

# **Relaxoon**

## **DIPLOMARBEIT**

verfasst im Rahmen der

**Reife- und Diplomprüfung**

an der

**Höheren Abteilung für IT-Medientechnik**

Eingereicht von:

Abdulrahman Al Sabagh  
Moritz Eder

Betreuer:

Thomas Stütz

Projektpartner:

Solvistas, Macolution

Leonding, April 2024

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Die Arbeit wurde bisher in gleicher oder ähnlicher Weise keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Die vorliegende Diplomarbeit ist mit dem elektronisch übermittelten Textdokument identisch.

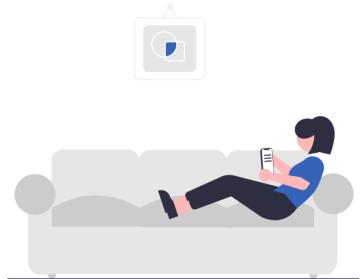
Leonding, April 2024

Abdulrahman Al Sabagh & Moritz Eder

# Abstract

In order to give stressed people a simple way to relax, a stress management app has been developed.

Relaxoon is a play on the words 'relax' and 'soon'. Relaxoon offers relaxation exercises in the form of videos, texts, articles and more, which are presented in a simple and clear way.



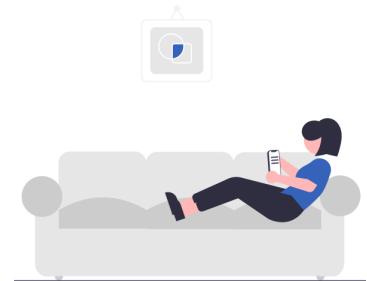
In addition to implementing the front end in React Native and the back end with the content management system Strapi, the deployment using Kubernetes is described in detail.

# Zusammenfassung

Damit gestresste Personen eine simple Möglichkeit haben, sich zu entspannen, wurde eine App zur Stressbewältigung entwickelt.

Relaxoon ist ein Wortspiel aus den englischen Wörtern 'relax' und 'soon'. Relaxoon bietet Entspannungsübungen in Form von Videos, Texten, Artikeln und mehr, die in einfacher und übersichtlicher Weise dargestellt sind.

Neben der Implementierung des Frontends in React Native und des Backends mit dem Content-Management-System Strapi wird auf das Deployment mittels Kubernetes im Detail eingegangen.



# Inhaltsverzeichnis

<b>1 Ausgangssituation</b>	<b>1</b>
<b>2 Problemstellung</b>	<b>2</b>
<b>3 Ziele</b>	<b>3</b>
3.1 Projektziele . . . . .	4
<b>4 Aufgabenstellung</b>	<b>7</b>
4.1 Use-Case-Diagramm (UCD) . . . . .	8
<b>5 Marktanalyse</b>	<b>9</b>
5.1 Vorgehensweise . . . . .	9
5.2 Analyse . . . . .	10
<b>6 Projektanforderungen</b>	<b>15</b>
6.1 Funktionale Anforderungen: . . . . .	15
6.2 Nicht funktionale Anforderungen: . . . . .	15
<b>7 Technologien</b>	<b>17</b>
7.1 Strapi . . . . .	17
7.2 Firebase App Distribution . . . . .	17
<b>8 Systemarchitektur</b>	<b>18</b>
8.1 Komponentendiagramm . . . . .	18
<b>9 Entwurfsentscheidungen</b>	<b>20</b>
9.1 React Native . . . . .	20
9.2 Strapi . . . . .	22
9.3 Component Library . . . . .	24

<b>10 Design</b>	<b>25</b>
10.1 Mockups . . . . .	25
10.2 Tutorial . . . . .	26
10.3 Login- und Registrierungsformular . . . . .	26
10.4 Home-Screen . . . . .	27
10.5 Kategorien . . . . .	28
10.6 Suchleiste . . . . .	29
10.7 Media Player . . . . .	30
10.8 Favoriten . . . . .	31
10.9 Einstellungen . . . . .	32
10.10 Light/Dark Mode . . . . .	33
10.11 Logo . . . . .	34
<b>11 Implementierung</b>	<b>35</b>
11.1 Entity Relationship Diagram (ERD) . . . . .	35
11.2 Medias und Articles . . . . .	35
11.3 TutorialPage . . . . .	36
11.4 Suche und Filterungen . . . . .	36
11.5 State Management . . . . .	38
11.6 Dark/Light Mode . . . . .	38
11.7 Help und Info Screen . . . . .	39
11.8 Authentifizierung . . . . .	39
11.9 Validierung . . . . .	41
11.10 Deployment . . . . .	41
11.11 Hochladen einer Mobileapp auf Firebase App Distribution . . . . .	47
<b>12 Ausgewählte Aspekte und Probleme</b>	<b>51</b>
12.1 Probleme mit localhost und https . . . . .	51
12.2 Inkompatible Libraries beim Build-Prozess . . . . .	51
12.3 Probleme mit Thumbnails . . . . .	52
12.4 Dauer von Videos . . . . .	53
12.5 Probleme mit useEffect und React-navigation Library . . . . .	53
12.6 Relaxoon Plakat . . . . .	54
<b>13 Resümee</b>	<b>56</b>

<b>14 Anhang</b>	<b>VI</b>
14.1 Prompt-Verlauf . . . . .	VI
<b>Literaturverzeichnis</b>	<b>VIII</b>
<b>Abbildungsverzeichnis</b>	<b>XI</b>
<b>Tabellenverzeichnis</b>	<b>XII</b>
<b>Quellcodeverzeichnis</b>	<b>XIII</b>

# 1 Ausgangssituation

Heutzutage gibt es Stresssituationen, die dazu führen können, dass man manchmal den Überblick über die eigenen Aufgaben verliert. Um auch auf lange Zeit und produktiv Aufgaben erfüllen zu können, ist es sehr wichtig, dass man Ruhe und Entspannungsphasen in den Alltag einbaut, sei es in der Schule, der Universität oder in der Arbeit.

Langes und hartes Arbeiten kann zu zahlreichen gesundheitlichen Folgen führen, wobei Stress und Leistungsdruck das noch zusätzlich verschlimmern können. Eine häufige Konsequenz ist zum Beispiel ein Burnout. Gemäß einer repräsentativen Studie [1] aus dem Jahr 2016/17 fühlen sich nur 52 Prozent der Österreicher:innen tatsächlich gesund. 19 Prozent der Proband:innen befinden sich bereits in einem Burnout-Frühstadium, 17 Prozent sogar in einem Übergangsstadium. Schließlich ergab diese Studie, dass 8 Prozent der Österreicher:innen unter dem Burnout-Syndrom leiden.

Deshalb greifen viele Menschen auf die Musik zurück, wodurch sie sich beim Arbeiten entweder besser konzentrieren oder kurz eine Pause einlegen können.

Zur Stressbewältigung wurde von der MACOLUTION GmbH und dem Data Science Unternehmen solvistas GmbH eine Lösung für dieses Problem entwickelt. Die MACOLUTION GmbH ist ein Unternehmen, welches es sich zur Aufgabe gemacht hat, Management- und Coaching-Solutions mit modernsten Technologien und bewährten Methoden zu entwickeln. [2] Mit Relaxoon soll diese Idee Wirklichkeit werden.



Abbildung 1: Logo MACOLUTION



Abbildung 2: Logo solvistas

## 2 Problemstellung

In der heutigen Welt entstehen immer mehr Termine für Besprechungen oder Meetings. Dadurch wird der generelle Alltag hektischer und schneller, was sowohl oft im Arbeitsleben als auch im privaten Bereich zutrifft. Daher empfinden viele Menschen den Ablauf des Tages persönlich als stressig.

Es fehlt außerdem die Möglichkeit sich zu entspannen und die Betroffenen verlieren den Mut sich Hilfe zu holen.

Stress oder Leistungsdruck kann neben einem Burnout auch noch weitere gesundheitliche Folgen hervorrufen. Mögliche Folgen von hohem Stress oder hohem Leistungsdruck können sein:

- Zeichen von Nervosität
- Verspannungen, die zu Kopf-, Genick- und Rückenschmerzen führen können
- Vergesslichkeit
- Depressionen
- Psychische Störungen

Anhaltender Stress kann auch zu dauerhaften Herz/Kreislauf- und Nierenerkrankungen, Stoffwechselstörungen, Allergien oder Entzündungskrankheiten führen. [3] Gesundheit ist ein sehr wichtiger Faktor, deswegen benötigen gestresste Menschen eine Möglichkeit sich wieder entspannen zu können.

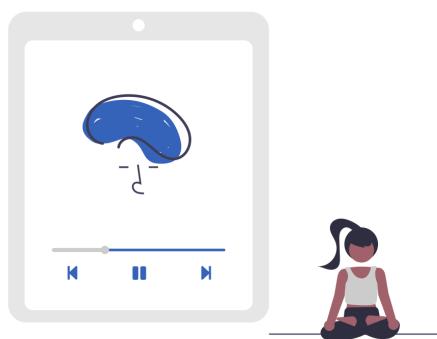


Abbildung 3: Musik dient zur Entspannung

## 3 Ziele

Es gibt einen klaren Unterschied zwischen dem Setzen von Zielen und dem tatsächlichen Erreichen von Zielen. Das einfache Festlegen von Zielen allein genügt nicht. Um Erfolg in einem bestimmten Vorhaben zu haben, ist es notwendig, klare Ziele zu definieren. Ein Ziel repräsentiert einen Zustand, ein Ergebnis oder einen bestimmten Ort. Der Grad und die Art der Zielerreichung sind entscheidend für die Definition von Erfolg. Daher ist das Festlegen eines Ziels lediglich der Ausgangspunkt. Dafür müssen Fortschritte oder Meilensteine erreicht und der vorgegebene Zeitrahmen eingehalten werden – und dies alles unter Verwendung legaler und legitimer Mittel und Methoden. Oftmals scheitert man nicht an mangelnder Disziplin oder Motivation, sondern an falsch formulierten oder unpassenden Zielen.

Ziele charakterisieren sich durch einfache Aufgaben, die nach der Reihe erledigt werden – sie repräsentieren feste Absichten. Hinter jedem Ziel steht immer ein konkretes Bestreben. Ziele sind nicht nur das Ergebnis rationaler Überlegungen, sondern vor allem eine Angelegenheit des Herzens und der Motivation.[4]



Abbildung 4

Ziele zu setzen ist wichtig, deswegen müssen sie von Anfang an **richtig** gesetzt werden. Beim Setzen der Ziele kann man sich an folgenden Punkten orientieren: [4]

- Klare Konkretisierung ist entscheidend für Ziele! Je präziser die Beschreibung ist, desto einfacher ist es, darauf hinzuarbeiten.
- Ziele müssen realistisch sein! Zu ehrgeizige Ziele können Frustration und Selbstzweifel auslösen. Sowohl das angestrebte Ergebnis als auch die dafür vorgesehene Zeit sollten der Realität entsprechen.
- Wenn Ziele öffentlich gemacht werden, erhöht das auch die Erfolgswahrscheinlichkeit. Der Austausch über Ziele mit Familie, Freund:innen oder Kolleg:innen steigert die Motivation und erzeugt Verbindlichkeit, da man es anderen beweisen möchte.
- Immer positiv bleiben! Positive Formulierungen der Ziele motivieren langfristig mehr.
- Ein eigener Antrieb ist notwendig, um die Zielabsicht vor Augen zu haben und um dranbleiben zu können.
- Ziele erfordern Flexibilität. Der Weg zum Ziel kann sich ändern, ebenso die Zeit für das Erreichen eines Meilensteins oder sogar das Ziel selbst. Ziele sind dynamisch und nie statisch oder unveränderlich.
- Zur Visualisierung der Ziele sollten sie stets in der Gegenwart formuliert werden, damit man sich gleich Bilder im Kopf vorstellen kann. Das steigert außerdem die Motivation.

## 3.1 Projektziele

Bei den Projektzielen ist es wichtig sich nicht zu große Ziele vorzunehmen und nicht nur das eine Ziel im Auge zu haben. Es wichtig ein großes Ziel aufzuteilen, sodass viele kurzfristige Ziele zu einem langfristigen Ziel führen können. Langfristige Ziele bilden dabei auch die Leistungswirkung.

Relaxoon wurde in jeweils zweiwöchigen Sprints entwickelt. Es wurden am Anfang jedes Sprints einige kleine/kurzfristige Ziele definiert, die in den darauffolgenden Wochen erreicht werden sollten. Am Ende des Sprints wurde dann der damalige Stand des Projektes mit den Zielen verglichen, um zu wissen, ob alle Ziele erreicht wurden und welche Ziele für den nächsten Sprint vorgesehen waren.

Übersicht der wichtigsten kurzfristigen Ziele pro Sprint:

- Sprint 0: Wahl des CMS zwischen Strapi und Wordpress
- Sprint 1: Strapi- sowie App-Setup fertigstellen, Beginn UI-Design der App
- Sprint 2: Zugriff von Frontend auf Backend möglich
- Sprint 3: Beginn Implementierung des Category-Screens
- Sprint 4: Datenmodell von Relaxoon mit Strapi abgleichen
- Sprint 5: Implementierung der Datenstruktur in Strapi
- Sprint 6: UI-Design in der App umsetzen, Entwicklung Suchleiste
- Sprint 7: Echte Daten des Backends am Frontend anzeigen
- Sprint 8: Implementierung des Registrierungsformulars und das Setzen eines Favoriten
- Sprint 9: Anmeldung in der App ermöglichen, Umsetzung Light und Dark Mode
- Sprint 10: Auto-Login, Tags für Medien hinzufügen

Faktoren wie Zeit, Kosten, Nutzen und Qualität sollten ebenfalls bei jedem Ziel berücksichtigt werden. Man sollte darauf achten, dass sich diese Faktoren gut miteinander vereinen lassen:

- Zeit: Kann das Ziel in der geplanten Zeit erreicht werden?
- Kosten: Welche Kosten werden bei der Umsetzung anfallen?
- Nutzen: Was wird der Nutzen dieses Ziels sein? Für welchen Zweck wird es gebraucht?
- Qualität: In welcher Qualität, soll das Ziel umgesetzt werden?

### 3.1.1 Leistungswirkungen

Leistungswirkungen eines Projektes beschreiben nicht das Ziel an sich, sondern was aus den Zielen folgt. Die Entspannungsapp Relaxoon soll bei den Usern und Userinnen der App für geringen Stress und eine hohe Stressreduktion sorgen, was die Personen produktiver und glücklicher macht. Eine langfristige Erhaltung oder Verbesserung der Gesundheit wäre eine ideale Leistungswirkung.

Um die formulierten Ziele in gewünschter Qualität und fristgerecht zu erreichen, mussten neben Schul- bzw. Lernanforderungen die Ausdauer und Priorität auf das Projekt gesetzt werden.

## 4 Aufgabenstellung

Die Firma Macolution hat Kontakte zu Streaming- und Gesundheitsexperten. Diese Fachleute erstellen viele Videos, Übungen und Artikel zu diesen Themen. Der Auftraggeber will eine Applikation für diesen Kunden entwickeln, die folgende Kriterien erfüllt:

- User:innen müssen sich in der App nicht um die eigene Erstellung von Entspannungsmedien wie Texten oder Videos kümmern, weil diese Medien bereitgestellt werden, welche im Backend eingepflegt sind.
- Medien individuell favorisierbar und mehreren Kategorien zugewiesen zu sein, die per Suchleiste in der App einfach zu finden sind.

Das Minimum Viable Product (MVP) von Relaxoon soll realisiert werden, damit möglichst früh Feedback von Nutzer:innen eingeholt werden kann, welches für die Weiterentwicklung und für die Verbesserung einer App essenziell ist.

Dies bestätigt der deutsche Content-Autor Philipp Steubel, der sich im amerikanischen Softwareunternehmen Asana auf die Bereiche Marketing und Projektmanagement spezialisiert hat: „Ein MVP ist sicherlich ein gutes Hilfsmittel, wenn es darum geht, den Entwicklungsprozess an den Kundenbedürfnissen zu orientieren. Immerhin bekommen Sie so schon während der Entwicklung des Produktes wertvolles Kundenfeedback. Somit erfahren Sie aber auch mehr über die Zielgruppe selbst, welche Personengruppen von dem Produkt bzw. dem Service überzeugt sind und wie Sie die Marketingkampagnen aufsetzen können.“ [5]

Wichtig dabei ist:

- die Feinabstimmung der Qualität der Inhalte
- die optimale Bereitstellung der sorgfältig ausgewählten Inhalte
- ein ansprechendes und benutzerfreundliches Design für den passenden „Look and Feel“ der App.

## 4.1 Use-Case-Diagramm (UCD)

Für das Relaxoon gibt es im Wesentlichen zwei Hauptakteure: der Content-Manager und die Benutzer der App. Diese Akteure interagieren mit verschiedenen Funktionen innerhalb der Anwendung, wie in der folgenden Grafik dargestellt.

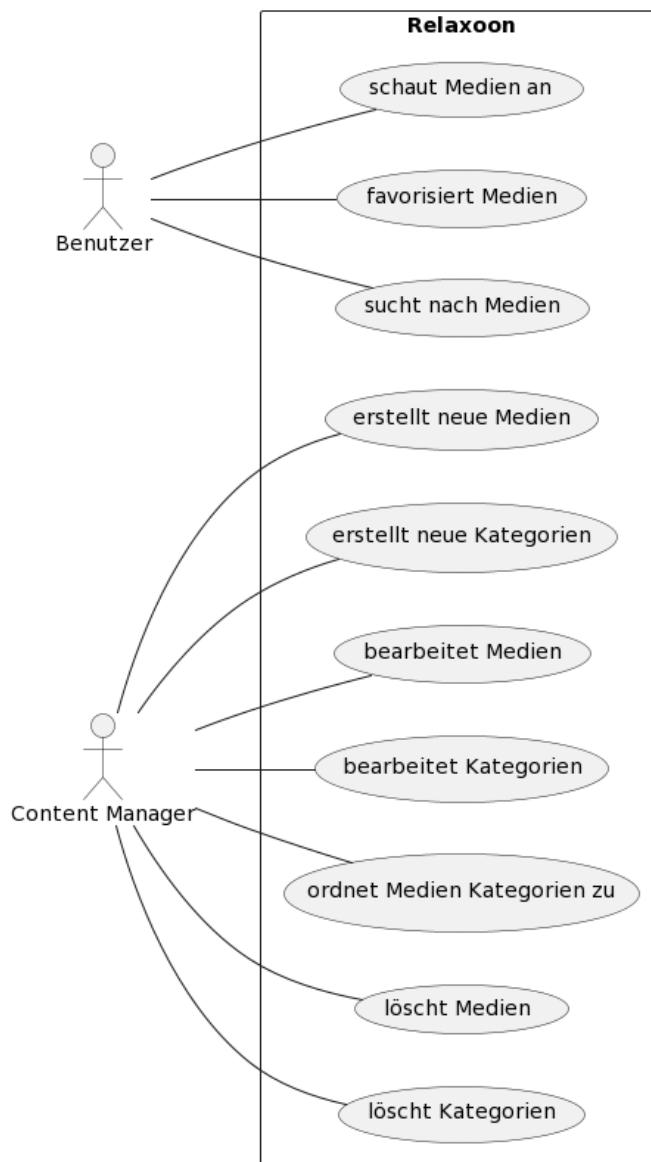


Abbildung 5: Use-Case-Diagramm

**Benutzer:in**

- 

**Content Manager:in**

-

# **5 Marktanalyse**

„ Marktanalysen sind die Grundlage für wichtige strategische Geschäftsentscheidungen. Nicht nur Standortentscheidungen und die Wahl von Marketingmaßnahmen hängen davon ab, sie können – je nach Schwerpunkt – auch ausschlaggebende Informationen zur Preispolitik, Expansionsplänen und Produktentwicklung beisteuern. So helfen sie etwa dabei, dass ein Markteintritt oder die Neueinführung eines Produktes gelingt oder ganz neue Märkte erschlossen werden können.“ [6]

## **5.1 Vorgehensweise**

In einem ersten Schritt wird der Markt und die Zielgruppe definiert. Ist die App global verfügbar oder nur in bestimmten Regionen? Wer sind die potenziellen Benutzer:innen? Wie viel Interesse besteht an der App von den Kund:innen?

In einem nächsten Schritt muss der Wettbewerb untersucht werden. Welche Entspannungsapps sind auf dem Markt, welche davon sind am erfolgreichsten und warum? Dazu kommt noch eine Bewertung der aktuellen Trends im Bereich Gesundheit und Wohlbefinden. Gibt es aktuelle oder neue Technologien oder Forschungsergebnisse, die in die App integriert werden könnten?

Danach werden verschiedene Geschäftsmodelle in Erwägung gezogen. Soll die App kostenlos sein und durch Anzeigen oder In-App-Käufe finanziert werden? Oder wird es eine kostenpflichtige Premium-Version geben, mit der man mehr Funktionen innerhalb der App freischaltet?

Schließlich muss eine Marketingstrategie entwickelt werden, um die Entspannungsapp bekannt zu machen und in Umlauf zu bringen. Welche Kanäle sind am besten dazu geeignet, die gewünschten Zielgruppen zu erreichen?

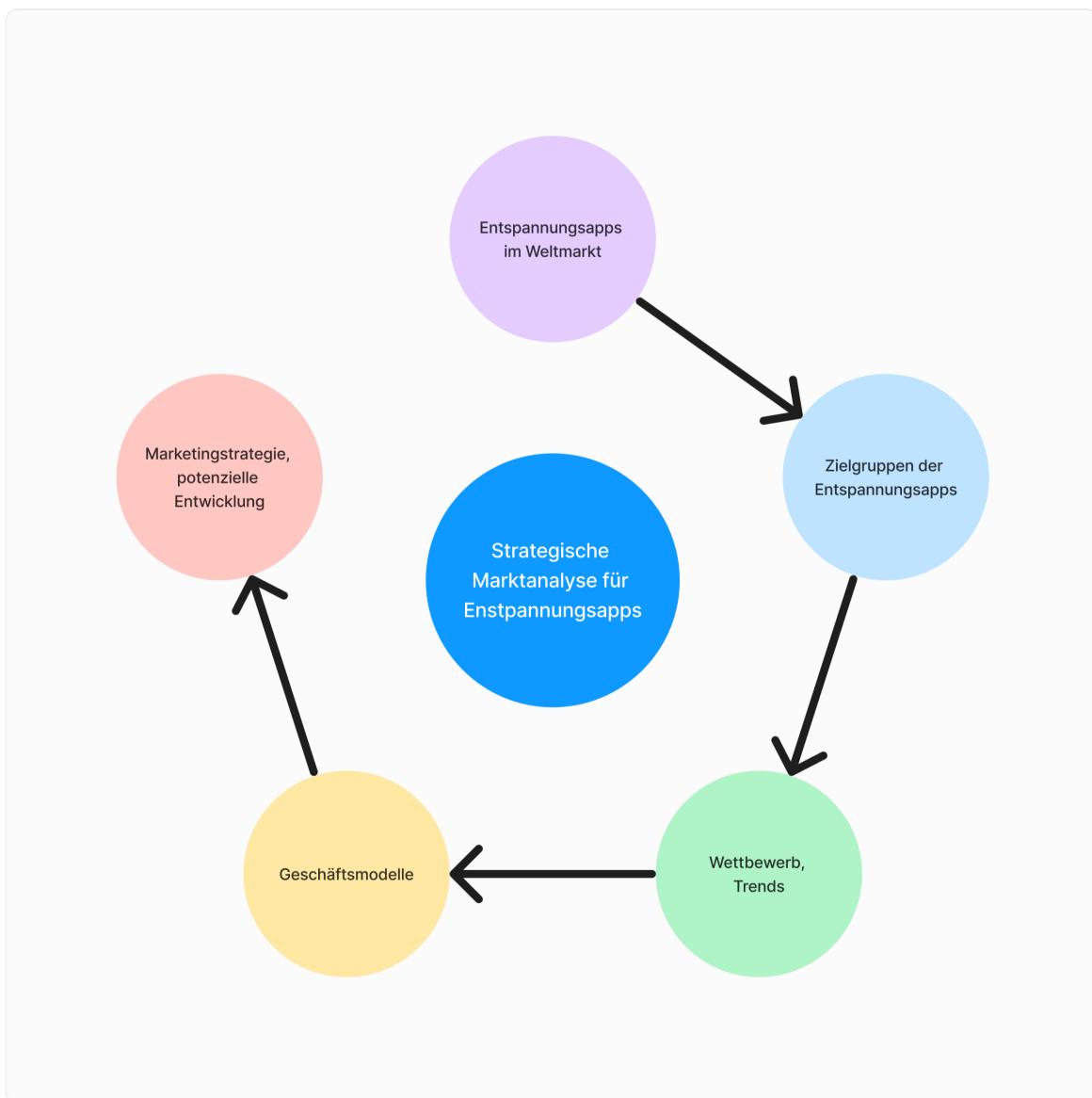


Abbildung 6: Ablauf

Durch diese Analyse erhält man wichtige Erkenntnisse über die aktuelle Situation sowie potenzielle Entwicklungen auf dem Markt für Entspannungsapps.

## 5.2 Analyse

Wie schon bei der Planung des Projektes formuliert war, soll die App Relaxoon, welche Stress bei bedürftigen Personen reduziert, im Play Store und im App Store international verfügbar sein. Der Markt bei Entspannungsapps ist sehr groß, was zu hoher Konkurrenz führt.

### 5.2.1 Wettbewerb und Trends

Einerseits gibt es viele Content-Creator, die ihre selbst erstellten Entspannungsmedien und -übungen, ohne die Verwendung einer konkreten Entspannungsapp, auf den bekanntesten und auch größten Plattformen wie zum Beispiel YouTube oder Spotify direkt hochladen. Das könnte bei einigen Menschen dazu führen, sich gar keine App installieren zu wollen, weil es unnötig erscheint. Andererseits bieten Apps, die extra auf Entspannung konzipiert sind, massive Vorteile.

Einige markante Features von Entspannungsapps sind:

- Kurse, die eine Reihe von Entspannungsübungen in gezielter Reihenfolge bereitstellen
- Viele verschiedene Funktionen zum Abspielen von beruhigenden Geräuschen und Musik
- Anleitungen zur Meditation, welche ebenfalls als Einschlafhilfe dienen und für einen besseren Schlaf sorgen

Bei Entspannungsapps gibt es auch verschiedene Arten. Zum einen welche, die sich ausschließlich auf Meditation spezialisiert haben, wohingegen andere nur audiovisuelle Inhalte zur Verfügung stellen. Es gibt ebenfalls Apps auf dem Markt, die ausschließlich Spiele innerhalb der App zur Entspannung implementiert haben.

Die Funktion von spielerischem Entspannen soll in Relaxoon nicht vorhanden sein, sondern Relaxoon soll die Möglichkeit bieten, alle Bedürfnisse der User:innen abzudecken, wobei man sich persönlich als User:in auch dafür entscheiden kann, nach bestimmten Vorgaben zu filtern.

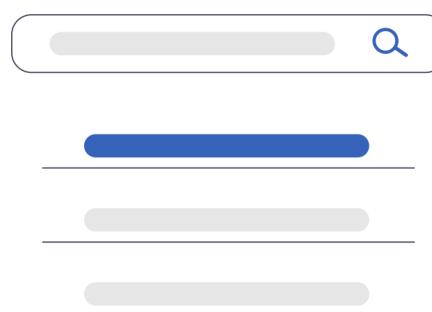


Abbildung 7

Man kann sich also dafür entscheiden, einen Text zum Entspannen zu Lesen, ruhige Musik zu hören, sich ein entspannendes Video anzusehen, oder eine eigene Übung zu machen, in der mehrere Sinne angesprochen werden.

### 5.2.2 Geschäftsmodelle

Bei den Geschäftsmodellen gibt es, wie bereits oben erwähnt, für eine Entspannungsapp verschiedene Varianten.

#### **Werbung**

Das Prinzip von Werbung in der App wäre die erste Möglichkeit mit dem Projekt Geld zu verdienen. Es würde jedoch den Zweck, dass man sich bei der Verwendung der App entspannen soll, sehr stark negativ beeinflussen. Wenn man sich zum Beispiel vor jeder Übung ein 30-sekündiges Werbevideo anschauen muss, oder wenn Pop-Up-Fenster auftauchen würden, die der Nutzer oder die Nutzerin ständig wegtippen müssten, um fortfahren zu können, wäre das kontraproduktiv. Das würde sehr viele Leute abschrecken und darin resultieren, dass die App häufig direkt wieder deinstalliert wird.

#### **Einzelne In-App-Käufe**

Da ist das Prinzip von In-App-Käufen sehr viel ansprechender. Es behindert die Nutzer:innen in keiner Weise daran, die App ohne Unterbrechungen zu verwenden. Man könnte In-App-Käufe den Kund:innen so anbieten, dass sie zum Beispiel für eine kleine Summe eine Übung freischalten können, oder auch für eine höhere Summe gleich zehn komplette Übungen auf einmal erwerben können.

#### **Premium Version**

Die wahrscheinlich geeignetste Lösung für Monetarisierung ist eine Premium Version für Relaxoon zu verkaufen, mit der man die gesamten Features der App benutzen kann. Diese könnte man einmalig bezahlbar machen, sodass die User:innen für immer Zugriff auf Premium Inhalte bekommen, oder auch monatlich verrechnen, dass jeden Monat wie bei einem Abonnement gezahlt werden muss.

Es wichtig zu erwähnen, dass man die App auch schon im Play- und im App Store käuflich machen könnte. Das würde jedoch die Verbreitung und Reichweite der App einschränken, weil sich durch den Preis, der schon vorab zu bezahlen wäre, weniger Leute die App tatsächlich kaufen und dann erst herunterladen würden. Eine von Grund auf kostenlose App passt deswegen besser.

## **Marketingstrategie**

Im Online- und Web-Marketing steht die Steigerung der Sichtbarkeit und die Umwandlung von Besuchern in zahlende Kunden im Vordergrund. Es wird oft auf Strategien wie Suchmaschinenoptimierung (SEO), Content-Marketing und Social-Media-Werbung gesetzt wird, um eine breite Online-Sichtbarkeit zu erreichen. Im Gegensatz dazu legt das App-Marketing den Fokus auf Aspekte wie App Store-Optimierung (ASO), gezielte Werbekampagnen innerhalb von App-Plattformen und die Schaffung einer positiven Nutzererfahrung, um sowohl die Anzahl der Downloads als auch das langfristige Engagement der Nutzer zu steigern. [7]

Ein Problem was häufig dabei entsteht ist, dass Personen eine App oft gar nicht lange installiert haben und oft schon diese nach nur einmaligem Nutzen wieder deinstallieren.

Auf der Website [absatzwirtschaft.de](http://absatzwirtschaft.de) wurde das noch genauer erläutert: [8]

„App-Marketer haben weniger als sechs Tage Zeit, um das Interesse von Nutzern erneut zu wecken: Der durchschnittliche Nutzer wartet zwischen dem letzten Öffnen der App und der Deinstallation knapp sechs Tage. Dabei variiert die Zeitspanne stark zwischen den einzelnen App-Kategorien – von weniger als drei Stunden bis zu etwa 15 Tagen. Entertainment- und Lifestyle-Apps werden besonders schnell verworfen, ihre Nutzer deinstallieren Apps im Durchschnitt nach einem halben bis ganzen Tag. E-Commerce- und Reise-Apps hingegen haben eine wesentlich längere Lebensdauer und werden in der Regel zehn bis elf Tage nach der letzten Session eines Nutzers gelöscht.“

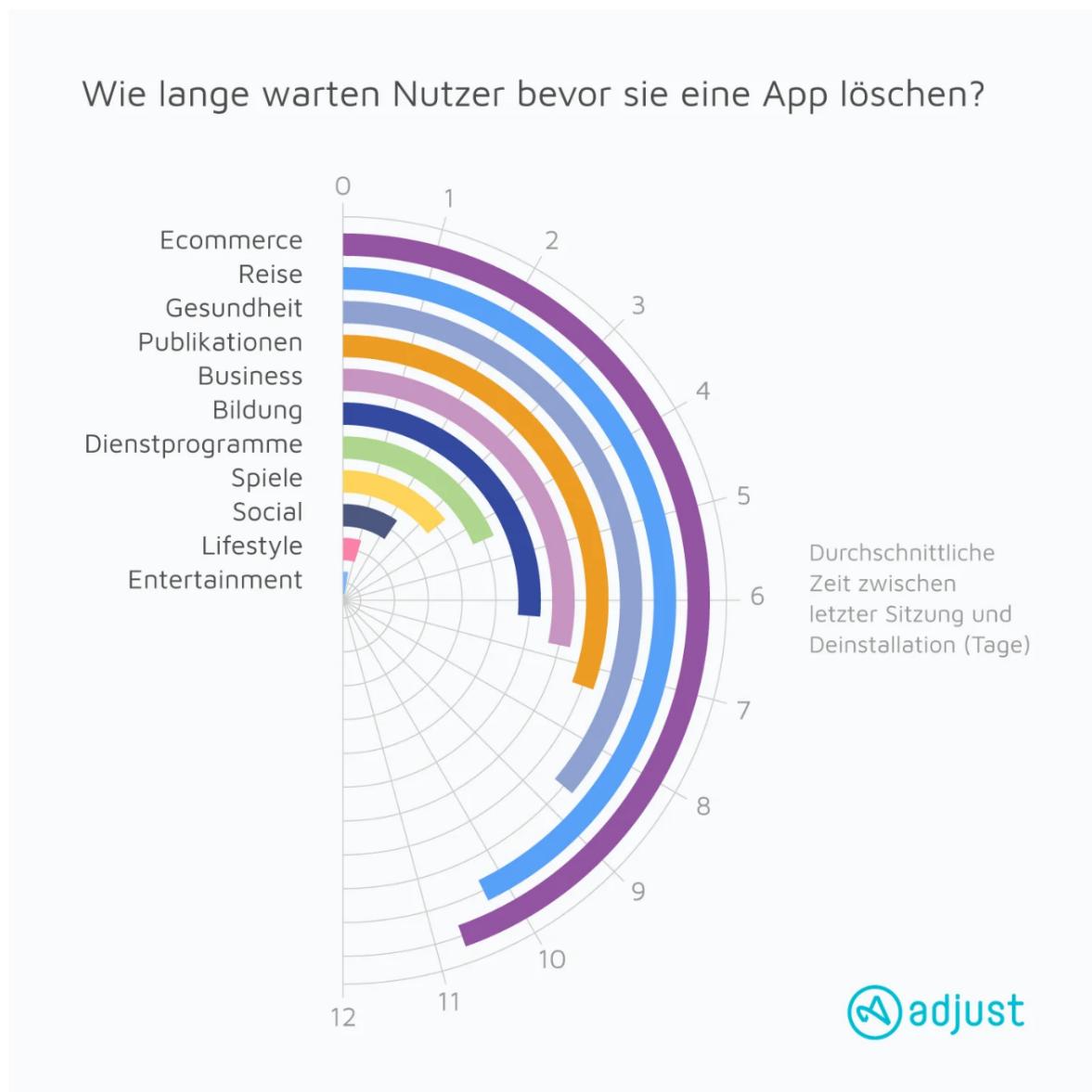


Abbildung 8: App Statistik [8]

# 6 Projektanforderungen

Beim Projektmanagement wird grundsätzlich zwischen zwei Hauptkategorien von Anforderungen unterschieden.

## 6.1 Funktionale Anforderungen:

- beschreiben, welche Funktionen das System bereitstellen soll und welche Aufgaben es ausführen muss.
- definieren die spezifischen Funktionen, Dienste oder Aufgaben, die das System erfüllen muss, um die Benutzeranforderungen zu erfüllen.
- konzentrieren sich auf die Was-Aspekte des Systems und beschreiben, welche Ergebnisse erwartet werden.
- Beispiel: „Das System muss dem Benutzer ermöglichen, sich mit einem Benutzernamen und Passwort anzumelden.“

## 6.2 Nicht funktionale Anforderungen:

- beschreiben Eigenschaften des Systems, die nicht unbedingt Funktionen sind, sondern Qualitätsmerkmale, die das System erfüllen muss.
- betreffen oft die Wie-Aspekte des Systems und definieren Qualitätsmerkmale wie Leistung, Sicherheit, Benutzerfreundlichkeit und Zuverlässigkeit.
- berücksichtigen Aspekte wie Skalierbarkeit, Wartbarkeit, Zuverlässigkeit, Leistung und Benutzerfreundlichkeit.
- Beispiel: „Das System muss eine Antwortzeit von weniger als 2 Sekunden für Benutzeranfragen sicherstellen.“

Die oberen zwei Abschnitte 6.1 und 6.2 wurden von einem KI-Modell generiert.[9]

Bei Relaxoon können alle Anforderungen auch in funktionale und nicht funktionale Anforderungen unterteilt werden. Eine nicht funktionale Anforderung wäre zum Beispiel die einfache Bedienung der App. Der User oder die Userin soll sich in der App problemlos zurechtfinden können, ohne dafür ein Manual zu benötigen. Dazu braucht man:

- eine Erklärung am Anfang bzw. ein Tutorial, wenn man die App das erste Mal startet, um überhaupt zu wissen, wie die App funktioniert und wie sie zu verwenden ist.
- ein schlichtes und übersichtliches Registrierungsformular mit Begründung dazu, damit der User oder die Userin weiß, warum er sich registrieren muss.
- eine Homepage, auf der man sich leicht zurechtfindet und eine Übersicht über populäre Medien hat, die bereits hochgeladen worden sind.
- eine Navigationsleiste, mit der man einfach zu anderen Seiten oder Features der App weitergeleitet wird.
- einen klaren Weg, um selbst personalisierte Einstellungen in der App treffen zu können.

Auf der anderen Seite wäre ein Beispiel für eine funktionale Anforderung die Möglichkeit, sich wieder einloggen zu können, wenn man sich bereits schonmal registriert hat, damit ein zu lange dauernder Login die User:innen nicht verleitet, die App direkt wieder zu verlassen. Das kann man mit einer „Login Daten merken“-Option umsetzen oder man verwendet eine „Ich besitze bereits einen Account“-Option bei User:innen, die sich die App auf einem zusätzlichen Endgerät heruntergeladen haben. Eine weitere Projektanforderung für Relaxoon ist die **Übertragbarkeit** d.h. die App für mehrere Betriebssysteme nutzbar machen. Wenn eine App übertragbar ist, schafft das einige Vorteile:

- erhöhte Zielgruppenabdeckung
- keine Vorgabe des Mobilgerätetyps
- Wettbewerbsvorteil gegenüber anderen

# 7 Technologien

## 7.1 Strapi

Strapi ist ein Headless Content Management System (CMS), welches eine vorgefertigte Benutzeroberfläche für Content-Creator und auch für die Entwickler bereitstellt.

Mit der Verwendung davon sind Inhalte und dazu gebrauchte technische Funktionalitäten sehr einfach erstellbar. [10]

Beispiele dafür sind:

- REpresentational State Transfer (REST)- bzw. Graph Query Language (Graphql)-Schnittstellen
- Logik für die Authentifizierung
- Create, Read, Update und Delete (CRUD) Funktionalitäten jeder Business Entität

## 7.2 Firebase App Distribution

“ Firebase App Distribution macht die Verteilung Ihrer Apps an vertrauenswürdige Tester problemlos. Indem Sie Ihre Apps schnell auf die Geräte der Tester übertragen, können Sie frühzeitig und häufig Feedback einholen. Und wenn Sie Crashlytics in Ihren Apps verwenden, erhalten Sie automatisch Stabilitätsmetriken für alle Ihre Builds, sodass Sie wissen, wann Sie zur Auslieferung bereit sind. ”[11]

Der Vorteil von Firebase App Distribution ist, dass man die Applikation auf dem eigenen Mobilegerät ausprobieren kann, ohne die App auf dem Play Store bzw. App Store hochladen zu müssen.

# 8 Systemarchitektur

## 8.1 Komponentendiagramm

Für eine klare funktionale Übersicht auf der vorliegenden Diplomarbeit wurde ein sogenanntes Komponentendiagramm für das technische System von Relaxoon erstellt. Da Relaxoon eine Applikation ist, die auf Mobilgeräten läuft, wurde für die Entwicklung das cross-plattform **JavaScript (js)** Framework "React Native" eingesetzt. Damit der Kunde Inhalte in die App hinzufügen kann, wurde ein "Node.js" basiertes Headless CMS namens "Strapi" verwendet.

Für die Kommunikation zwischen Frontend und Backend wird REST verwendet. Für die Persistierungsebene hat sich das Team für "PostgreSQL" entschieden.

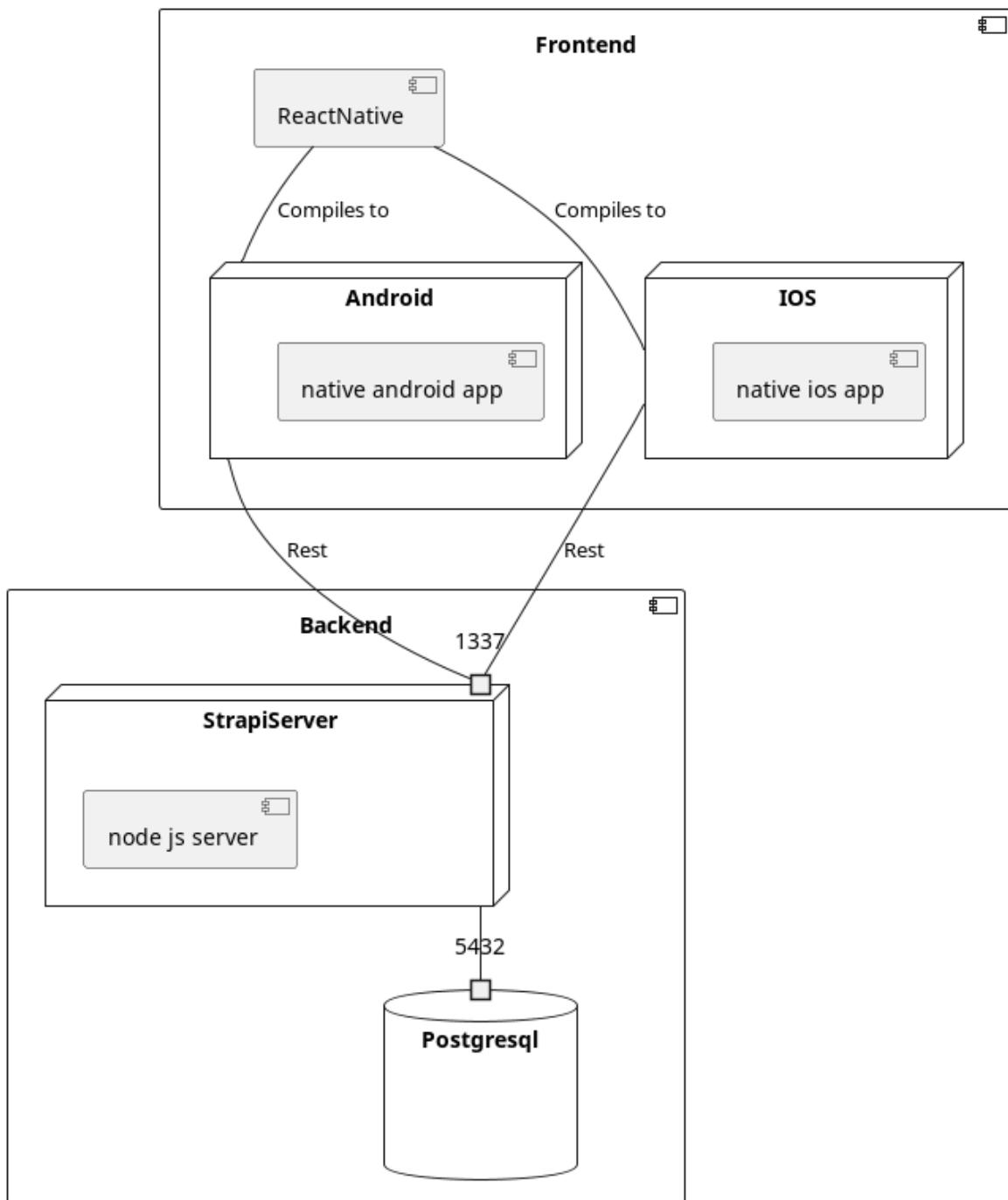


Abbildung 9: Systemarchitektur

# 9 Entwurfsentscheidungen

## 9.1 React Native

Grundsätzlich wurde React Native aus mehreren Gründen verwendet:

- React Native ist das Standardframework für Cross-Platform-Lösungen bei der Firma Solvistas.
- Vorhandene Kenntnisse in den Sprachen Javascript bzw. Typescript
- Der Auftraggeber forderte eine Cross-Platform-Lösung für das Frontend, um IOS-Android- Betriebssysteme abzudecken
- Endergebnis von React Native ist eine native App

### 9.1.1 Was ist eine Cross-Platform-Lösung

“ Eine Cross-Platform App besteht aus einem einzigen Code, der jeweils in die native Systemsprache von Apple, Android & Co. kompiliert wird. Dadurch erhält man eine App, die mit wenig Entwicklungsaufwand auf mehreren Betriebssystemen zur Verfügung steht, sich aber dennoch wie eine native App anfühlt.” [12]

### 9.1.2 Warum wurde eine Cross-Platform-Lösung angefordert

Der Client von Relaxoon besteht aus einer einfachen Applikation, die keine Business Logik und auch keine Performance kritische Funktionalitäten hat. Daher ist die Verwendung einer Cross-Platform-Lösung sehr vorteilhaft, da alle Plattformen von einem Codebase gepflegt werden können.

### 9.1.3 Warum ist Relaxoon native und nicht hybrid

hybride Applikationen:

“ Eine hybride App kombiniert die besten Elemente von [N]ativen und Web-Apps. Sie werden wie eine native App installiert, aber es ist eigentlich eine Web-App innerhalb des Endgeräts. Hybride Apps werden in den gängigsten Sprachen für die Web-App Entwicklung, wie z.B. HTML und CSS, programmiert. Dies bedeutet, dass sie auf verschiedenen Plattformen verwendet werden können. Obwohl sie in der Sprache der Webanwendung entwickelt wurden, haben sie die gleiche Fähigkeit wie native Apps, sich an verschiedene Geräte, wie ein Tablet, Smartphone usw. anzupassen. ” [13]

native Applikationen:

“ Eine native App ist eine Anwendung, die entwickelt wurde, um auf einer bestimmten Plattform oder einem bestimmten Endgerät zu arbeiten. Aus diesem Grund können native Apps mit den auf der jeweiligen Plattform installierten Betriebssystem[s]funktionen interagieren und diese nutzen. ”  
[13]

Grundsätzlich sind native Applikationen viel schneller und barrierefreier als hybride Applikationen [13]

### 9.1.4 Alternativen für React Native

#### Kotlin Multiplatform

Kotlin Multiplatform (KMP) bietet bessere Performance als React Native und hat auch eine modulare Integration.

##### modulare Integration:

“ Probably the biggest benefit in favor of Kotlin Multiplatform is that it's an SDK and not a framework. This means that teams with existing apps can simply add a module or migrate a small part to assess its viability without a huge commitment. This really helps Kotlin address the biggest deterrent when moving to a new codebase. ” [14]

Allerdings ist das Problem davon, dass KMP noch immer in der Beta-Version ist und daher ist diese nicht stabil. Außerdem ist die Community von dieser Alternative nicht so groß wie die Community von React Native. Gute Kenntnisse in Swift User Interface (UI) und Android Software Development Kit (SDK) werden für die Verwendung von Kotlin Multiplatform auch verlangt. Grund dafür ist, dass KMP keine Bibliotheken für UI-Elementen wie React Native oder Flutter bereitstellt, sondern die Syntax der OS-Technologie verwendet. [14]

#### Flutter

Flutter an sich ist viel schneller als React Native. Außerdem unterstützt Flutter die Betriebssysteme Windows, Linux und macOS zusätzlich zu Web, Android und iOS. [15] Das Problem bei Flutter ist, dass man gute Kenntnisse in der Programmiersprache "Dart" haben muss. [15] Für die Entwicklung von React Native sind gute Kenntnisse in den Programmiersprachen Javascript und Typescript von Vorteil. Außerdem ist die Unterstützung von macOS, Linux und Windows nicht relevant für eine mobile App.

#### Xamarin

Xamarin ist eine C# Framework, die für die Entwicklung von nativen Cross-Platform-Lösungen zuständig ist. Die Performance von Xamarin ist viel besser als die Performance von React Native. [16]

Die Entwickler haben sich aus den Gründen, die bei Flutter bereits erwähnt wurden, für React Native entschieden. Außerdem gibt es beim Xamarin kein "hot reloading" und daher muss das Program bei jeder kleinen Änderung neugestartet werden.[16]

### 9.1.5 Nutzwertanalyse für das Frontend-Framework

Kriterien	React Native	Xamarin	Flutter	KMP
Vorhandene Kenntnisse max. 50%	50%	0%	0%	10%
Barrierefreiheit max. 30%	15%	5%	30%	20%
Performance max. 20%	5%	20%	15%	10%
Summe	70%	25%	45%	40%

## 9.2 Strapi

### 9.2.1 Was ist ein CMS

“Ein Content-Management-System (CMS) ist eine Softwareanwendung, die es Benutzern ermöglicht, digitale Inhalte zu erstellen, zu bearbeiten, gemeinsam zu editieren, zu veröffentlichen und zu speichern. Content-Management-Systeme werden typischerweise für Enterprise Content Management (ECM) und Web Content Management (WCM) eingesetzt.” [17]

### 9.2.2 Warum wurde ein CMS verwendet

Der Auftragsgeber wollte die Applikation so schnell wie möglich veröffentlichen, da die Anzahl der Apps, welche die gleichen Anwendungsfälle wie Relaxoon haben, nicht so groß ist. Daher ist die Verwendung eines fertigen CMSes viel schneller als die Implementierung eines Backends.

### 9.2.3 Was ist ein Headless-CMS

“ Ein Headless CMS ist sowohl eine Weiterentwicklung als auch eine Verknappung eines klassischen CMS. Dem System werden integrale Bestandteile genommen, um es für unterschiedlichste Ausgaben kompatibel zu machen. Das gelingt dadurch, dass Frontend und Backend in einem Headless CMS nicht mehr monolithisch miteinander verknüpft sind. Das fehlende Frontend ist auch der Grund, wieso derartige CMS-Systeme als „kopflos“ (englisch: „headless“) bezeichnet werden. ” [18]

### 9.2.4 Warum wurde ein Headless-CMS verwendet

- Vorgabe des Auftraggebers
- Andere Arten von CMSes generieren statische HTML Seiten, welche für Relaxoon gar nicht gebraucht werden, da Relaxoon eine mobile App ist.
- Die Firma Solvistas beschäftigt sich intensiv mit Data Science. Da man auf das CMS mittels REST zugreifen kann, will Solvistas in der Zukunft ein Datenzentrum aus diesem erstellen. Zusätzlich wird das CMS für Data Science Zwecke verwendet.

## 9.2.5 Warum Strapi und nicht Wordpress Application Interface (API)

### Vorteile von Wordpress API

- Es gibt sehr viele Einstellungen und Features
- Unterstützt Search Engine Optimization (SEO)
- Ein guter Community Support
- Es ist einfach zu verwenden

[10]

### Nachteile von Wordpress API

- Limitierte Flexibilität, da viele Plugins und Addons kostenpflichtig sind. die kostenlose Verwendung von einigen Addons und Plugins ist limitiert.
- Es ist nicht geeignet für eine Software mit großer Skalierung.
- Wordpress wird von vielen Menschen verwendet und deshalb ist es für die Hacker nicht unrelevant.
- Die Benutzeroberfläche davon ist nicht gut designt und deshalb ist die Verwendung davon meistens sehr verwirrend und unangenehm zu bedienen.

[10]

### Vorteile von Strapi

Im Gegensatz zu Wordpress, kann man mit Strapi beliebig viele Plugins und Addons verwenden. Außerdem gibt es eine eingebaute Authentifizierung bzw. Autorisierungsfunktion zusätzlich zur Unterstützung von 20 unterschiedlichen Sprachen und REST bzw. Graphql APIs. [10]

### Nachteile von Strapi

Ein Grundwissen in der Programmierung ist erforderlich, wenn man sich für Strapi entscheidet. Außerdem ist es nicht so weit verbreitet wie Wordpress und somit ist die Anzahl der 3rd-Party-Libraries, die man in Strapi einbetten kann, nicht so groß. [10]

### Ergebnis

Alle erwähnten Pros und Kontras zeigen, dass die User EXperience (UX) und die Skalierbarkeit von Strapi viel besser als Wordpress ist. Außerdem ist Strapi viel schneller als Wordpress, da Strapi in Node.js geschrieben ist und Wordpress in PHP. [10]

## 9.2.6 Nutzwertanalyse für das Headless-CMS

Kriterien	Strapi	Wordpress API
Security max.30%	30%	15%
Flexibilität max. 30%	30%	15%
Community Support max. 30%	10%	25%
Performance max. 10%	10%	5%
Summe	80%	65%

## 9.3 Component Library

Der Standardweg, um UI-Elementen in React Native zu stylen, ist JSS. Diese ist eine Technologie, wo man Cascading Style Sheet (CSS) in Form von Javascript Objekten schreibt. Das Problem bei dieser Technologie liegt dabei, dass viele CSS Attribute und Funktionen nicht dabei inkludiert sind. Es gibt auch außerdem spezifische OS-Einstellungen für Barrierfreiheit wie zum Beispiel die Einstellungen für Screenreader, die man bei der Entwicklung immer wieder vergessen kann. Aus diesen Gründen hat sich das Team für die Verwendung von 'Native Base' entschieden.

# 10 Design

## 10.1 Mockups

Als Grundlage für die Entwicklung des Design von Relaxoon standen Design-Mockups des Auftraggebers zur Verfügung.

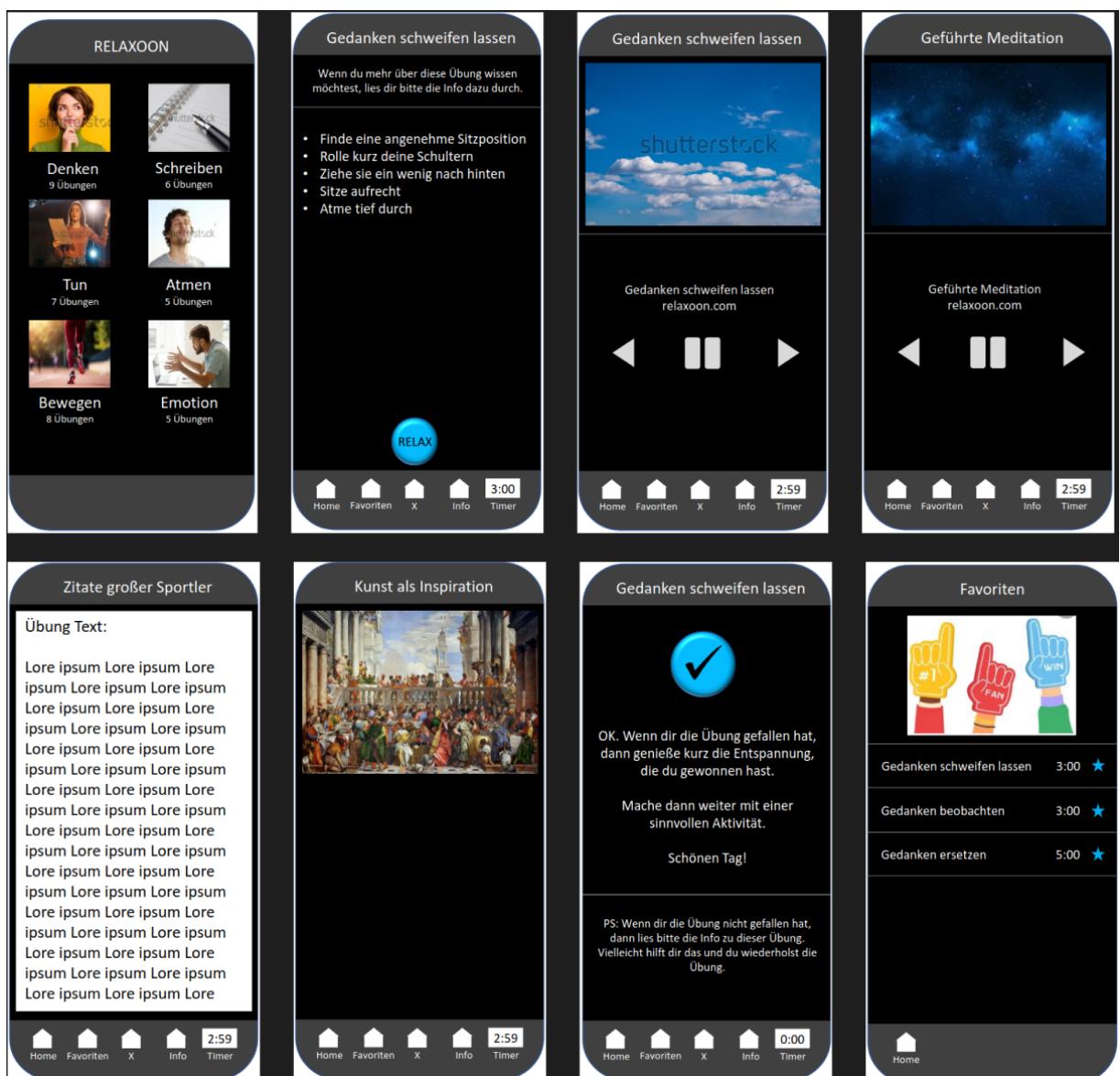


Abbildung 10: Mockups

Von oben links bis unten rechts auf dem Bild sind folgende Seiten abgebildet:

1. Home-Screen
2. Übung: Intro
3. Übung: Video
4. Übung: Ton
5. Übung: Text
6. Übung: Foto
7. Übung: Outro
8. Favoritenliste

## 10.2 Tutorial

<Bilder einfügen>

## 10.3 Login- und Registrierungsformular

<Bilder einfügen>

## 10.4 Home-Screen

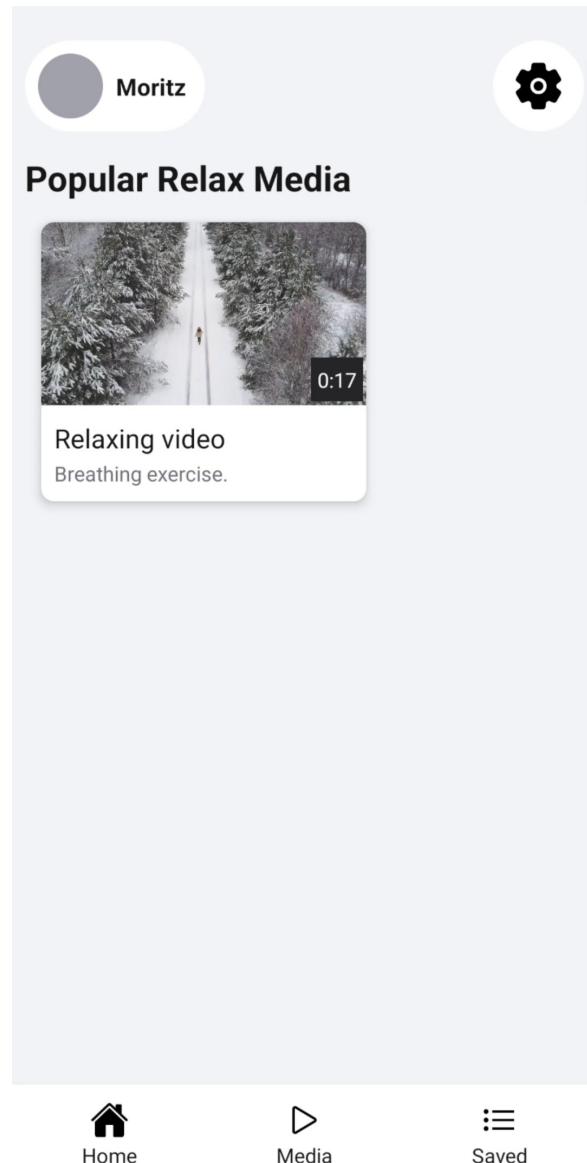


Abbildung 11: Home-Screen

## 10.5 Kategorien

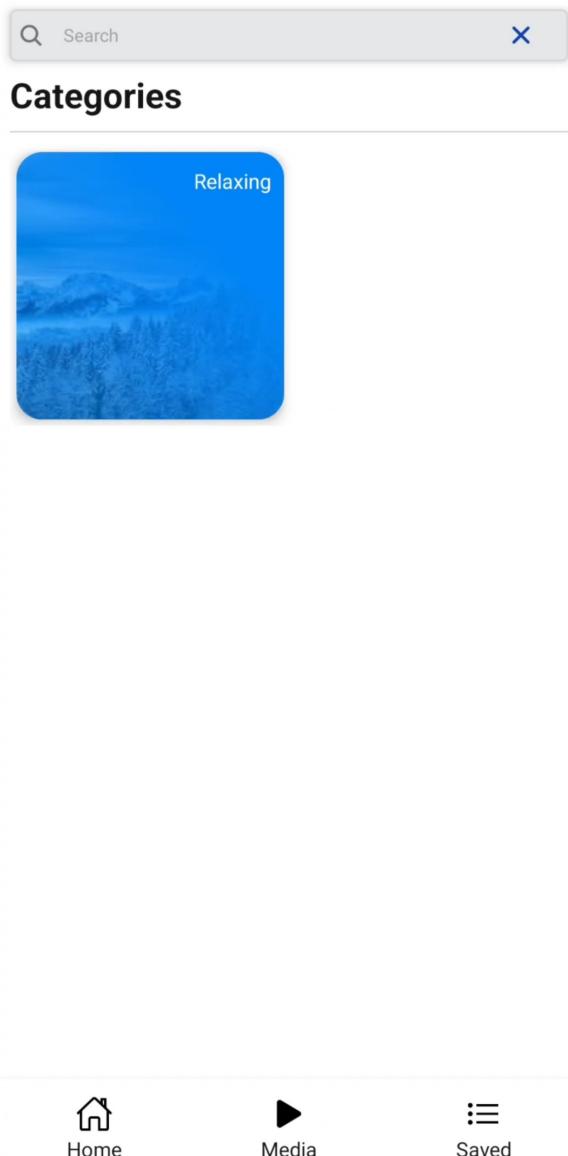


Abbildung 12: Kategorien

## 10.6 Suchleiste

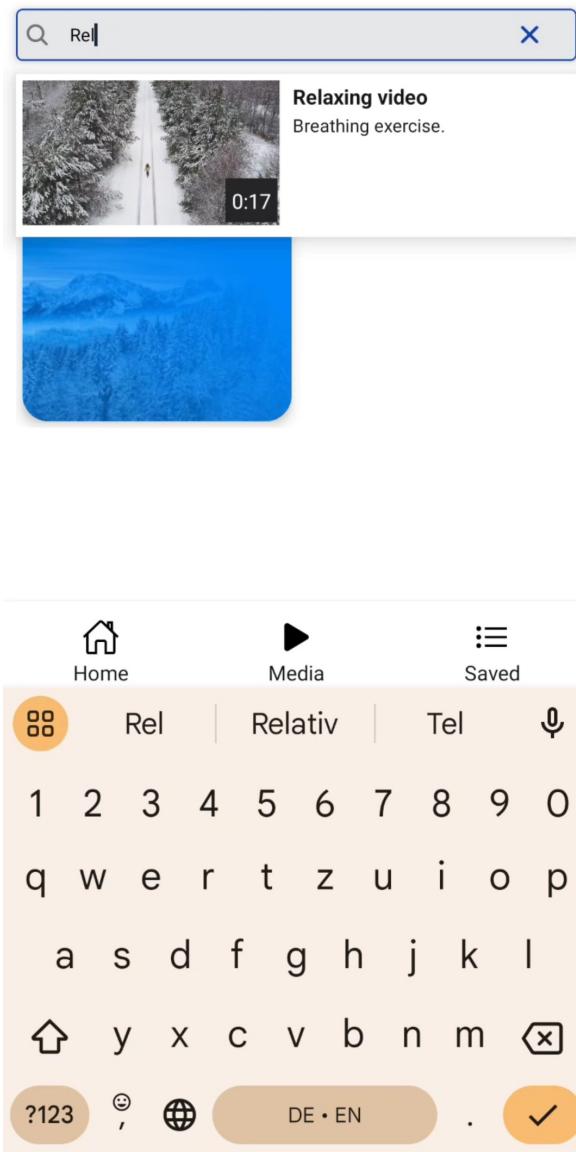


Abbildung 13: Suchleiste

## 10.7 Media Player

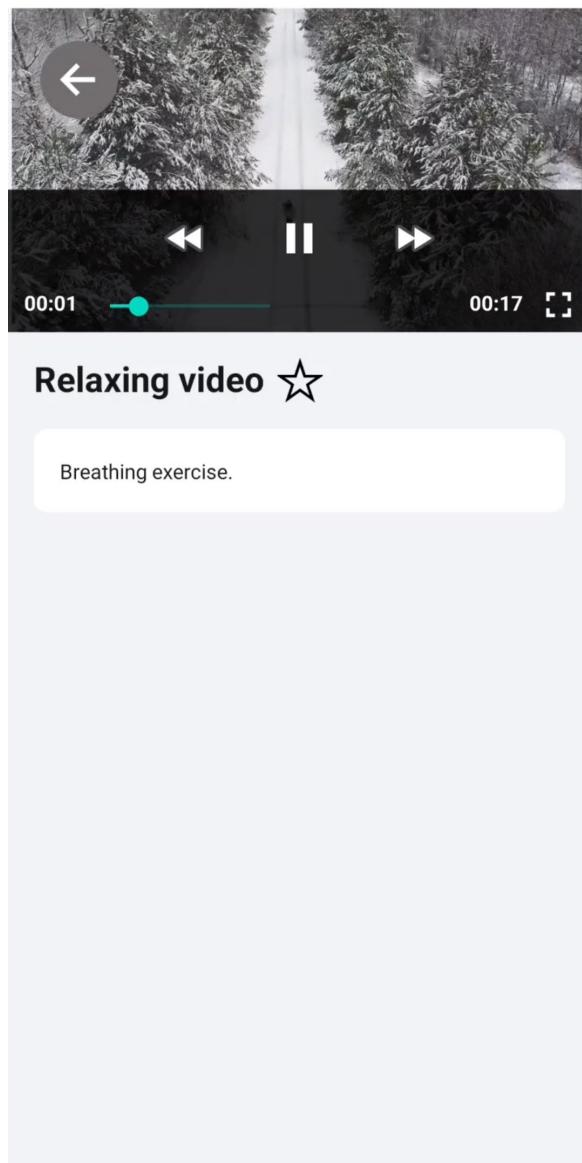


Abbildung 14: Eingepflegtes Video aus dem Backend

## 10.8 Favoriten



Home



Media



Saved

Abbildung 15: Kein Favorit gesetzt



Home



Media



Saved

Abbildung 16: Ein Favorit gesetzt

## 10.9 Einstellungen

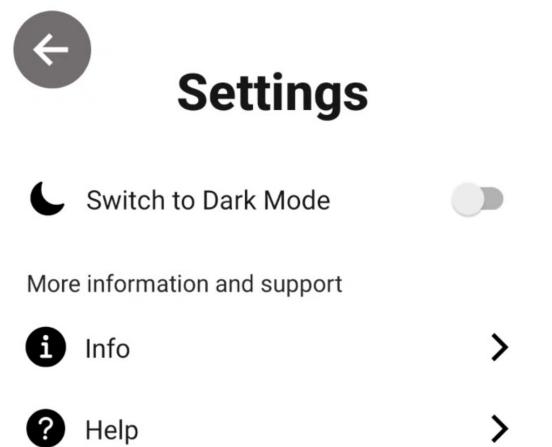


Abbildung 17: Einstellungen

## 10.10 Light/Dark Mode

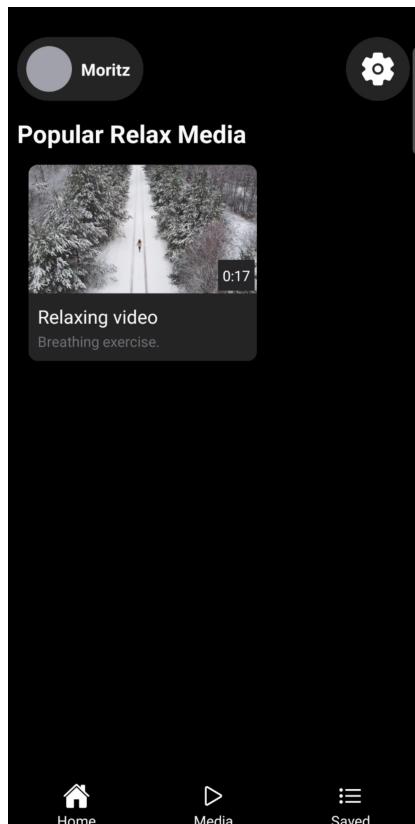


Abbildung 18: Dark Home-Screen

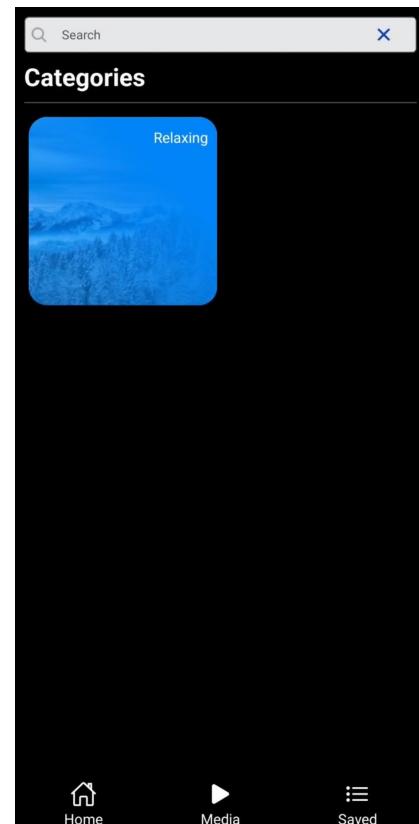


Abbildung 19: Dark Kategorie-Screen

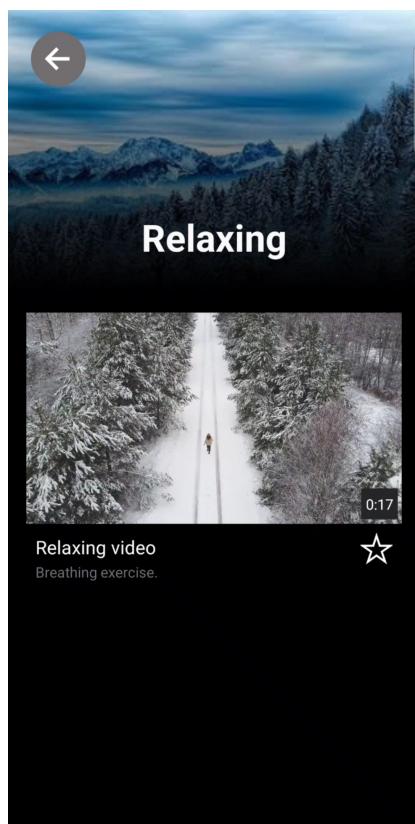


Abbildung 20: Dark Media-Screen

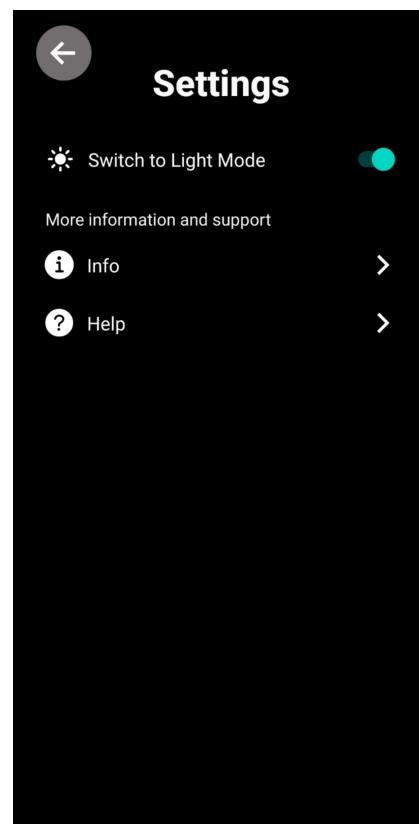


Abbildung 21: Dark Settings-Screen

## **10.11 Logo**

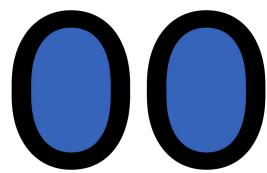


Abbildung 22: Logo white

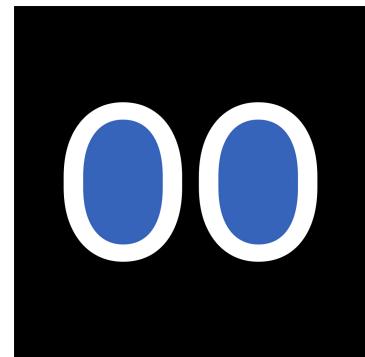


Abbildung 23: Logo black

# 11 Implementierung

## 11.1 Entity Relationship Diagram (ERD)

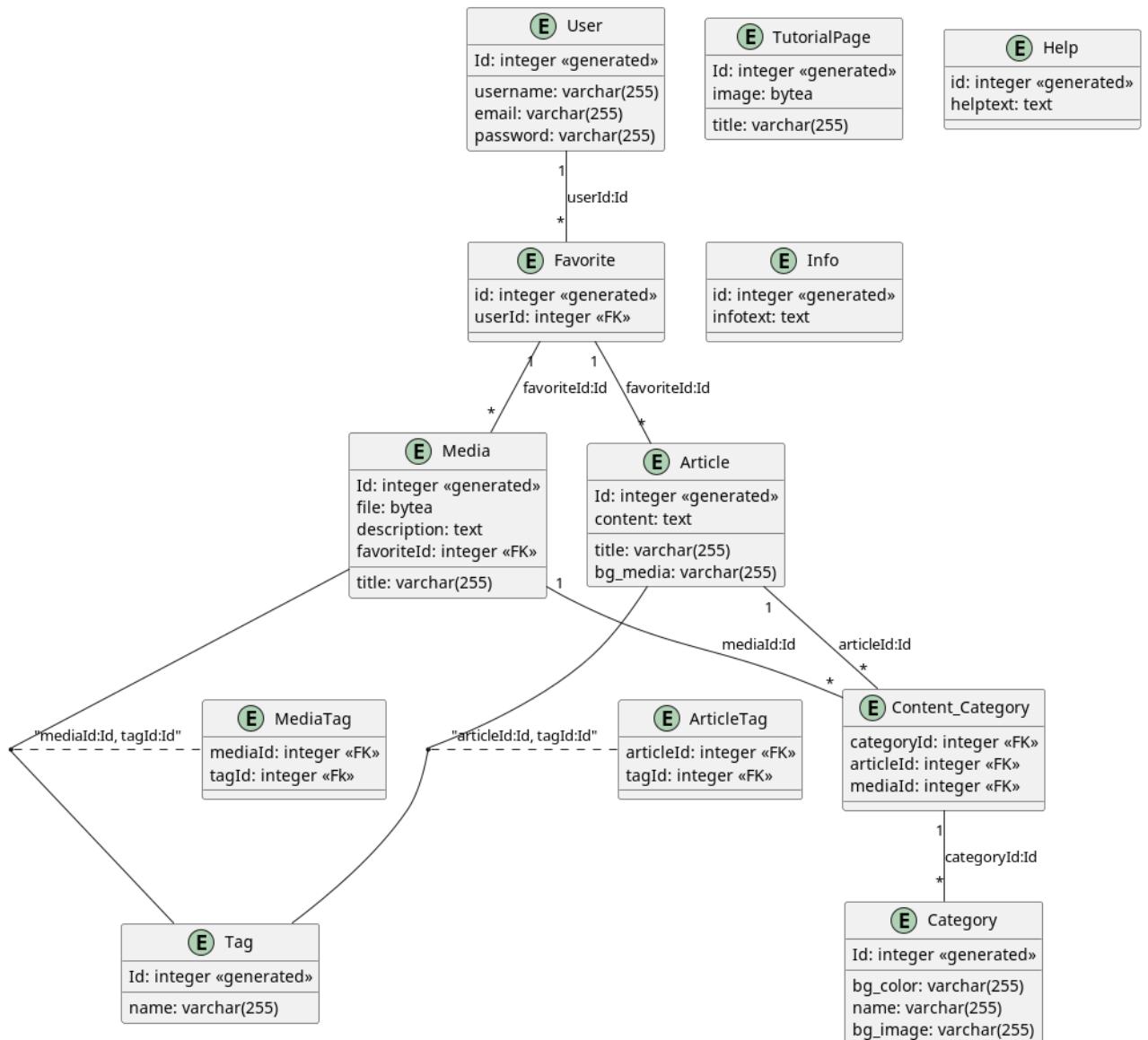


Abbildung 24: ERD

## 11.2 Medias und Articles

Grundsätzlich wurde als Team entschieden, die Artikel und die Medien in zwei separaten Tabellen zu speichern, da Artikel mehrere Bilder, Videos und Audios Inhalte enthalten

können. Diesen können dann mittels einem "Rich Text Editor" hinzugefügt werden. Bei "Media" handelt es sich nur um ein Medienelement, nämlich ein Bild, Video oder Audio File. Es ist aber wichtig anzumerken, dass keine Benutzeroberfläche für die Artikel implementiert wurde, da der Kunde diese nicht in dem **Minimal Viable Product (MVP)** haben wollte. Für die Realisierung des kompletten Datenmodels war aber das Einfügen der Artikel erforderlich.

## 11.3 TutorialPage

Die Entität "TutorialPage" hat keine Beziehungen zu anderen Entitäten, da sie nur für den Inhalt des Tutorial-Slideshows, die nach der erfolgreichen Installation der App angezeigt wird, gedacht ist.

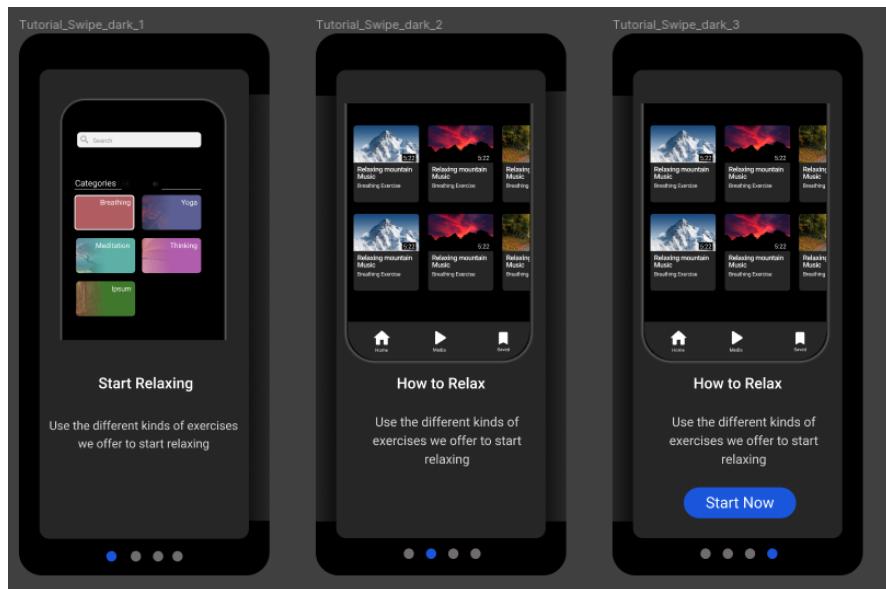


Abbildung 25: Screenshot aus dem UI Prototyp von Relaxoon

Mit dem Persistieren der Inhalte von der Slideshow könnte die Erstellung von neuen Builds bei jeder Änderung vermieden werden.

## 11.4 Suche und Filterungen

Bei den Filterungen wurde der "Interactive Query Builder" verwendet, um die Filterparameter der Abfragen zu generieren und diese dann bei den REST Abfragen anzuhängen. Es gibt auch eine "Query Engine", welche die gleiche Funktion wie diese "Interactive Query Builder" hat. Diese kann in dem **Object Relational Mapper (ORM)** angewendet

werden. Mit der Verwendung dieser "Interactive Query Builder" kann man die Daten, die man von den REST-Resourcen bekommt, anpassen. Es gibt auch die Möglichkeit, die REST-Ressource im Backend mittels dieser "Query Engine" anzupassen. Allerdings wurde diese "Query Engine" nicht verwendet, da die Dokumentation bzw. IntelliSense von dem ORM sehr ungenau war.

The screenshot shows a code editor with dark mode styling. A tooltip is displayed over the word '\$containsi' in the code, indicating it is a type alias for 'string'. The code itself is a function named 'queryParamFilter' that uses the qs library's stringify method to generate a query object. The query object includes filters for 'de', 'title', 'categories', and 'tags' properties, each with their own '\$containsi' filter. The code is well-formatted with color-coded syntax (blue for keywords, purple for objects, etc.).

```
const queryParamFilter = qs.stringify({
  filters: {
    $or: [
      {
        de: (property) $containsi: string
        $containsi: text
      },
      {
        title: {
          $containsi: text
        }
      },
      {
        categories: {
          name: {
            $containsi: text
          }
        }
      },
      {
        tags: {
          name: {
            $containsi: text
          }
        }
      }
    ]
  }
}, { encodeValuesOnly: true })
```

Abbildung 26: Beispiel für den Interactive Query Builder

**API-Route mit den angehängten generierten Abfrageparametern:**  
`/medias?&populate[tags]=true &populate[file]=true &populate[favorite][populate]=users_permissions_user`

## 11.5 State Management

“ Der Begriff State ist in React-Applikationen überall präsent. Generell bezeichnet State den Zustand einer Komponente, also die dynamischen Daten, die eine Komponente den Benutzer:nnen anzeigt. Eine Änderung am State führt dazu, dass die Komponente neu gerendert wird und so die Datenänderung für die Benutzer:in sichtbar wird. Im einfachsten Fall verwaltet jede Komponente ihren eigenen State. Es gibt Fälle, in denen es jedoch erforderlich wird, den State zwischen mehreren Komponenten zu teilen. ” [19]

Um die Zustände der Applikation zu behandeln wurde meistens mit dem “useState Hook” von React gearbeitet. Der Einsatz einer State Management Library wie zum Beispiel ”RXJS” oder ”Redux” war gar nicht nötig, da die Anzahl der verwendeten UI-Komponenten nicht so groß ist.

### 11.5.1 useState Hook

“useState is a React Hook that lets you add a state variable to your component.” [20]

## 11.6 Dark/Light Mode

Da das Team sich für die Verwendung von der UI-Library ”Native Base” entschieden hat, war die Realisierung von diesem Feature sehr einfach. Die Komponenten von Native Base haben die Properties `_light` und `_dark`, wo man die Farbe einer Komponente je nach Modus definieren kann. Native Base verwendet im Hintergrund einen `useContext` Hook, wo die Modi der Applikation gespeichert werden.

### 11.6.1 useContext Hook

“ useContext is a React hook that provides a way to share data (context) across multiple components without explicitly passing it through props. It is part of the React Context API, which is built into the React library. ” [21]

## 11.7 Help und Info Screen

Mit der Speicherung der Inhalte von den Ansichten "Help" und "Info" ist ein neues Build für das Frontend nicht mehr nötig. Somit kann der Content-Manager mehrere Freiheiten haben. Diese Inhalte wurden mit sogenannten "Single Types" persistiert. Ein "Single Type" ist nichts anders als eine Tabelle, die nur eine Zeile enthält. Bei einer Änderung des Werts von diesem "Single Type" wird diese eine Zeile aktualisiert.

## 11.8 Authentifizierung

Für die Authentifizierung wurde mit Json Web Token (JWT) gearbeitet. Die Authentifizierungslogik im Server ist im Strapi eingebaut. Für den Client wurde die Logik so umgesetzt, dass der User die App erst verwenden kann, wenn er bereits authentifiziert ist. Die Ansicht fürs Login soll aber nicht angezeigt werden, wenn der User bereits authentifiziert ist. Um zu überprüfen, ob der Nutzer noch eingeloggt ist oder nicht wurde der Token im AsyncStorage gespeichert. Dieser ist ähnlich zu dem localStorage in der Webentwicklung. Somit wurde bei der Öffnung der App geprüft, ob der Token valid und nicht abgelaufen ist. Damit der User auf die anderen Screens nicht zugreifen kann, wenn er nicht eingeloggt ist, wurde eine boolische Zustandsvariable verwendet, welche das LoginScreen als die zweite Ebene unserer Stack-Navigation rendert, wenn der Token invalid oder abgelaufen ist. Die erste Ebene ist für den TutorialScreen reserviert. Im folgenden Codestück ist die ganze Logik der Authentifizierung zu finden:

Listing 1: protected screens

```

1 import { createBottomTabNavigator } from '@react-navigation/bottom-tabs';
2 import { createStackNavigator } from '@react-navigation/stack';
3 import { useColorMode } from 'native-base';
4 import * as React from 'react';
5 import { useEffect, useState } from 'react';
6 import { jwtDecode } from 'jwt-decode';
7 import CategoryScreen from './screens/CategoryScreen';
8 import HomeScreen from './screens/HomeScreen';
9 import MediaScreen from './screens/MediaScreen';
10 import SavedScreen from './screens/SavedScreen';
11 import SettingsScreen from './screens/SettingsScreen';

```

```

12 import VideoScreen from './screens/VideoScreen';
13 import WaitScreen from './screens/WaitScreen';
14 import HelpScreen from './screens/setting_tabs/HelpScreen';
15 import InfoScreen from './screens/setting_tabs/InfoScreen';
16 import { getToken } from './util/getUser';
17 import TabBar from './components/TabBar';
18 import TutorialScreen from './screens/intro/TutorialScreen';
19 import LoginScreen from './screens/registration/LoginScreen';
20 import RegistrationForm from './screens/registration/RegistrationForm';
21 import { colors } from './styles/ThemeColors';
22 import AsyncStorage from '@react-native-async-storage/async-storage';
23 // Screen names
24 type PageSettings = Record<string, { iconName: string; screen:
25   React.ComponentType<any> }>;
26 const settingsForScreenName: PageSettings = {
27   Home: { iconName: 'home', screen: HomeScreen },
28   Media: { iconName: 'play', screen: CategoryScreen },
29   Saved: { iconName: 'list', screen: SavedScreen }
30 };
31 const Tab = createBottomTabNavigator();
32 const Stack = createStackNavigator();
33
34 function MainContainer(): JSX.Element {
35   const { colorMode } = useColorMode();
36   const [tokenExists, setTokenExists] = useState<boolean>(false);
37   useEffect(() => {
38     getToken()
39       .then((data) => {
40         console.info(typeof data);
41         if (!data) throw new Error('no token');
42         const decodedToken: { iat: number; exp: number; id: number } =
43           jwtDecode(data);
44         const eighthoursAfterNow = new Date(new Date().getHours() +
45           8).getMilliseconds();
46         if (decodedToken.exp < eighthoursAfterNow) {
47           setTokenExists(false);
48           AsyncStorage.clear();
49           return;
50         }
51         setTokenExists(true);
52       })
53       .catch((e) => {
54         console.info('no token', e);
55         setTokenExists(false);
56       });
57     console.info(tokenExists);
58   }, []);
59
60   return (
61     <Stack.Navigator
62       screenOptions={{ headerShown: false }}
63     >
64       {/* */
65       <Stack.Screen name={"WaitScreen"} component={WaitScreen}/>
66     */}
67     {/* */
68     {!tokenExists && <Stack.Screen name="Tutorial"
69       component=[TutorialScreen]/>}
70     */}
71     {!tokenExists && <Stack.Screen name="Login" component={LoginScreen} />}
72     {!tokenExists && <Stack.Screen name="Registration"
73       component=[RegistrationForm] />}
74     <Stack.Screen
75       name="Main"
76       options={{ headerShown: false }}
77     >
78       {() => (
79         <Tab.Navigator
80           screenOptions={{ headerShown: false,
81             headerTitleStyle: {
82               fontSize: 16,
83               color: colorMode === 'dark' ? colors.fontDark : colors.fontLight
84             }
85           }}
86         >
87           tabBar={(props) => <TabBar color={colorMode} {...props} />}
88         >

```

```

89         {Object.entries(settingsForScreenName).map(([pageName, settings]) =>
90           (
91             <Tab.Screen key={pageName} name={pageName}
92               component={settings.screen} />
93           ))
94         </Tab.Navigator>
95       </Stack.Screen>
96       <Stack.Screen name="Content" component={MediaScreen} />
97       <Stack.Screen component={VideoScreen} name="Video" />
98       <Stack.Screen component={SettingsScreen} name="Settings" />
99       <Stack.Screen name="Info" component={InfoScreen} />
100      <Stack.Screen name="Help" component={HelpScreen} />
101    </Stack.Navigator>
102  );
103}
104 export default MainContainer;

```

## 11.9 Validierung

Strapi an sich hat eine eingebaute Validierung für Emails und Passwörter. Das Team hat aber eine Client-seitige Validierung implementiert, um die Anzahl der Requests zu minimieren und eine bessere UX zu schaffen. Für die Client-seitige Validierung wurde eine Library namens "Zod" verwendet. Diese verwendet Builder Pattern um ein Validierungsschema zu bauen.

Listing 2: Validierungsschemen

```

1 import { z } from 'zod';
2
3 export const nameValidator = z.string().min(3);
4 export const emailValidator = z.string().email();
5 export const passwordValidator = z.string().min(6);

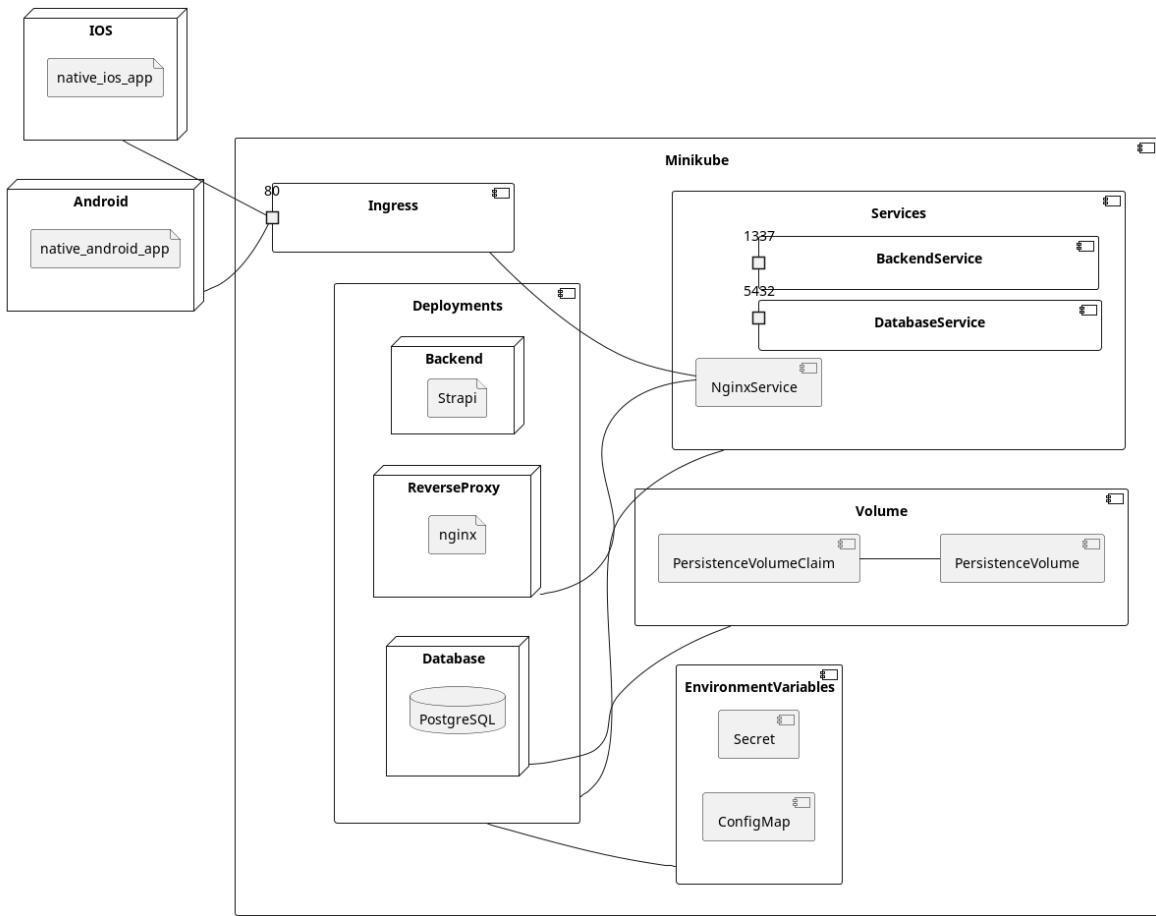
```

## 11.10 Deployment

### 11.10.1 Allgemeins

Die Veröffentlichung der Applikation im Play Store und im App Store wurde von dem Team nicht verlangt. Das Hochladen der Android-Applikation auf Firebase App Distribution war jedoch eine essentielle Aufgabe. Das Backend wurde auf den Servern von den Firmen Solvistas und Macolution deployt. Für Demonstrationszwecke wurde ein lokales Deployment auf Minikube erstellt.

### 11.10.2 Deployment Diagram



### 11.10.3 Deployment von dem Backend auf Minikube

Damit ein funktionsfähiges Deployment erstellt werden kann sind diese Komponenten zu deployen:

- Strapi
- Datenbank

Für jede dieser genannten Bauteile des Backends sind Kubernetes (K8S) Deployments und Kubernetes Services zu erstellen.

### 11.10.4 K8S-Konfigurationen für die Datenbank

Zusätzlich zu einer Deployment- und Service-Komponente ist für die Sicherung von den Credentials der Datenbank eine sogenannte Secret-Komponente gebraucht. Für die Vermeidung von Datenverlusten in Fällen wie Neustart eines Pods oder die Aktualisierung der Konfiguration von dem Pod ist das Anglegen einer PersistenceVolume-Komponente

sehr dringend. Damit das Deployment auf die PersistenceVolume-Komponente zugreifen kann, ist eine sogenannte PersistenceVolumeClaim-Komponente zu definieren.

Für die Konfiguration der Secret-, Service- und Deployment-Komponente wurde die kubectl Command Line Interface verwendet und für die PersistenceVolume-Komponente bzw. die PersistenceVolumeClaim-Komponente wurden fertige Konfigurationen genommen.

Listing 3: K8S PVC

```

1  apiVersion: v1
2  kind: PersistentVolumeClaim
3  metadata:
4    finalizers:
5      - kubernetes.io/pvc-protection
6    name: strapi-pvc
7    namespace: default
8  spec:
9    accessModes:
10      - ReadWriteMany
11    resources:
12      requests:
13        storage: 10Mi
14    storageClassName: standard

```

Listing 4: K8S PV

```

1  apiVersion: v1
2  kind: PersistentVolume
3  metadata:
4    finalizers:
5      - kubernetes.io/pv-protection
6    labels:
7      type: local
8    name: strapi-volume
9    resourceVersion: "33077"
10   uid: ae6d772a-0090-4074-b3ac-1edb929daf29
11
12  spec:
13    accessModes:
14      - ReadWriteOnce
15    capacity:
16      storage: 10Gi
17    hostPath:
18      path: /mnt/data
19      type: ""
20    persistentVolumeReclaimPolicy: Retain
21    storageClassName: manual
22    volumeMode: Filesystem
23    status:
24      phase: Available

```

## 11.10.5 K8S-Konfigurationen für Strapi

Für Strapi wird auch eine Secret-Komponente benötigt, da der Server geheime Umgebungsvariablen benötigt. Es gibt auch einige. Damit das Deployment von Strapi erstellt werden kann, muss die Anwendung zuerst in ein Docker Image umgewandelt werden. Dies kann mit der Verwendung des folgenden Dockerfiles erfolgen.

Listing 5: Strapi Dockerfile

```

1  # Creating multi-stage build for production

```

```

2   FROM node:18-alpine as build
3     RUN apk update && apk add --no-cache build-base gcc autoconf automake zlib-dev
4       libpng-dev vips-dev git > /dev/null 2>&1
5     ARG NODE_ENV=production
6     ENV NODE_ENV=${NODE_ENV}
7
8     WORKDIR /opt/
9     COPY package.json package-lock.json .
10    RUN npm install -g node-gyp
11    RUN npm config set fetch-retry-maxtimeout 600000 -g && npm install
12      --only=production
13    ENV PATH /opt/node_modules/.bin:$PATH
14    WORKDIR /opt/app
15    COPY . .
16    RUN npm run build
17    # Creating final production image
18    FROM node:18-alpine
19    RUN apk add --no-cache vips-dev
20    ARG NODE_ENV=production
21    ENV NODE_ENV=${NODE_ENV}
22    ENV HOST=0.0.0.0
23    ENV PORT=1337
24    ENV APP_KEYS=secret
25    ENV API_TOKEN_SALT=secret
26    ENV ADMIN_JWT_SECRET=secret
27    ENV TRANSFER_TOKEN_SALT=secret
28    # Database
29    ENV DATABASE_CLIENT=postgres
30    ENV DATABASE_HOST=localhost
31    ENV DATABASE_PORT=5432
32    ENV DATABASE_NAME=strapi
33    ENV DATABASE_USERNAME=strapi
34    ENV DATABASE_PASSWORD=strapi
35    ENV DATABASE_SSL=false
36    ENV JWT_SECRET=cVRog3q5woTNB8EJ+vKPFA==
37
38
39
40
41    WORKDIR /opt/
42    COPY --from=build /opt/node_modules ./node_modules
43    WORKDIR /opt/app
44    COPY --from=build /opt/app ./
45    ENV PATH /opt/node_modules/.bin:$PATH
46
47    RUN chown -R node:node /opt/app
48    USER node
49    EXPOSE 1337
50    CMD [ "npm", "run", "start" ]

```

Dieses Image soll dann in einem Container Registry wie zum Beispiel Dockerhub oder Github Container Registry hochgeladen werden, damit die K8S-Deployment-Komponente dieses Image in Einsatz nimmt.

## 11.10.6 K8S-Secrets

Damit man Umgebungsvariablen in einem K8S-Cluster definieren kann gibt es zwei Möglichkeiten. Man kann entweder eine ConfigMap-Komponente verwenden oder eine Secret-Komponente. Es gibt auch die Möglichkeit, diese Umgebungsvariablen direkt in den Konfigurationen von der Deployment-Komponente händisch einzutragen. Für geheime bzw. sensible Daten werden K8S-Secrets verwendet. Das Anlegen der benötigten Secrets für unsere Anwendung erfolgte durch die Verwendung folgender Befehle:

Listing 6: Secrets für Strapi

```

1   kubectl create secret generic strapi-server-secret \
2   --from-literal=PORT=1337 \
3   --from-literal=APP_KEYS=secret
4   --from-literal=API_TOKEN_SALT=secret \
5   --from-literal=ADMIN_JWT_SECRET=secret \
6   --from-literal=TRANSFER_TOKEN_SALT=secret \
7   --from-literal=DATABASE_CLIENT=postgres \
8   --from-literal=DATABASE_PORT=5432 \
9   --from-literal=DATABASE_NAME=secret \
10  --from-literal=DATABASE_USERNAME=secret \
11  --from-literal=DATABASE_PASSWORD=secret \
12  --from-literal=DATABASE_SSL=false \
13  --from-literal=JWT_SECRET=secret

```

Listing 7: Secrets für die Datenbank

```

1 kubectl create secret generic strapi-secret \
2 --from-literal=POSTGRES_USER=strapi \
3 --from-literal=POSTGRES_PASSWORD=strapi \
4 --from-literal=POSTGRES_DB=strapi

```

### 11.10.7 Was ist eine Deployment-Komponente

Eine Deployment-Komponente dient dazu, dass ein Pod erstellt wird und die benötigten Ressourcen bzw. Volumes und Zugriffsrechten dafür definiert werden.[22]

Für das Anlegen einer Deployment-Komponente muss man zuerst die Anwendung containerisieren. Für weit verbreitete Software wie z.B.: PostgreSQL gibt es bereits viele vorhandene Images auf unterschiedliche Container-Registries.

Folgende Befehle wurden verwendet, um die Deployment-Komponenten für Strapi und PostgreSQL zu erstellen:

Listing 8: create k8s deployments

```

1 kubectl create deployment relaxoon-db --image=postgres:12.16-bullseye --port=5432
2 kubectl create deployment relaxoon-strapi
    --image=ghcr.io/Abdulrahman-AL-Sabagh/relaxoon-strapi:latest --port=8080

```

### Was ist ein Pod

“ Ein Pod (übersetzt Gruppe/Schote, wie z. B. eine Gruppe von Walen oder eine Erbsenschote) ist eine Gruppe von einem oder mehreren Containern mit gemeinsam genutzten Speicher- und Netzwerkressourcen und einer Spezifikation für die Ausführung der Container. Die Ressourcen eines Pods befinden sich immer auf dem gleichen (virtuellen) Server, werden gemeinsam geplant und in einem gemeinsamen Kontext ausgeführt. Ein Pod modelliert einen anwendungsspezifischen ”logischen Server”: Er enthält eine

oder mehrere containerisierte Anwendungen, die relativ stark voneinander abhängen.“ [23]

### 11.10.8 Erstellung einer Service-Komponente

Mit dem Einsatz eines Services in Kubernetes können Pods, die sich im gleichen Cluster befinden, miteinander kommunizieren. [24]

Für das Anlegen einer Service-Komponente kann man diesen Befehl nutzen:

Listing 9: create a service component

```
1 kubectl expose deployments/<Name des Pods> --port=5432
```

### 11.10.9 Erstellung einer Ingress-Komponente

Damit der Emulator, der sich außerhalb des lokalen Clusters von Minikube befindet, mit dem deployten Backend kommunizieren kann, ist das Anlegen einer Ingress notwendig.

//TODO Config für die Ingress-Komponente noch hingeben

//TODO Dieses Kapitel noch fertigschreiben

“ In einem Computernetzwerk befindet sich ein einfacher Reverse-Proxy zwischen einer Gruppe von Servern und den Clients, die sie verwenden wollen. Als Client gilt jede Hardware oder Software, die Anfragen an einen Server senden kann. [...]. Der Reverse-Proxy fängt alle Anfragen von den Clients an die Server ab und liefert auch alle Antworten und Dienste von den Servern wieder an die Clients zurück. Aus Sicht des Kunden sieht es so aus, als würde alles von einer einzigen Stelle ausgehen.

” [25]

### 11.10.10 Buildprozess für das Frontend

Damit man das Frontend deployt, muss man zuerst den nativen Android und IOS Code generieren. Danach sollen die Android bzw. IOS Anwendungen mit einer IDE oder mit der Kommando-Zeile gebaut werden. Um den nativen Code zu generieren ist Folgendes einzugeben:

Listing 10: generate android and ios

```
1 npx expo prebuild
```

### 11.10.11 Erstellung einer .apk Datei

Für Android wurde die .Android PacKage (apk) mit dem Einsatz von gradle Wrapper generiert. Folgendes muss installiert und richtig konfiguriert werden, damit eine .apk Datei generierbar ist:

- Java 11
- Installation von sdkmanager
- Lizenzen von sdkmanager müssen akzeptiert sein
- Installation von einer Android SDK
- Konfigurationen für JAVA\_HOME und ANDROID\_HOME
- Installation einer CLI namens "ninja"

Der Generierungsbefehl für die Build-Datei schaut dann wie folgendes aus:

Listing 11: generate apk

```
1 ./gradlew assembleRelease
```

## 11.11 Hochladen einer Mobileapp auf Firebase App Distribution

Nachdem Anlegen eines Accounts bei Firebase sind folgende und die Erstellung eines Projektes Schritte zu machen:

Damit die benötigten Firebase Services für die Applikationen aktiviert werden, muss man zuerst die benötigten Konfigurationen eingeben.

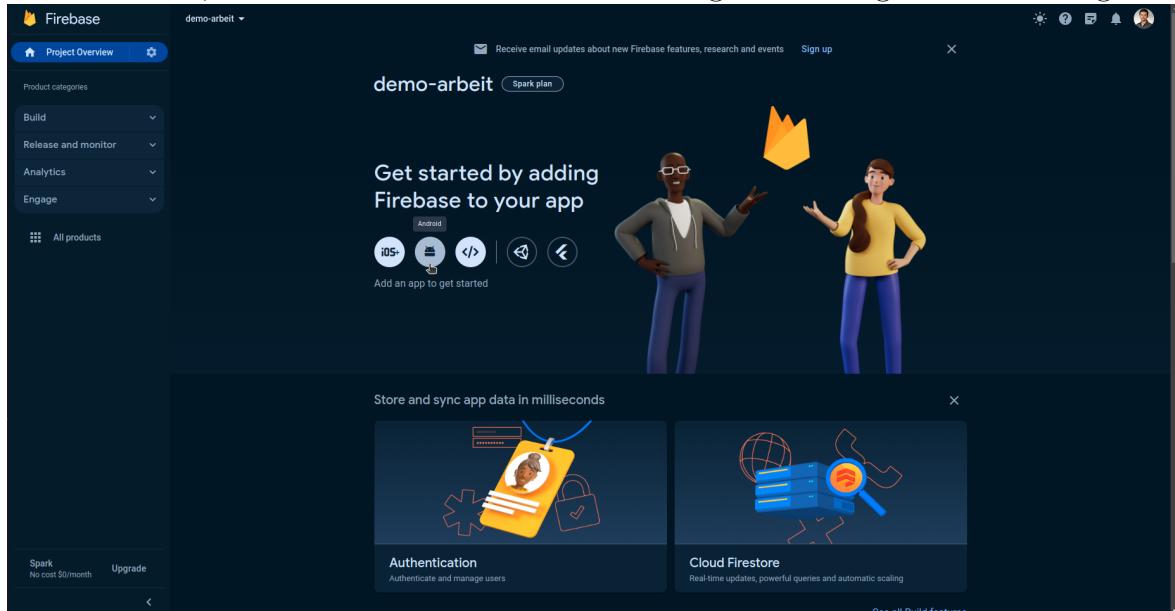


Abbildung 27: create android config

Folgende Daten müssen eingegeben werden, damit die Firebase Services für die Android Applikation eingeschaltet werden können.

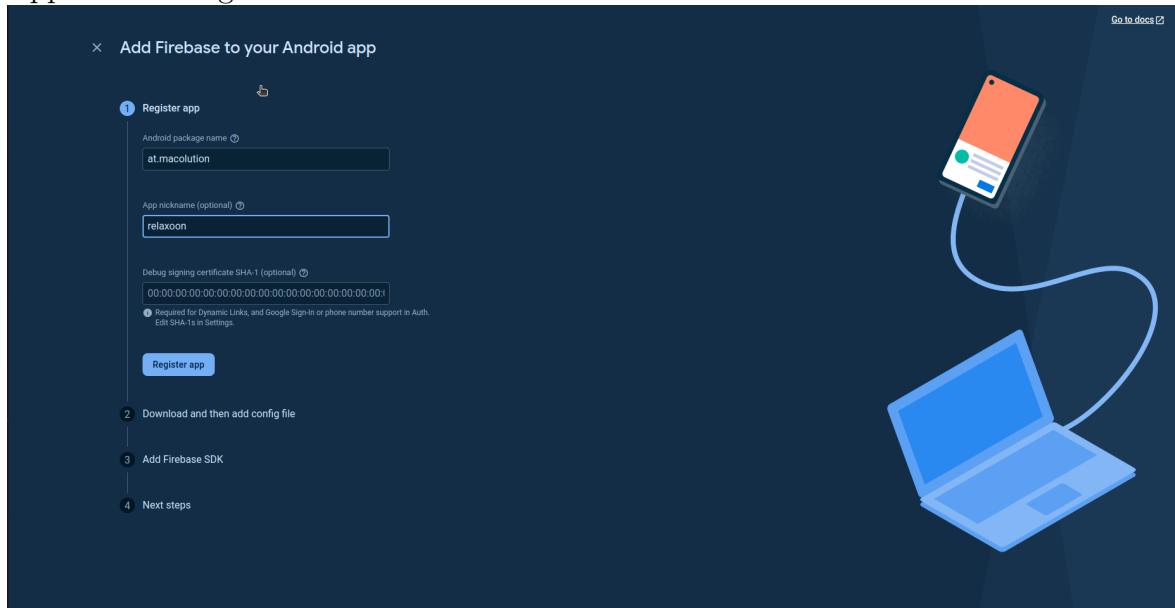


Abbildung 28: Android Config

Die restlichen Punkte sind für das Projekt Relaxoon irrelevant, da die Firebase-SDK in der vorliegende Arbeit nicht eingesetzt wird.

Danach soll man auf die App Distribution gehen und dann auf "Get Started" klicken

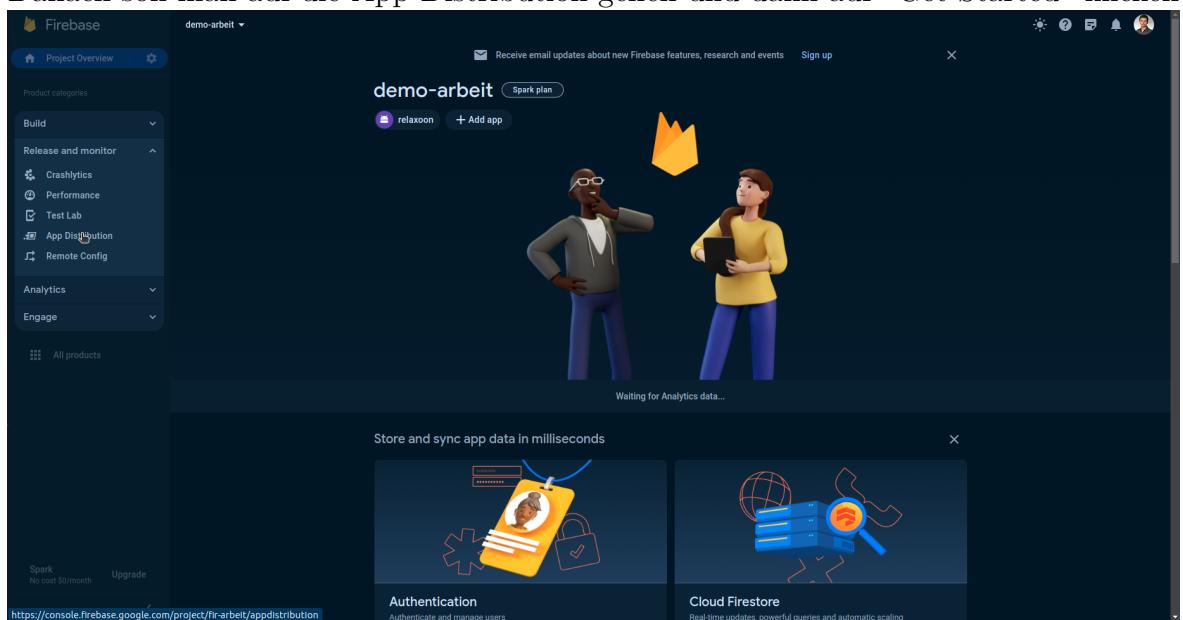


Abbildung 29: App Distribution

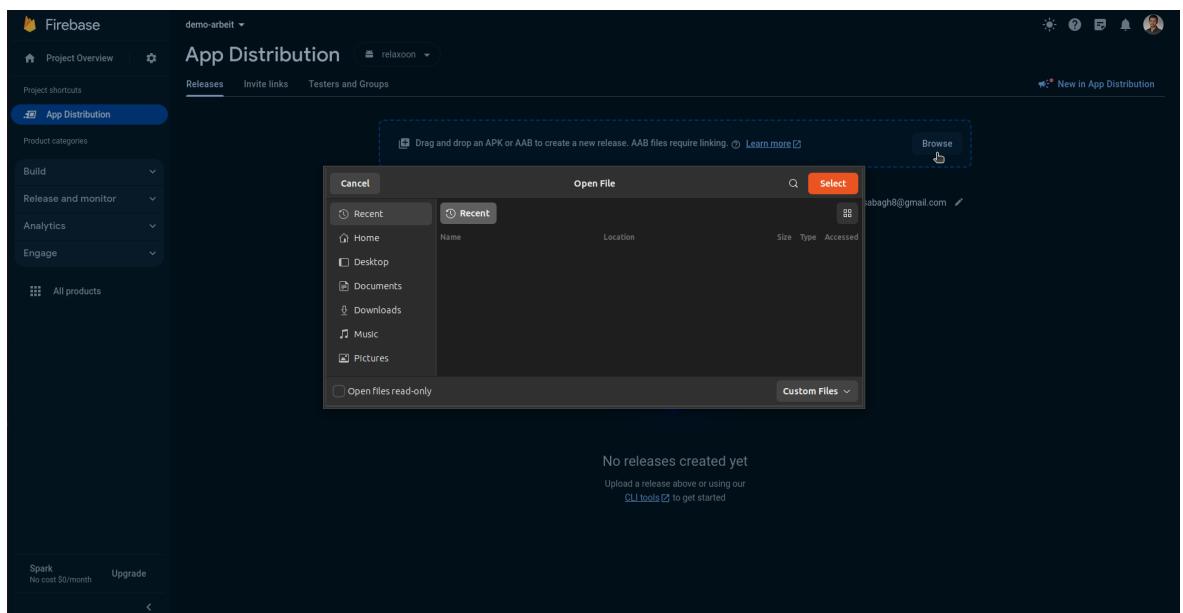


Abbildung 30: App Upload

Die App kann mit diesem "Browse Fenster" oder mit "Drag and Drop" hochgeladen werden

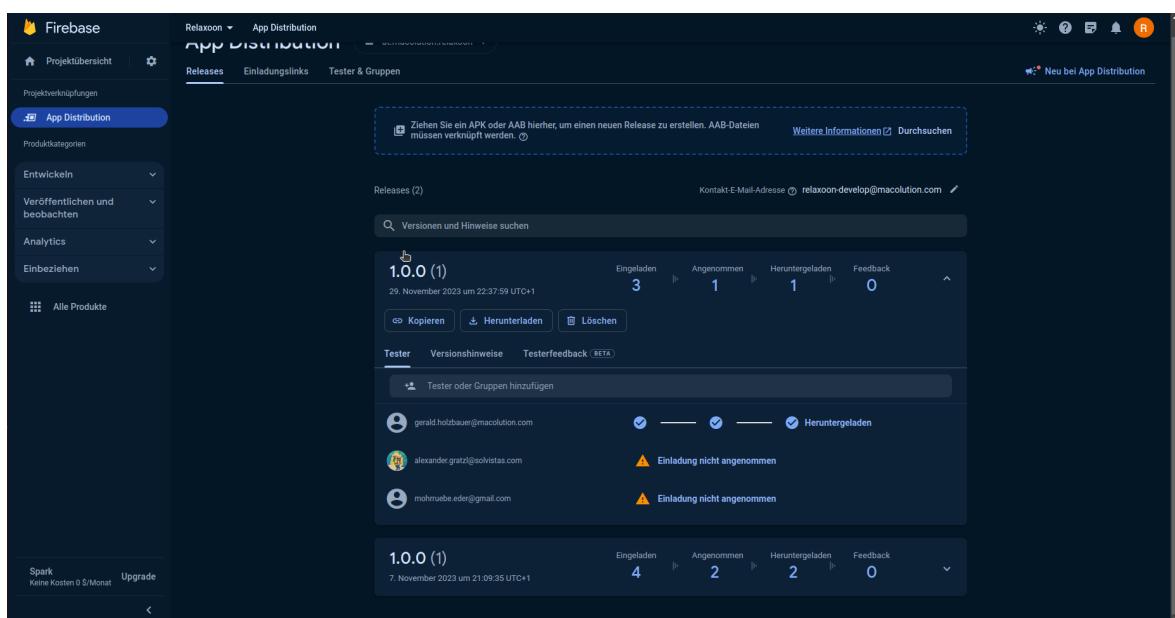


Abbildung 31: Releases

Das Produkt kann dann an andere Testuser verteilt werden, indem man die Email-Adressen dieser Tester beim Release eingibt.

# 12 Ausgewählte Aspekte und Probleme

## 12.1 Probleme mit localhost und https

Im Projekt wurde die Uniform Resource Locator (URL) für den (API)-Server in einem ".env-File" gespeichert. Für die lokale Entwicklung wurde Strapi auf dem lokalen Host immer verwendet. Bei IOS und Android ist das Holen der Daten von nicht sicheren Quellen beziehungsweise nur von HTTP gar nicht möglich. Auf Android gibt aber einige Ausnahmen für das Holen der Daten von einem lokalen Host. Bei Android kann man bei den generierten Build und Project Files einige Konfigurationen ändern. Das ist jedoch keine gute Lösung, da diese Files bei jedem Build geändert werden können. Außerdem können diese generierten Files nicht in das Version Control System (VCS) eingeccheckt werden. Bei Android muss man die Adresse 10.0.2.2 statt localhost verwenden.[26] Bei IOS gibt es keine einzige Möglichkeit, um Daten aus HTTP-Quellen oder aus dem localhost zu holen. Als Lösung wurde in der lokalen Entwicklung eine Command Line Interface (CLI) namens "ngrok" verwendet, welche den lokalen Port des Servers in das Internet mittels eines Tunnels weiterleitet und eine HTTPS-Adresse für den weitergeleiteten Server generiert.

Das Problem tauchte wiederholt in der Testphase auf und war für die Entwickler nicht lösbar, da der Staging Server für das Backend von Relaxoon auf den Servern der Solvistas deployed ist. Die Entwickler verfügen über keine Zugriffsrechte auf diesem Server. Das Problem wurde mithilfe von den Netzwerkadministratoren von Solvistas gelöst. Es wurde ein "Let's-encrypt-Zertifikat" für den Staging Server eingebaut.

## 12.2 Inkompatible Libraries beim Build-Prozess

In der Entwicklungsphase wurden einige Libraries mittels Node Package Manager (npm) installiert. Einige Libraries waren mit der Node Version oder mit anderen Libraries nicht kompatibel. Eine Lösung wäre, das Label "--legacy-peer-deps" oder "--force" beim Befehl "npm install" anzuhängen. Während der Entwicklung wurde das Label "--legacy-peer-deps" immer beim "npm install" integriert und danach hat alles problemlos funktioniert. Später beim Deployment von Android ist das Problem bei der Generierung von Android apk bzw. Android App Bundle (aab) aufgetaucht. Die Apk bzw. Aab wurde erfolgreich generiert, aber die App war nicht funktionsfähig. Es wurde für das Finden des Problems auf Android ein Tool namens "logcat" verwendet, um die Fehlermeldung zu finden. Dieses Problem an sich ist in der Community sehr stark verbreitet und es gibt dafür mehrere Gründe und mehrere Lösungen. Unter diesem Githublink [27] werden unterschiedliche Gründe und mehrere Lösungsmöglichkeiten für das Problem beschrieben. Es wurde jede einzelne Lösungsmöglichkeit versucht und keine von diesen vorgeschlagenen Lösungen funktionierte. Aus Erfahrung hat man gewusst, dass der Codebase einige Libraries besitzt, die nicht mehr gebraucht bzw. nicht mehr verwendet werden. Beim Löschen dieser Libraries und der Wiederinstallation von den Modulen mittels "npm install" tauchte die "legacy-peer-deps" Meldung wieder auf. Zur

Problemlösung wurde ein Tool namens "depcheck" installiert, welches den Codebase scannst und dem Entwickler bekannt gibt, ob eine Library im Codebase verwendet wird oder nicht. [28] Nach dem Löschen aller nicht benötigten Libraries und die Generierung der Apk bzw. AAB hat die Applikation problemlos funktioniert.

### 12.2.1 npm install --legacy-peer-deps

"The '--legacy-peer-deps' flag is used when you encounter compatibility issues with peer dependencies while installing packages. Peer dependencies are required by a package but aren't automatically installed alongside it. In some cases, when a package has not been updated to support the latest version of its peer dependency, the installation may fail due to conflicting versions. Adding the '--legacy-peer-deps' flag allows npm to use an older, compatible version of the peer dependency, ensuring a successful installation." [29]

### 12.2.2 npm install --force

"The '--force' flag is a more drastic option and should be used with caution. It instructs npm to forcefully install packages, even if it encounters errors or conflicts. This can be useful in situations where you want to override any version or compatibility checks and forcibly install packages. However, it is important to note that using '--force' may lead to unexpected issues, such as breaking dependencies or introducing incompatibilities, so it should be used sparingly and with a good understanding of its consequences." [29]

## 12.3 Probleme mit Thumbnails

Relaxoon zeigt mehrere Videos für Antistress-Meditationen an. Damit diese schön darstellbar sind, muss jedes Video unbedingt ein Thumbnail haben. Der Content-Manager könnte aber beim Erstellen des Videos vergessen, ein Thumbnail für das Video zu erstellen. Auf der Adminoberfläche können keine Thumbnails zu dem Video hinzugefügt werden, da diese Aktion sehr schlechte User Experience verursacht. Man kann zwar eine Funktion implementieren, die nach dem Hochladen eines Videos ein Thumbnail mittels **Fast Forward Moving Picture Experts Group (ffmpeg)** aus dem ersten Bild des Videos erstellt. Diese Möglichkeit ist aber sehr aufwendig und nicht empfehlenswert, da die damalige Dokumentation von Strapi nicht sehr genau war. Außerdem könnte diese Alternative bei den Updates von Dependencies nicht mehr funktionsfähig sein. Darauf wurde eine Expo-Library gefunden, die im Frontend Thumbnails für die Videos erstellt. Die Verwendung davon war aber nicht sehr vorteilhaft, da der Generierungsprozess sehr langsam war. Als Lösung wurde im Code ein nicht abspielbares Videoelement deklariert, dass für den Zuschauer als Thumbnail dient. Das Video ist erst abspielbar, wenn man darauf klickt. Diese Lösung wurde mit den Konzepten "Lazy Loading" und "Suspenses" zusammen kombiniert, damit Loading-Spinners statt leere Flächen für den User angezeigt werden, wenn das Laden des Videos etwas länger dauert.

### 12.3.1 Suspense

“<Suspense> lets you display a fallback until its children have finished loading.” [30]

### 12.3.2 Lazy Loading

“lazy lets you defer loading component’s code until it is rendered for the first time.” [31]

## 12.4 Dauer von Videos

Beim Hochladen eines Videos in Strapi wird die Dauer des Videos nur auf der Oberfläche angezeigt. Diese Information ist aber in den REST-Schnittstellen der Meta-Daten von den Files nicht inkludiert. Bei diesem Problem ist die Verwendung von ffmpeg auch eine Lösungsmöglichkeit. Es wurde aber nicht mit ffmpeg gearbeitet (siehe obiges Problem). Für die Berechnung der Dauer wurde im Frontend das onLoadEvent, welches in den Properties (props) des Videoelements ist, verwendet, um die Dauer zu lesen und diese in das Format "mm:ss" umzuwandeln zu können.

## 12.5 Probleme mit useEffect und React-navigation Library

Für das Holen der Daten aus dem Server wurde die Fetch API verwendet. Diese wurde auch in einem ”useEffect Hook” eingegeben, damit die Daten bei jeder Aktualisierung eines spezifischen Zustandes aus dem Server geholt werden können. Beim Anklicken der unterschiedlichen Navigationsbuttons wurde bemerkt, dass die Daten sich gar nicht ändern. Am Anfang wurde vermutet, dass das Problem ein Caching Problem sein könnte. Später wurde herausgefunden, dass die Library ”React-navigation” den ”useEffect Hook” im Hintergrund deaktiviert. Für das Holen der Daten ist die Übergabe eines sogenannten ”useCallback Hooks” als Parameter in einem custom hook namens ”useFocusEffect” von React-navigation notwendig.[32]

### 12.5.1 Hooks

“Hooks let you use different React features from your components. You can either use the built-in Hooks or combine them to build your own. This page lists all built-in Hooks in React.” [33]

### 12.5.2 useEffect

“useEffect is a React Hook that lets you synchronize a component with an external system.” [34]

“Some components need to synchronize with external systems. For example, you might want to control a non-React component based on the React state, set up a server connection, or send an analytics log when a component

appears on the screen. Effects let you run some code after rendering so that you can synchronize your component with some system outside of React.[...] Effects let you specify side effects that are caused by rendering itself, rather than by a particular event.” [35]

### 12.5.3 useCallback

“useCallback is a React Hook that lets you cache a function definition between re-renders.” [36]

### 12.5.4 useFocusEffect

“The useFocusEffect is analogous to React’s useEffect hook. The only difference is that it only runs if the screen is currently focused. The effect will run whenever the dependencies passed to React.useCallback change, i.e. it’ll run on initial render (if the screen is focused) as well as on subsequent renders if the dependencies have changed. If you don’t wrap your effect in React.useCallback, the effect will run every render if the screen is focused.” [37]

## 12.6 Relaxoon Plakat

Im Zuge des Projektes wurde mithilfe von Adobe Illustrator ein Plakat zur Repräsentation der mobilen App entwickelt.

### 12.6.1 Umsetzungsschritte

1. Ein Farbverlauf wurde bestimmt, der sich von der hellblauen Farbe #64C5DF (Hexadezimalschreibweise) bis zu einem dunkleren Blau #3A62AC streckt. Der Farbverlauf dient als eine Grundlage des Plakates.
2. Abgerundete Dreiecke sowie die abstrakte Form, welche am Rand des Plakats entlang geht, wurden in Kombination mit dem Farbverlauf erzeugt und platziert.
3. Die Logos von Relaxoon, der HTL Leonding, den Firmen Solvistas und Macolution, dem CMS Strapi und der Programmiersprache React Native wurden hinzugefügt.
4. Der Diplomarbeitsbetreuer und die Projektpartner wurden namentlich erwähnt.
5. Das Grundlayout eines Smartphones wurde erstellt und mit einem Entwurf des Tutorials der App versehen.
6. Fotos und Namen der App-Entwickler wurden eingefügt.
7. Mit einem kurzen Beschreibungstext wurde das Plakat fertiggestellt.



Abbildung 32: Plakat

# **13 Resümee**

Coming soon

# 14 Anhang

## 14.1 Prompt-Verlauf

### Eingabe:

Erkläre mir den Unterschied zwischen funktionalen und nicht funktionalen Anforderungen.

### KI-Modell:

Funktionale und nicht-funktionale Anforderungen sind zwei Hauptkategorien von Anforderungen in der Softwareentwicklung und anderen Bereichen des Projektmanagements. Hier sind die grundlegenden Unterschiede zwischen ihnen:

Funktionale Anforderungen: Beschreiben, welche Funktionen das System bereitstellen soll und welche Aufgaben es ausführen muss. Definieren die spezifischen Funktionen, Dienste oder Aufgaben, die das System erfüllen muss, um die Benutzeranforderungen zu erfüllen. Konzentrieren sich auf die "WasAspekte des Systems und beschreiben, welche Ergebnisse erwartet werden. Beispiel: "Das System muss dem Benutzer ermöglichen, sich mit einem Benutzernamen und Passwort anzumelden."

Nicht-funktionale Anforderungen: Beschreiben Eigenschaften des Systems, die nicht unbedingt Funktionen sind, sondern Qualitätsmerkmale, die das System erfüllen muss. Betreffen oft die "WieAspekte des Systems und definieren Qualitätsmerkmale wie Leistung, Sicherheit, Benutzerfreundlichkeit und Zuverlässigkeit. Berücksichtigen Aspekte wie Skalierbarkeit, Wartbarkeit, Zuverlässigkeit, Leistung und Benutzerfreundlichkeit. Beispiel: "Das System muss eine Antwortzeit von weniger als 2 Sekunden für Benutzeranfragen sicherstellen."

In einem Softwareprojekt sind funktionale Anforderungen eher auf die eigentlichen Aufgaben und Funktionen der Software ausgerichtet, während nicht-funktionale Anforderungen die Qualität und Leistung des Systems definieren. Beide Arten von Anforderungen sind wichtig, um ein vollständiges und erfolgreiches Produkt zu entwickeln, da sie gemeinsam die grundlegenden Spezifikationen und Erwartungen für das System festlegen.



# Literaturverzeichnis

- [1] Unbekannt, „VOLKSLEIDEN BURNOUT,” 2021. Online verfügbar: [https://www.promentesteiermark.at/blog/volksleiden-burnout-das-ausbrennen-wird-noch-immer-unterschaetzt/#:~:text=19%20Prozent%20der%20Proband\\*innen,Syndrom%20sollten%20keinesfalls%20ignoriert%20werden](https://www.promentesteiermark.at/blog/volksleiden-burnout-das-ausbrennen-wird-noch-immer-unterschaetzt/#:~:text=19%20Prozent%20der%20Proband*innen,Syndrom%20sollten%20keinesfalls%20ignoriert%20werden).
- [2] T. Aichinger, „MACOLUTION,” 2023. Online verfügbar: <https://www.solvistas.com/de/news/solvistas-group-waechst-weiter>
- [3] Unbekannt, „Stress.” Online verfügbar: [https://www.internisten-im-netz.de/fachgebiete/psyche-koerper/stress.html#:~:text=Zeichen%20von%20Nervosit%C3%A4t%20\(Z%C3%A4hneknirschen%20in,Stoffwechselst%C3%B6rungen%20Allergien%20und%20Entz%C3%BCndungskrankheiten%20f%C3%BCr%20Chren.\)](https://www.internisten-im-netz.de/fachgebiete/psyche-koerper/stress.html#:~:text=Zeichen%20von%20Nervosit%C3%A4t%20(Z%C3%A4hneknirschen%20in,Stoffwechselst%C3%B6rungen%20Allergien%20und%20Entz%C3%BCndungskrankheiten%20f%C3%BCr%20Chren.))
- [4] ——, „Ziele setzen,” 2023, letzter Zugriff am 20.12.2023. Online verfügbar: <https://karrierebibel.de/ziele-setzen/>
- [5] P. Steubel, „Was ist ein MVP?” 2023. Online verfügbar: <https://asana.com/de/resources/minimum-viable-product>
- [6] www.creditreform.de, „SO FUNKTIONIERT DIE MARKTANALYSE:METHODEN UND BEISPIELE,” 2024. Online verfügbar: <https://www.creditreform.de/aktuelles-wissen/praxisratgeber/marktanalyse>
- [7] B. Penin, „App Marketing,” 2023. Online verfügbar: <https://www.itportal24.de/ratgeber/app-marketing-wie-vermarkte-ich-meine-app#App-Marketing-Kanaele>
- [8] absatzwirtschaft.de, „Apps deinstallieren,” 2018. Online verfügbar: <https://www.absatzwirtschaft.de/top-studie-top-oder-flop-app-nutzerentscheiden-innerhalb-von-sechs-tagen-ob-sie-eine-app-deinstallieren-221132/>
- [9] OpenAI. (2024) GPT-3.5-based Chatbot. Answer provided by the GPT-3.5-based language model. Online verfügbar: <https://www.openai.com/gpt-3>
- [10] C. Wannakhaos, „Strapi vs. WordPress,” 2022. Online verfügbar: <https://www.trienpont.com/strapi-vs-wordpress-which-one-should-you-use-for-your-next-cms-project/#:~:text=Strapi%20is%20again%20the%20winner,that%20need%20to%20be%20upd>
- [11] firebase.google.com, „Firebase-App-Verteilung,” 2023, letzter Zugriff am 20.12.2023. Online verfügbar: <https://firebase.google.com/docs/app-distribution?hl=de>
- [12] L. Hahn, „Cross-Platform App – Plattformübergreifende Entwicklung mit Flutter, React Native & Co.” 2023. Online verfügbar: <https://www.itportal24.de/ratgeber/cross-platform-app>
- [13] yeeply.com, „Native vs. hybrid,” 2023, letzter Zugriff am 10.11.2023. Online verfügbar: <https://www.yeeply.com/de/blog/was-sind-native-web-und-hybride-apps/>

- [14] N. Mansour, „React Native vs Kotlin Multiplatform: The 2023 Guide,” 2023. Online verfügbar: <https://www.instabug.com/blog/react-native-vs-kotlin-multiplatform-guide>
- [15] L. Hahn, „Flutter vs. React Native,” 2023. Online verfügbar: <https://www.itportal24.de/ratgeber/flutter-vs-react-native#React-vs-Flutter>
- [16] L. von Arcitech, „Xamarin vs. React Native,” 2023. Online verfügbar: <https://www.linkedin.com/pulse/xamarin-vs-react-native-comprehensive-comparison-cross-platform/>
- [17] F. Churchville, „Content-Management-System (CMS),” 2021. Online verfügbar: <https://www.computerweekly.com/de/definition/Content-Management-System-CMS>
- [18] ionos, „Headless CMS,” 2022. Online verfügbar: <https://www.ionos.at/digitalguide/hosting/cms/headless-cms-was-sind-die-vorteile/>
- [19] S. Springer, „Zentrales State Management in React ,” 2022. Online verfügbar: <https://entwickler.de/react/zentrales-state-management-react>
- [20] react.dev, „useState,” 2024, letzter Zugriff am 23.2.2024. Online verfügbar: <https://react.dev/reference/react/useState>
- [21] M. Gold, „Using useContext in React: a comprehensive guide,” 2023. Online verfügbar: <https://medium.com/@msgold/using-usecontext-in-react-a-comprehensive-guide-8a9f5271f7a8#:~:text=useContext%20is%20a%20React%20hook,built%20into%20the%20React%20library>.
- [22] kubernetes.io, „Deployments,” 2023, letzter Zugriff am 19.12.2023. Online verfügbar: <https://kubernetes.io/docs/concepts/workloads/controllers/deployment/>
- [23] —, „Pods,” 2023, letzter Zugriff am 19.12.2023. Online verfügbar: <https://kubernetes.io/de/docs/concepts/workloads/pods/>
- [24] —, „Service,” 2023, letzter Zugriff am 19.12.2023. Online verfügbar: <https://kubernetes.io/docs/concepts/services-networking/service/>
- [25] C. Pinchefske, „Reverse-Proxy-Server erläutert: Was sie sind und wie man sie benutzt,” 2020. Online verfügbar: <https://www.avast.com/>
- [26] StackOverflow, „fetching data from localhost on android,” 2016. Online verfügbar: <https://stackoverflow.com/questions/33704130/react-native-android-fetch-failing-on-connection-to-local-api>
- [27] facebook/react-native github mhrpatel12, „couldn't find DSO to load: libjscexecutor.so caused by: dlopen failed: library "libjsc.so" not found,” 2019. Online verfügbar: <https://github.com/facebook/react-native/issues/25537>
- [28] npmjs.com depcheck, „Wie funktioniert depcheck,” 2023. Online verfügbar: <https://www.npmjs.com/package/depcheck>
- [29] S. Saleem, „npm install --legacy-peer-deps vs --force,” 2023. Online verfügbar: <https://www.linkedin.com/pulse/npm-install-legacy-peer-deps-vs-force-shaharyar-saleem/>
- [30] react.dev, „Suspenses,” 2023. Online verfügbar: <https://react.dev/reference/react/Suspense>

- [31] ——, „Lazy Loading,” 2023. Online verfügbar: <https://react.dev/reference/react/lazy>
- [32] StackOverflow, „useEffect not called in React Native when back to screen,” 2020. Online verfügbar: <https://stackoverflow.com/questions/60182942/useeffect-not-called-in-react-native-when-back-to-screen>
- [33] react.dev, „Hooks,” 2023. Online verfügbar: <https://react.dev/reference/react/hooks>
- [34] ——, „useEffect hook,” 2023. Online verfügbar: <https://react.dev/reference/react/useEffect>
- [35] ——, „Synchronizing with Effects,” 2023. Online verfügbar: <https://react.dev/learn/synchronizing-with-effects>
- [36] ——, „useCallback,” 2023. Online verfügbar: <https://react.dev/reference/react/useCallback>
- [37] reactnavigation.org, „useFocusEffect,” 2023. Online verfügbar: <https://reactnavigation.org/docs/use-focus-effect/>

# Abbildungsverzeichnis

1	Logo MACOLUTION . . . . .	1
2	Logo solvistas . . . . .	1
3	Musik dient zur Entspannung . . . . .	2
4	. . . . .	3
5	Use-Case-Diagramm . . . . .	8
6	Ablauf . . . . .	10
7	. . . . .	11
8	App Statistik [8] . . . . .	14
9	Systemarchitektur . . . . .	19
10	Mockups . . . . .	25
11	Home-Screen . . . . .	27
12	Kategorien . . . . .	28
13	Suchleiste . . . . .	29
14	Eingepflegtes Video aus dem Backend . . . . .	30
15	Kein Favorit gesetzt . . . . .	31
16	Ein Favorit gesetzt . . . . .	31
17	Einstellungen . . . . .	32
18	Dark Home-Screen . . . . .	33
19	Dark Kategorie-Screen . . . . .	33
20	Dark Media-Screen . . . . .	33
21	Dark Settings-Screen . . . . .	33
22	Logo white . . . . .	34
23	Logo black . . . . .	34
24	ERD . . . . .	35
25	Screenshot aus dem UI Prototyp von Relaxoon . . . . .	36
26	Beispiel für den Interactive Query Builder . . . . .	37
27	create android config . . . . .	48
28	Android Config . . . . .	48
29	App Distribution . . . . .	49
30	App Upload . . . . .	49
31	Releases . . . . .	50
32	Plakat . . . . .	55

# **Tabellenverzeichnis**

# Quellcodeverzeichnis

1	protected screens . . . . .	39
2	Validierungsschemen . . . . .	41
3	K8S PVC . . . . .	43
4	K8S PV . . . . .	43
5	Strapi Dockerfile . . . . .	43
6	Secrets für Strapi . . . . .	45
7	Secrets für die Datenbank . . . . .	45
8	create k8s deployments . . . . .	45
9	create a service component . . . . .	46
10	generate android and ios . . . . .	46
11	generate apk . . . . .	47