben. Alle aufzuzählen würde den Bericht nur weiter in die Länge ziehen.

Dies ist das Ende von dem analytischen Teil des Berichts. Auf der nächsten Seite beginnt der designtechnische Teil.

3 Design

Nun widmen wir uns den designtechnischen Teil des Berichtes. Hier wird das vorläufige Fensterdesign anhand des Moodboards erklärt, die Navigation zwischen den Seiten und am Ende kommt noch einmal kurz das Relational Model und einem Beispiel aus unserem SQL-Code.

3.1 Moodboard

Erst widmen wir uns kurz dem Moodboard, um eine Grundlage für das Fensterdesign zu schaffen. Anhand des Moodboardes haben wir das Fensterdesign entwickelt. Vor der Erstellung des Moodboardes hat jeder sein eigenes Moodboard erstellt und am Ende haben wir es zu einem gemeinsamen Moodboard zusammengeführt.

Im Appendix in Abbildung 3 sehen Sie das von uns erstelle Moodboard.

Wie man im Moodboard erkennt, haben wir uns geeinigt, dass „Songify“ in einem dunkleren Farbton mit orangenen Akzenten sein soll. Außerdem haben wir im Moodboard alle musikalischen Genres abgeglichen, die auch in „Songify“ dabei sein sollen (z. B. Rock, Pop, Acoustic, A cappella, …). Auch soll das Moodboard zeigen, dass man sich mit „Songify“ entspannen kann.

3.2 Fensterdesign

Anhand des Moodboardes aus dem vorherigen Abschnitt haben wir ein vorläufiges Fensterdesign entwickelt. Falls es sich im späteren Verlauf als unpraktisch zeigt, werden wir dies natürlich ändern.

Sie können im Appendix in Abbildung 4 das Fensterdesign sehen. Dies soll die Hauptseite aus der Sicht des „Artist“ sein.

Man erkennt oben links das Symbol von „Songify“, oben in der Mitte die Suchleiste, um Songs zu suchen und oben rechts den „Upload“ Knopf und den Account Knopf. Links sieht man Kategorien, die man aufklappen kann. Der untere graue Balken soll ein Platzhalter sein, denn dort soll der aktuell gespielte Song oder Podcast angezeigt werden. Die Kreise in der Mitte sind auch Platzhalter. Dort sollen im Endprodukt verschiedenste Kategorien angezeigt werden, z. B. kürzlich gehörte Playlists, Musik- oder Podcastempfehlungen, neue Veröffentlichungen von Künstlern etc.

Da es nur den „Artist“ mittleren Alters widerspiegelt, sind andere Gruppen in dem Beispiel außen vor. Wenn zum Beispiel bei Alter eine hohe oder niedrige Zahl eingegeben wird, wird die Seite anders aussehen, nämlich auf diese Alter zugeschnitten, z. B. ist die Schrift größer, der Kontrast ist höher etc. Dies werden wir in der Implementation mitbedenken.

3.3 Navigation

Da wir nun mit dem Fensterdesign vertraut sind, können wir ein Blick auf die Navigation zwischen den Seiten werden. Da dies ein größeres Bild ist, werden Sie das Bild im Appendix in Abbildung 5 finden.

Man erkennt die einzelnen Seiten mit ihren Designs. Jedes Bild ist eine Seite mit einer dazugehörigen Überschrift. Die Überschrift soll erklären, was im Design zu erkennen ist (z. B. „Payment“ ist die

Bezahlseite einer Bestellung). Dazu haben wir kenntlich gemacht, welche Seiten nur vom Künstler und Admin gesichtet werden können. Außerdem sieht man, dass die Hauptseite von allen Seiten durch Klicken auf das Logo oben links zu erreichen ist.

Die Struktur der Navigation ähnelt einem Fluss, denn von einer Seite kann man auf die nächste Seite kommen, bis man wieder zur Quelle, der Hauptseite, zurückmuss. Die Designs sind noch nicht vollkommen ausgereift und können sich während der Implementationsphase noch ändern. Wir werden jedoch versuchen, die Designs so zu übernehmen.

3.4 Relational Model

Kurz vor dem Ende werden wir uns dem Relational Model widmen. Dieses Modell entstand aus dem ER-Model und ist, wie bei dem ER-Model, in Absprache mit Herrn Salz, Frau Holz und den Student Coaches ständig überarbeitet und verbessert worden.

Das Relational Model können Sie im Appendix in Abbildung 6 sehen.

Dort ist zu erkennen, dass alle Entitäten aus dem ER-Modell eine eigene Relation sind. Jede dieser Relation hat eine ID, die als Primary Key gilt, denn damit wird auf andere Relationen verwiesen. Wenn auf eine andere Relation verwiesen wird, wird diese verwiesene ID dort in der Relation als Foreign Key aufgenommen. Das hat uns ermöglicht, alle im ER-Modell erstellten Relationen korrekt darzustellen.

Auch konnten wir dadurch sicherstellen, dass alle Funktionen so korrekt funktionieren. Wenn zum Beispiel eine Bestellung aufgenommen wird, dann muss eine Referenz vom Song, vom Account und von der Bezahlung kommen. Würde die Referenz von zum Beispiel dem Account fehlen, kann die Bestellung keinem Account zugeordnet werden. Wie wir im ER-Modell und in den Constraints jedoch festgelegt haben, muss eine Bestellung immer zu einem Account zugeordnet werden. Dadurch würde die Bestellung nicht richtig funktionieren. Deshalb haben wir die Referenzen dort eingebaut.

3.5 SQL

Dieser Teil wird sich kurz mit SQL befassen. Der Teil wird nicht sonderlich lang, weil wir noch an der Implementation der Datenbank arbeiten müssen.

Ein Beispiel unseres SQL-Codes (für die Tabelle „Account“) finden Sie im Appendix unter den Abbildungen.

Der SQL-Code orientiert sich an den Constraints und dem Relational Model. Wir haben die Constraints in den Attributen der Tabelle übernommen (z. B. alle Attribute not null, username größte Länge 50 Zeichen etc.) und die Primary Keys schon hinzugefügt. Die Foreign Keys haben wir auskommentiert bei der Implementation der Tabelle mit reingeschrieben, damit wir später nicht vergessen, wenn alle Tabellen implementiert sind, die Foreign Keys hinzuzufügen. Da aber im Code noch ein Teil fehlt, wird der Rest der Datenbank in den nächsten Wochen vor der Implementation der Website fertiggestellt.

Dies ist das Ende des designtechnischen Teils. Im nächsten Kapitel finden Sie die Zusammenfassung.

4 Zusammenfassung

Zusammenfassend lässt sich sagen, dass wir bisher auf einem guten Stand sind.

Wir haben Nutzergruppen identifiziert, sind auf deren Wünsche eingegangen und haben spezielle Anforderungen an die Seite gestellt, damit die Nutzergruppe keine Probleme beim Nutzen der Website hat. Auch die jeweiligen Use-Cases mit den Szenarien haben wir erarbeitet, sodass wir schon eine Grundlage an Aufgaben an die Website haben. Außerdem haben wir mit Hilfe von außen ein fertiges ER-Modell mit zugehörigen Constraints erarbeitet.

Das Fensterdesign wurde mit Hilfe eines vorher entwickelten Moodboardes konstruiert, woran wir uns während der Implementationsphase orientieren können. Das Navigationsdiagramm ist auch schon fertig. Außerdem haben wir schon grobe Fensterdesigns in dem Navigationsdiagramm erstellt, jedoch können sie noch in der Implementationsphase überarbeitet werden. Auch wurde aus dem ER-Modell ein fertiges Relational Model erstellt, aus dem wiederum SQL-Code entstand, auch wenn wir an dem Code noch arbeiten müssen.

Ich persönlich denke, dass wir als Team eine sehr gute Leistung erbracht haben. Es herrscht eine angenehme Arbeitsatmosphäre, alle sind gut gelaunt, Anmerkungen werden nicht ignoriert oder belächelt, sondern diskutiert und wenn etwas unklar ist, dann ist Nachfragen überhaupt kein Problem. Ich finde es gut, dass wir schon so viel geschafft haben, und würde mich freuen, wenn wir auch in Zukunft so weiterarbeiten wie bisher.

In den nächsten Wochen werden wir uns der Implementation widmen. Dazu werden wir an einen Crash Course in CSS, HTML-5 und PHP teilnehmen, um auf die Implementation der Website vorbereitet zu sein. Am Ende der Implementationsphase wird dann hoffentlich ein ansehbares Produkt namens „Songify“ rauskommen, auf den man auch echte Musik hören kann.

5 Appendix

Abbildung 3: Gruppenmoodboard

Abbildung 4: Hauptseite aus Sicht des Künstlers

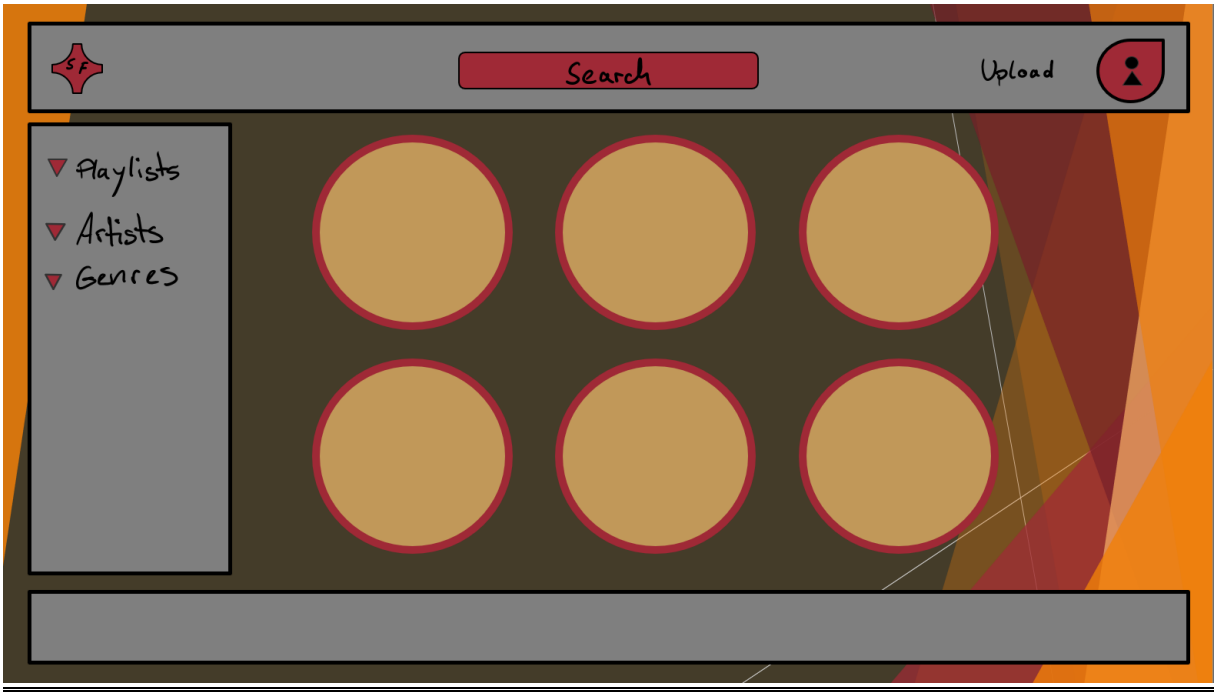


Abbildung 5: Navigationsdiagramm

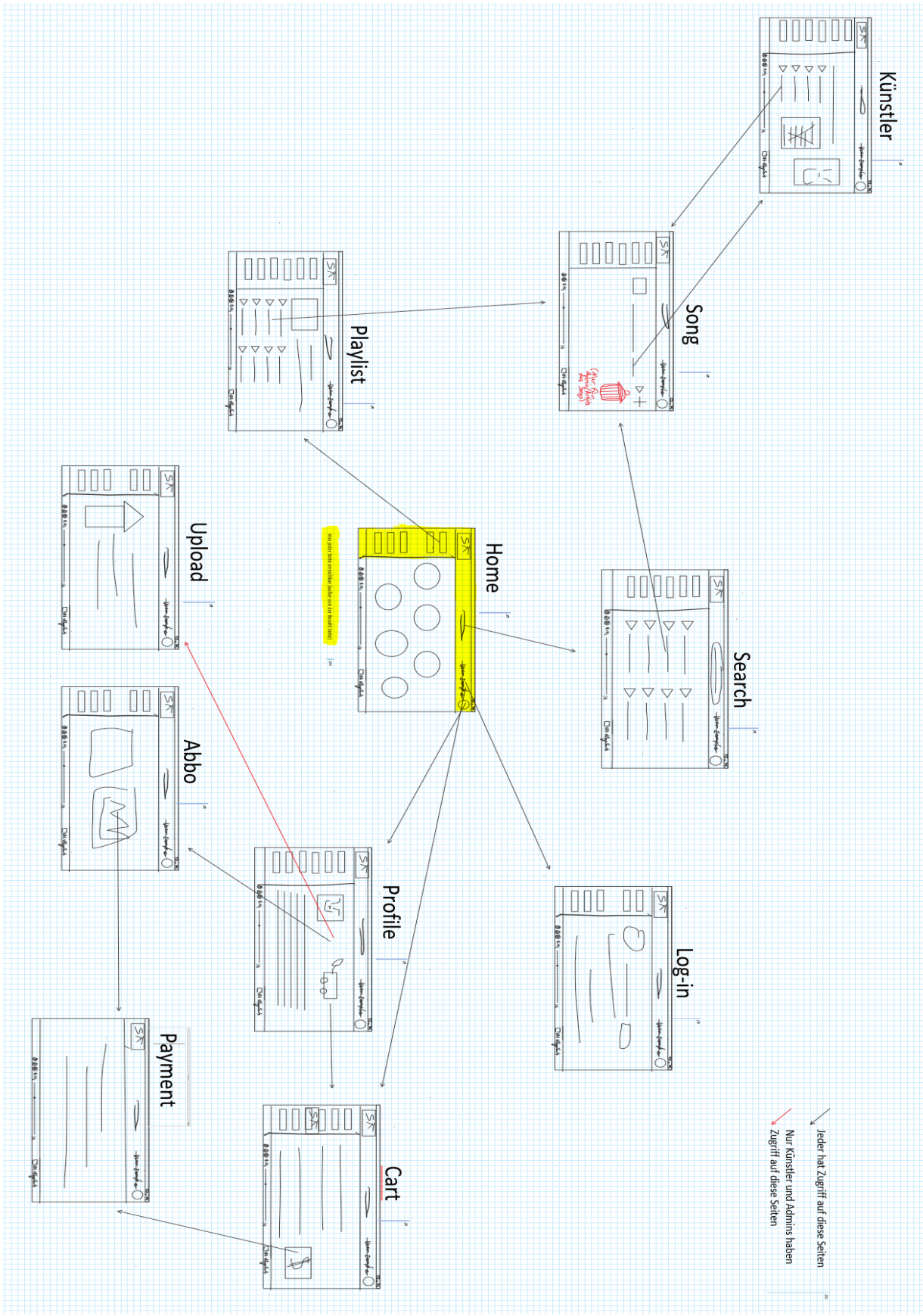
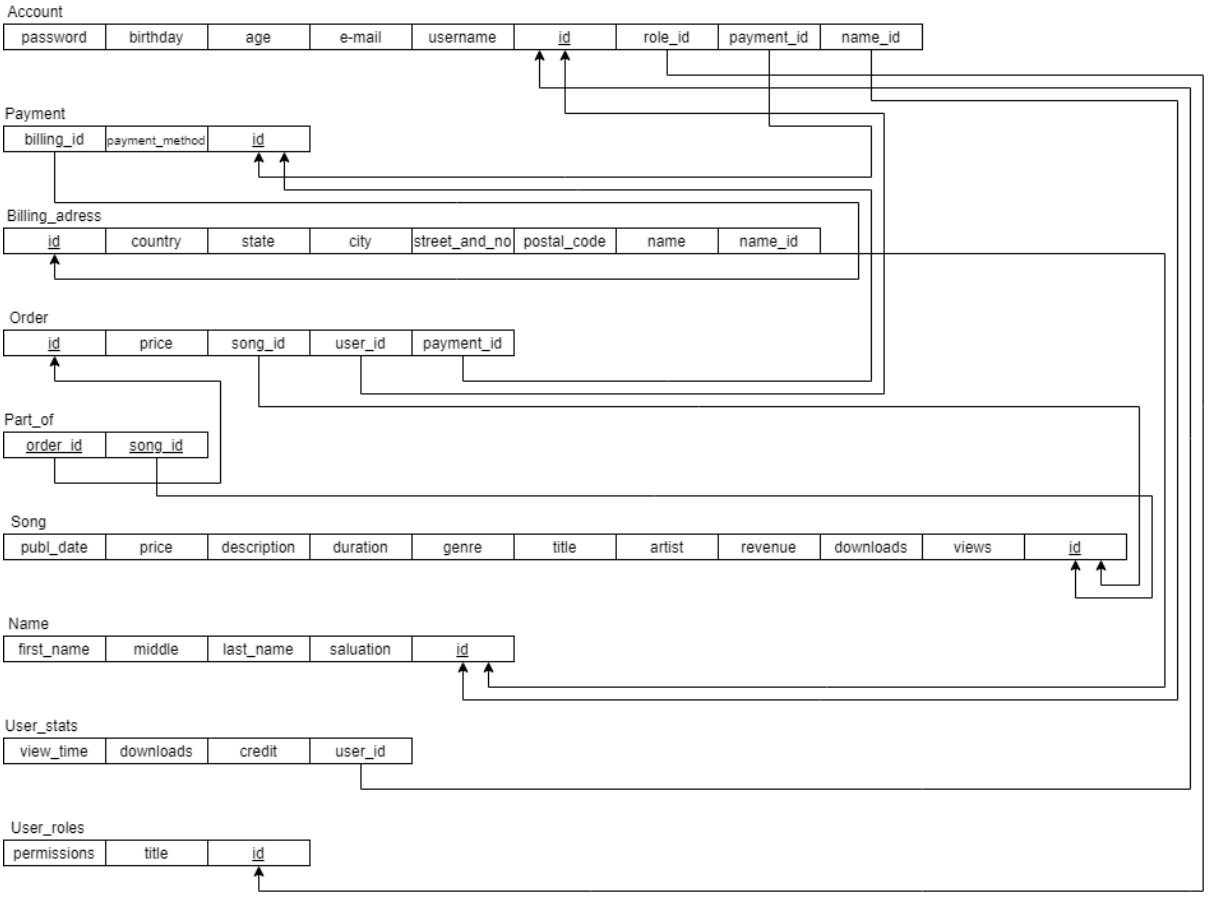


Abbildung 6: Relational Model



SQL-Code:

```
create table Account (
        password varchar (250) not null,
        birthday date not null,
        personal_age int not null,
        e_mail varchar not null,
        username varchar (35) not null,
        account_id int not null,
        role_id int not null,
        payment_id int not null,
        name_id int not null,
        primary key (account_id)
        -- foreign key (role_id) references User_roles (id)
        -- foreign key (payment_id) references Payment (id)
        -- foreign key (name_id) references Name (id));
```