

# Jahresprojekt 2

Entwicklung einer Umfrageapp



Gruppe Querify

# Inhaltsverzeichnis

<b>1 RAHMENBEDINGUNGEN .....</b>	<b>2</b>
<b>2 EPICS UND USER-STORIES .....</b>	<b>2</b>
2.1 MUSS-ANFORDERUNGEN .....	2
2.2 KANN-ANFORDERUNGEN .....	3
2.3 NICHT-ANFORDERUNGEN .....	4
<b>3 UX-KONZEPT .....</b>	<b>5</b>
3.1 WIREFRAMES .....	5
3.2 INTERAKTIONSDESIGN .....	6
3.3 PROTOTYP .....	7
<b>4 UI-DESIGN .....</b>	<b>7</b>
4.1 NAME DER APP .....	7
4.2 SCHRIFTARTEN UND GRÖßEN .....	8
4.3 FARBSHEMA .....	8
4.4 ICONS UND BUTTONS .....	10
4.5 LOGO UND GRAFIKEN .....	11
4.6 BEISPIELHAFTE LAYOUTS .....	13
<b>5 ARCHITEKTUR .....</b>	<b>13</b>
<b>6 TECHNOLOGIEN .....</b>	<b>15</b>
6.1 FRONTEND .....	15
6.2 BACKEND .....	19
<b>7 LESSONS LEARNED .....</b>	<b>23</b>
7.1 TECHNISCHE HERAUSFORDERUNGEN UND IHRE BEWÄLTIGUNG .....	23
7.2 NÜTZLICHE TOOLS UND BIBLIOTHEKEN .....	23
7.3 BEWÄHRTE PRAKTIKEN .....	24
7.4 WISSENSLÜCKEN UND ENGPÄSSE .....	25
7.5 PERSÖNLICHE LERNERGEBNISSE .....	25
<b>8 NÜTZLICHE LINKS .....</b>	<b>26</b>

# 1 Rahmenbedingungen

1) Technische Anforderungen an den Nutzenden, damit die App funktionsfähig ist:

- Mobiles Gerät mit Internetzugang
- Ausreichend Rechenleistung

2) Zielgruppe:

- Bildungseinrichtungen
- Unternehmen
- Privatpersonen

## 2 Epics und User-Stories

Im Folgenden werden umzusetzende Funktionen der App in Form von User-Stories beschrieben. Das Kapitel ist gegliedert in Anforderungen, welche innerhalb des Projekts umgesetzt werden müssen und Anforderungen, welche innerhalb des Projekts umgesetzt werden können sowie in Anforderungen, die nach technischer Prüfung nicht innerhalb des Projekts umgesetzt werden.

### 2.1 Muss-Anforderungen

ID	Epic	User-Story
MB01	Benutzerverwaltung	Als Nutzer:in möchte ich mich mit E-Mailadresse registrieren können
MB02	Benutzerverwaltung	Als Nutzer:in möchte ich mich mit registrierter E-Mailadresse und Passwort anmelden können
MB03	Benutzerverwaltung	Als Nutzer:in möchte ich mein Profil bearbeiten können, um meine persönlichen Informationen aktualisieren zu können.
MB04	Benutzerverwaltung	Als Nutzer:in möchte ich mein Passwort ändern können, um meine Sicherheit zu erhöhen.
MB05	Benutzerverwaltung	Als Nutzer:in möchte ich meine E-Mail-Adresse ändern können, um meine Kontaktinformationen zu aktualisieren.
MB06	Benutzerverwaltung	Als Nutzer:in möchte ich eine Umfrage erstellen können, um Feedback von anderen Nutzer:innen zu einem bestimmten Thema zu erhalten.

MU01	Umfrage	Typ 1 Multiple Choice: Als Nutzer:in möchte ich verschiedene Fragetypen auswählen können, um meine Umfrage abwechslungsreicher zu gestalten.
MU02	Umfrage	Typ 2 Freitext: Als Nutzer:in möchte ich verschiedene Fragetypen auswählen können, um meine Umfrage abwechslungsreicher zu gestalten.
MU03	Umfrage	Typ 3 Single Choice: Als Nutzer:in möchte ich verschiedene Fragetypen auswählen können, um meine Umfrage abwechslungsreicher zu gestalten.
MU04	Umfrage	Als Nutzer:in möchte ich meine Umfrage mit anderen Nutzer:innen über einen Zahlen-Code teilen können, um ein größeres Publikum zu erreichen.
MU05	Umfrage	Als Nutzer:in möchte ich meine Umfrage mit anderen Nutzer:innen über einen Link teilen können, um ein größeres Publikum zu erreichen.
MU06	Umfrage	Als Nutzer:in möchte ich Ergebnisse und Statistiken zu meiner Umfrage sehen können, um zu verstehen, wie andere Nutzer:in geantwortet haben.
MU07	Umfrage	Als Nutzer:in möchte ich die Möglichkeit haben, Umfragen anderer Nutzer:in zu beantworten, um Feedback zu geben oder Informationen zu teilen.

## 2.2 Kann-Anforderungen

ID	Epic	User-Story
KU01	Umfrage	Als Nutzer:in möchte ich Umfragen suchen und filtern können, um bestimmte Themen zu finden oder nach bestimmten Kriterien zu sortieren. (Wenn Schlagwörter definiert sind).
KU02	Umfrage	Als Nutzer:in möchte ich Benachrichtigungen erhalten, wenn jemand meine Umfrage beantwortet oder wenn neue Umfragen verfügbar sind, die meinen Interessen entsprechen.
KU03	Umfrage	Als Nutzer:in möchte ich die Möglichkeit haben, anonym auf Umfragen zu antworten, um meine Antworten privat zu halten.
KU04	Umfrage	Als Nutzer:in möchte ich in der Lage sein, eine Umfrage mit einem QR-Code zu verknüpfen, um Teilnehmer leichter zu erreichen.
KU05	Umfrage	Als Nutzer:in möchte ich die Option haben, meine Umfrage öffentlich oder privat zu machen, damit ich

		entscheiden kann, wer daran teilnehmen kann. (Rollenkonzept -> Gruppendefinition)
KU06	Umfrage	Als Nutzer:in möchte ich die Möglichkeit haben, eine Umfrage mit verschiedenen Skalen (z.B. Likert-Skala) zu erstellen, um differenzierte Meinungen zu sammeln.
KU07	Umfrage	Als Nutzer:in möchte ich die Möglichkeit haben, eine Umfrage mit einem Timer zu erstellen, um Teilnehmer auf eine bestimmte Zeit zu beschränken.
KU08	Umfrage	Als Nutzer:in möchte ich die Fähigkeit haben, die Umfrage anonym oder mit Teilnehmer-Identifikation zu erstellen, um die Privatsphäre der Teilnehmer zu schützen oder eine Wiedererkennung zu ermöglichen.
KU09	Umfrage	Als Nutzer:in möchte ich die Möglichkeit haben, bei der Planung einer Veranstaltung eine Umfrage zu erstellen, die die aktuellen Wetterbedingungen berücksichtigt.
KU10	Umfrage	Als Nutzer:in möchte ich die Möglichkeit haben, eine Umfrage zu erstellen, die über einen scrollbaren Screen verfügt, um die Nutzer:inerfahrung zu verbessern.
KU11	Umfrage	Als Nutzer:in möchte ich meine Umfrage bearbeiten oder löschen können, wenn ich Änderungen vornehme oder sie nicht mehr benötige. (Ablaufdatum definieren)
KU12	Umfrage	Als Nutzer:in möchte ich die Option haben, meine Umfrage mit einer Kombination aus Ja/Nein-Fragen, Mehrfachauswahl-Fragen und Freitext-Fragen zu erstellen, um unterschiedliche Arten von Feedback zu erhalten.
KU13	Umfrage	Als Nutzer:in möchte ich in der Lage sein, die Ergebnisse meiner Ja/Nein-Fragen in Echtzeit zu sehen, um schnell ein Feedback zu erhalten. (optional: Sekündliche Aktualisierung aktivieren)

## 2.3 Nicht-Anforderungen

ID	Epic	User-Story
NB01	Benutzerverwaltung	Als Nutzer:in möchte ich Umfragen suchen und filtern können, um bestimmte Themen zu finden oder

		nach bestimmten Kriterien zu sortieren. (Wenn Schlagwörter definiert sind).
NU01	Umfrage	Als Nutzer:in möchte ich in der Lage sein, die Umfrageergebnisse herunterzuladen, um sie offline zu speichern oder weiterzuverarbeiten.
NU02	Umfrage	Als Nutzer:in möchte ich die Fähigkeit haben, auf die Freitext-Antworten meiner Umfrage zu reagieren und darauf zu antworten, um eine Diskussion zu ermöglichen.
NU03	Umfrage	Als Nutzer:in möchte ich die Möglichkeit haben, die Umfrage auf verschiedenen Sprachen zu erstellen, um eine internationale Zielgruppe zu erreichen.
NU04	Umfrage	Als Nutzer:in möchte ich die Option haben, eine Umfrage mit Bildern oder Videos zu erstellen, um visuelle Rückmeldungen von Teilnehmern zu erhalten. (Technisch OK/Umsetzung Kritisch -> Bildrechte/DSGVO/Filesserver/Speicher....)
NU05	Umfrage	Als Nutzer:in möchte ich die Option haben, meine Umfrage als Vorlage zu speichern, damit ich sie später wieder verwenden kann.
NU06	Umfrage	Als Nutzer:in möchte ich die Möglichkeit haben, mehrere Administratoren für eine Umfrage zu haben, um eine breitere Gruppe von Nutzer:inn in der Verwaltung zu beteiligen.
NU07	Umfrage	Als Nutzer:in möchte ich die Fähigkeit haben, unterschiedliche Zugriffsberechtigungen für verschiedene Administratoren festzulegen, um die Sicherheit und Integrität der Umfrage zu gewährleisten
NU08	Umfrage	Als Nutzer:in möchte ich meine Umfrage bearbeiten oder löschen können, wenn ich Änderungen vornehme oder sie nicht mehr benötige. (Ablaufdatum definieren)

### 3 UX-Konzept

Dieses Kapitel befasst sich mit der Beschreibung des Designprozesses für die App, einschließlich der Verwendung von Skizzen, Wireframes und Prototypen, um das Design iterativ zu entwickeln und zu testen.

#### 3.1 Wireframes

Die Erstellung von Wireframes dient folgenden Zwecken:

- **Konzeptualisierung und Planung:** Wireframes ermöglichen es dem Design-Team, das grundlegende Layout und die Struktur der Umfrage-App zu planen und konzipieren. Sie dienen als grundlegende Blaupause, um das Design zu skizzieren und die Platzierung von Elementen, Navigationsoptionen und Interaktionsmöglichkeiten zu planen.
- **Visualisierung der Oberfläche:** Wireframes bieten eine visuelle Darstellung der Benutzeroberfläche der Umfrage-App, ohne sich auf Design-Details wie Farben und Schriftarten zu konzentrieren. Sie ermöglichen es allen Beteiligten, einen Überblick über das Layout, die Anordnung der Elemente und die allgemeine Benutzererfahrung zu erhalten.
- **Iterative Verbesserung:** Sie ermöglichen es dem Design-Team, verschiedene Iterationen des Designs schnell zu erstellen und überarbeiten. Durch die Visualisierung der Oberfläche können Schwachstellen oder Probleme in der Navigation behoben werden, bevor das eigentliche Design beginnt. In diesem Abschnitt wird beschrieben, wie das Design der App durch Skizzen, Wireframes und Prototypen iterativ entwickelt und getestet wird, um sicherzustellen, dass die Benutzerbedürfnisse erfüllt werden.

## 3.2 Interaktionsdesign

Im Folgenden wird ein Überblick über die Hauptaktionen vermittelt, die ein Nutzer in „Querify“ ausführen kann. Es werden die Schritte und die Reihenfolge der Interaktionen nach der Anmeldung beschrieben, um sicherzustellen, dass der Benutzer die App effektiv nutzen kann, um Umfragen zu erstellen, zu beantworten und Ergebnisse anzuzeigen.

- **Umfrage erstellen:**
  - Der Benutzer klickt nach erfolgreicher Anmeldung auf den Button „Loslegen“ und gelangt zur Umfrage-Erstellungsseite
  - Auf dieser Seite kann der Benutzer zwischen verschiedenen Umfragearten wählen
  - Nach Auswahl einer Umfrageart mit einem Klick auf das entsprechende Bild kann dieser je nach Umfrageart Umfragetitel, die Fragen, Antwortoptionen und Stichtag festlegen.
  - Nachdem der Benutzer alle erforderlichen Informationen eingegeben hat, klickt er auf „Speichern“
  - Im Anschluss hat der Benutzer die Option die Möglichkeit Einstellungen zur Identifikation, Ergebniseinsicht und Wiederverwendung zu treffen.
  - Mit Klick auf „Speichern“ validiert die App die Eingaben und speichert die Umfrage in der Datenbank.
  - Der Benutzer wird zur Bestätigungsseite der erfolgreichen Umfrageerstellung weitergeleitet und hat die Übersicht zu URL, Eingabecode und QR-Code zur Umfrage.
- **Umfrage beantworten:**
  - Der Benutzer wählt in der unteren Navigationsleiste das Icon „Inbox“ aus

- Die App zeigt eine Übersicht über erstellte sowie mögliche Umfragen zum Teilnehmen an.
- Der Benutzer wählt die gewünschte Umfrage zum Beantworten aus.
- Nach Klick auf Abstimmen gelangt der Benutzer zur Beantwortungsseite
- Der Benutzer wählt die gewünschte Antwortoptionen für jede Frage aus.
- Nachdem der Benutzer alle Fragen beantwortet hat, klickt er auf „Abstimmen“.
- Die App validiert die Antworten und speichert die Ergebnisse in der Datenbank.
- Der Benutzer wird zur Übersichtsseite aller Umfragen zurückgeleitet.
- Ergebnisse anzeigen:
  - Der Benutzer wählt eine abgeschlossene Umfrage aus der Übersichtsseite aller Umfragen aus.
  - Die App zeigt die gesammelten Ergebnisse der Umfrage an.
  - Je nach Art der Umfrage werden die Ergebnisse in Form von Skalen, Werten präsentiert.

### 3.3 Prototyp

Nach Klärung der Notwendigkeit von Wireframes und des Interaktionsdesigns wurden diese angefertigt und unter folgendem Figma als Prototyp abgelegt: [https://www.figma.com/file/UnVjFwQspz5lWqEvixfi6M/Wish-\(Copy\)?type=design&node-id=0-1&t=cjZLrCI2FIkHylvD-0](https://www.figma.com/file/UnVjFwQspz5lWqEvixfi6M/Wish-(Copy)?type=design&node-id=0-1&t=cjZLrCI2FIkHylvD-0)

## 4 UI-Design

In diesem Abschnitt ist die Beschreibung des visuellen Erscheinungsbilds der App, einschließlich des Farbschemas, der Schriftarten und der Bildsprache zu finden.

### 4.1 Name der App

Der Name der App lautet „Querify“. „Querify“ ist ein eingängiger und aussagekräftiger Name für eine Umfrage-App, welcher das Konzept und die Funktion der Anwendung widerspiegelt. Der Name kombiniert das Wort „quer“ mit dem Suffix „-ify“, welches eine Verbindung zu dem englischen Begriff „Query“ (deutsch: Abfrage) herstellt. Durch die Kombination dieser Elemente entsteht ein starker und einprägsamer Begriff, der Aufmerksamkeit und Neugierde weckt. Zudem ist „Querify“ leicht auszusprechen und zu merken, welches seine Wiedererkennung und Verbreitung begünstigt. Der Name hat zudem einen modernen und dynamischen Klang, der die technologische Natur der App unterstreicht. Darüber hinaus ist „Querify“ flexibel und anpassungsfähig, da er verschiedene Bedeutungen und Interpretationen zulässt. Er kann sowohl für professionelle als auch für persönliche Umfragen verwendet werden und spricht ein breites Spektrum von Benutzern an. Insgesamt hat der Name das Potenzial eine starke Markenidentität zu schaffen und das Interesse der Zielgruppe zu wecken.



## 4.2 Schriftarten und Größen

Unter Berücksichtigung von Erscheinungsbild, Lesbarkeit und Modernität wurden verschiedene Schriftarten und -größen für das Logo, die Überschriften und den regulären Text innerhalb der Umfrage-App eingesetzt.

- Logo
  - Font: Italiana
  - Weight: 400
  - Size: 96px
  - Line height: 77px
  - Align: Center
- Überschriften
  - Font: Inter
  - Weight: 700
  - Size: 18px
  - Line height: 18px
- Standard-Text
  - Font: Manrope
  - Weight: 400
  - Size: 14px
  - Line height: 19.12px

## 4.3 Farbschema


Das Farbschema Violett, Rot und Blau wurde für die Umfrage-App ausgewählt, um eine bestimmte Stimmung und eine gezielte visuelle Wirkung zu erzeugen. Jede Farbe hat eine spezifische Bedeutung und trägt zur Gesamtwahrnehmung der App bei.

- Violett: Violett ist eine Farbe, die oft mit Kreativität und Weisheit assoziiert wird. Durch die Verwendung von Violett in der Umfrage-App wird eine Atmosphäre von Originalität und Innovation erzeugt. Es vermittelt den Benutzern ein Gefühl von Intellekt und Fantasie, was in einer Umgebung, in der Gedanken und Meinungen ausgetauscht werden, von Bedeutung ist.
- Rot: Rot ist eine leidenschaftliche und energetische Farbe, die Aufmerksamkeit erregt und Emotionen hervorruft. Durch die Integration von Rot in das Farbschema wird die App visuell ansprechend und anregend. Rot symbolisiert auch Entschlossenheit und Handlungsfreude, was die Benutzer ermutigt, aktiv an Umfragen teilzunehmen und ihre Meinungen zu äußern.
- Blau: Blau ist eine Farbe, die oft mit Ruhe und Stabilität in Verbindung gebracht wird. Durch die Integration von Blau in das Farbschema wird eine beruhigende und vertrauenswürdige Atmosphäre geschaffen, die den Benutzer ein Gefühl von Sicherheit vermittelt. Darüber hinaus kann Blau auch mit Verlässlichkeit und Seriosität assoziiert werden, was wichtig ist, um das Vertrauen der Benutzer in die Umfrage-App zu stärken.

Die Wahl der Hintergrundfarbe für die Umfrage-App-Screens wurde auf Weiß getroffen, um eine saubere, minimalistische und professionelle Atmosphäre zu schaffen. Weiß eignet sich insbesondere für eine gute Lesbarkeit, die Fokussierung auf textliche Inhalte und die Assoziation mit Reinheit und Objektivität. Lediglich der Homescreen beim Öffnen der App hat als Hintergrundfarbe Schwarz, da hier eine gewisse Eleganz und seriöse Ästhetik erzeugt werden soll.

Die einzelnen Farbcodes unserer Farbpalette sind aus folgender Tabelle zu entnehmen:

Farbliche Darstellung	Farbcode
	#ac2f67
	#902d6e
	#824bc3
	#422cb5
	#4d4c92
	#4e73e6
	#426fa8
	#318cb9
	#da3748
	#770f18
	#f7f7f7




















	#090a0a
-----------------------------------------------------------------------------------	---------

## 4.4 Icons und Buttons

Für die App wurden schlichte Icons und Buttons verwendet, da diese eine klare und einfache Form haben und leicht zu erkennen und zu verstehen sind. Durch die Verwendung derartiger Komponenten wird sichergestellt, dass Benutzer schnell und intuitiv die Bedeutung erfassen können. Dies verbessert die Benutzererfahrung und erleichtert die Navigation durch die App. Außerdem sorgen diese für eine reduzierte visuelle Ablenkung.

Die Icon und Buttons der Umfrage-App werden im Folgenden mit ihrer Bedeutung aufgelistet:

Icon und Button	Bedeutung
	Anmeldebutton, Icon symbolisiert Eingang
	Button zum Erstellen eines Kontos, Icon symbolisiert Konto-Profil
	Button für nächsten App-Screen, Pfeil-Icon symbolisiert nächsten Schritt
	Icon symbolisiert Kalender, Dient zur Auswahl des Geburtstages
	Button zum Anlegen eines Kontos
	Icon symbolisiert erfolgreiche Konto-Anlage
	Icon symbolisiert Konto
	Icon symbolisiert Inbox
	Icon symbolisiert Kalender
	Icon symbolisiert Statistik
	Button zum Erstellen einer neuen Umfrage

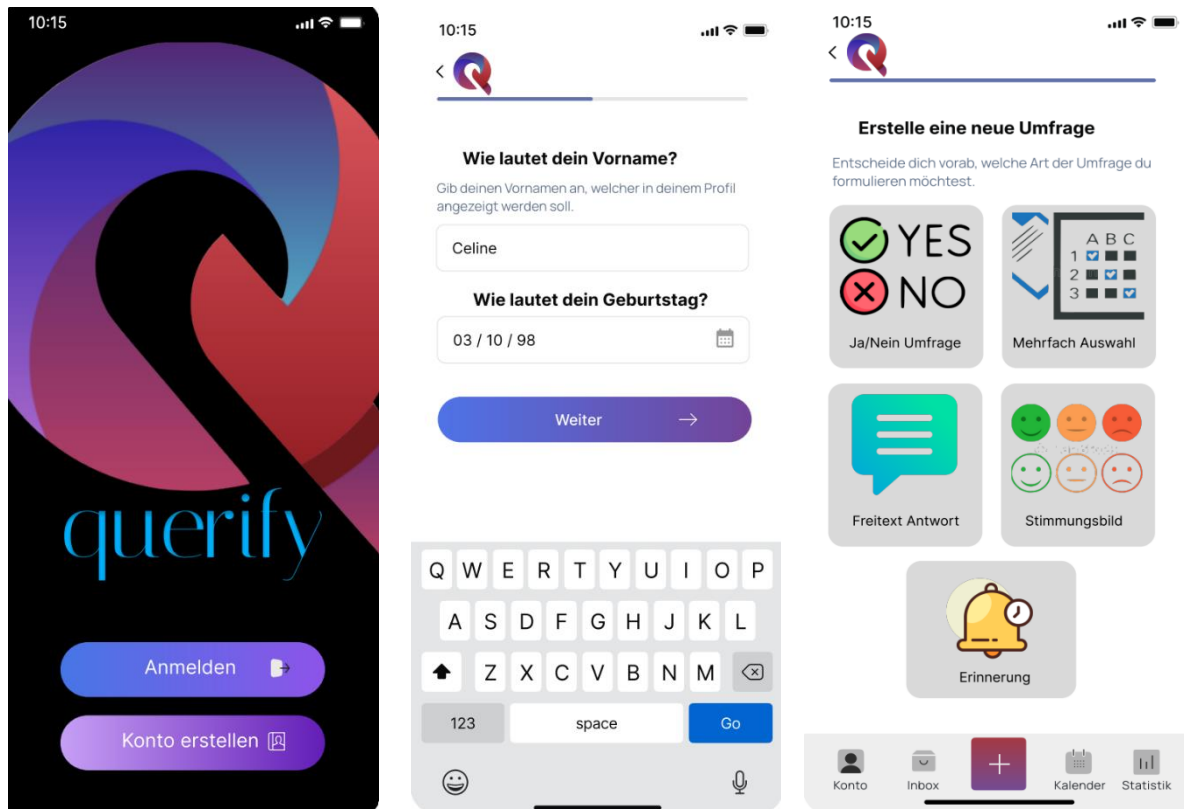
	Icon symbolisiert Auswahl des Zeitpunktes für einen Stichtag
	Button zum Kopieren einer URL, eines Eingabe-Codes oder eines QR-Codes
	Icon symbolisiert Profil
	Icon symbolisiert Passwort
	Icon symbolisiert Benachrichtungen
	Icon symbolisiert Bewertung & Feedback
	Icon symbolisiert Hilfe-Seite
	Icon symbolisiert Kontakt-Seite
 Abmelden	Button zum Abmelden aus dem Benutzerkonto
	Icon symbolisiert Umfrageersteller/-in
	Icon symbolisiert Abstimmfenster
	Icon symbolisiert Bearbeitungsmöglichkeit einer Umfrage
	Icon symbolisiert die Möglichkeit alle Stimmen einer Umfrage zu löschen
	Icon symbolisiert Möglichkeit eine Umfrage zu löschen
	Icon symbolisiert QR-Code-Anzeige
	Icon symbolisiert Umfrage-URL
	Icon symbolisiert Exportierungsvorgang
	Icon symbolisiert Teilnehmende
	Icon symbolisiert Stichtag

## 4.5 Logo und Grafiken



Das Logo für die App wurde entwickelt, um den Zweck der Anwendung visuell zu repräsentieren. Es wurde eine markante und moderne Gestaltung gewählt, um die Aufmerksamkeit der Benutzer zu erregen und eine starke visuelle Identität für die App zu schaffen. Das Logo besteht aus dem einprägsamen Symbol/Buchstaben „Q“, welcher den Anfangsbuchstaben des App-Namens widerspiegelt. Das Logo ist von seiner Form als auch in seiner Farbgebung einzigartig und hebt sich dadurch von anderen App-Logos ab. Die Farbpalette des Logos wurde sorgfältig ausgewählt, um sowohl Aufmerksamkeit zu erregen als auch eine positive Benutzererfahrung zu gewährleisten. Die detaillierte Begründung für das gewählte Farbschema ist Kapitel 4.3 zu entnehmen. Die Typografie des Logos ist gut lesbar. Der Name der App wird hierbei in einer klaren Schriftart dargestellt, die zum Charakter der Anwendung passt. Das Logo wurde mit Blick auf die Verwendung auf verschiedenen Bildschirmauflösungen optimiert. Es wurde darauf geachtet, dass es auch in kleineren Größen gut erkennbar ist und seine Wirkung beibehält, da die App hauptsächlich für Mobiltelefone ausgerichtet ist. Insgesamt schafft das Logo eine starke visuelle Identität und unterstützt die Benutzer bei der schnellen Wiedererkennung der App.

## 4.6 Beispielhafte Layouts



## 5 Architektur

Unsere Anwendung Querify zeichnet sich durch eine intuitive Benutzeroberfläche und leistungsstarke Funktionen aus, die eine einfache und effiziente Umfragenutzung ermöglichen. In diesem Kapitel wird die Architektur von Querify näher betrachtet und die Zusammenarbeit seiner Komponenten erläutert. Fokus liegt hierbei auf die Schlüsselaspekte der App, die seine Leistung und Benutzerfreundlichkeit auszeichnen.

Unsere App basiert auf etablierten Technologien, darunter React Native für das Frontend, Node.js für das Backend und MariaDB als relationales Datenbankmanagementsystem. Im Rahmen dieses Kapitels soll die Vorstellung der Architektur und seinen Funktionen erfolgen. Die Begründung der Auswahl erfolgt im nachfolgenden Kapitel.

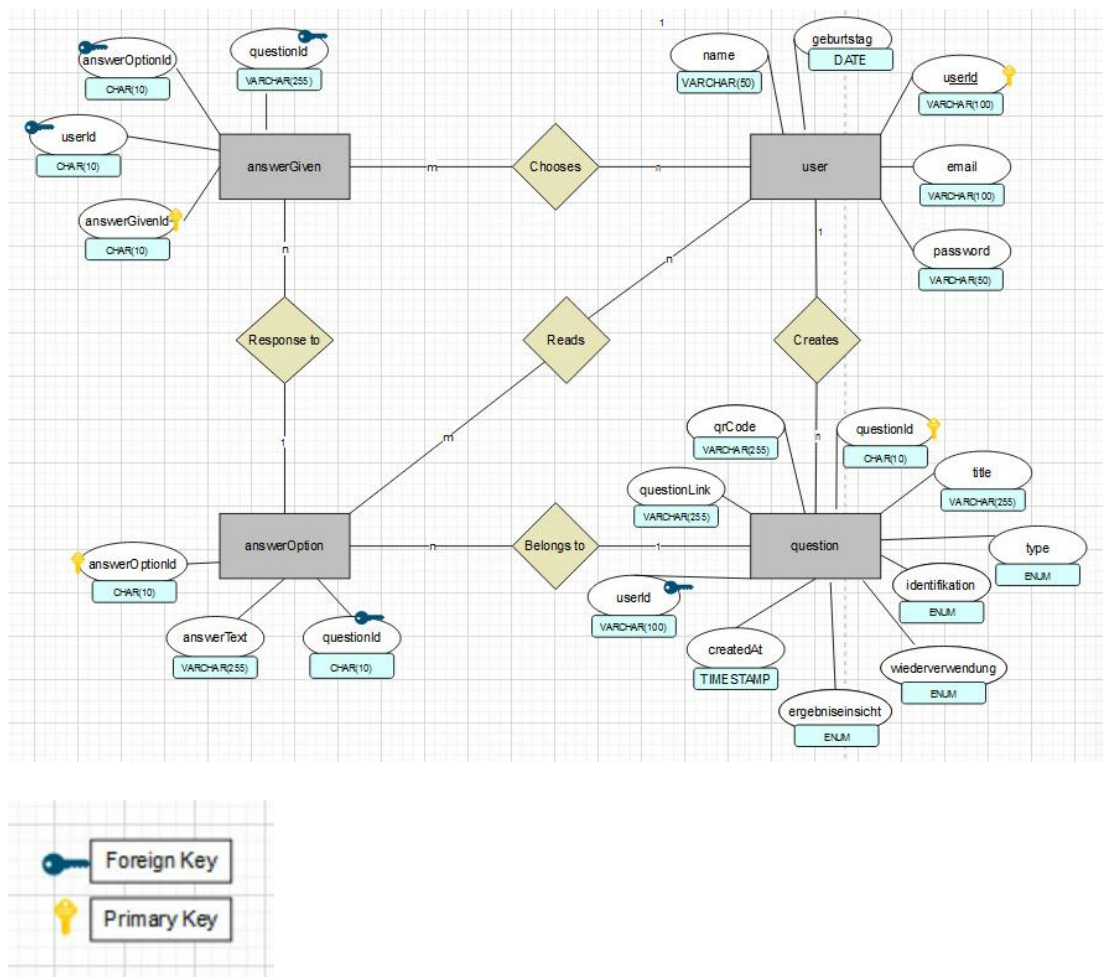
Die Komponenten sind wie folgt:

1. Querify-App (System):
  - Diese zentrale Einheit stellt unsere Umfrage-App dar.
  - Sie kommuniziert sowohl mit dem Frontend als auch mit dem Backend, um Benutzerinteraktionen zu verarbeiten und Daten zu verwalten.
2. React Native Frontend:

- Das React Native Frontend ist die Benutzerschnittstelle, die auf mobilen Geräten wie Smartphones läuft.
- Es ermöglicht Benutzern die Interaktion mit der Umfrage-App und sendet Anfragen an das Backend.
- Diese Anfragen beziehen sich auf die Frontendkomponenten BottomNavigation.js, ColorPalette.js, DateFormatter.js, DataInput.js, DescriptionInput.js, LogoTitle.js, SingleLineInput.js, StatButton.js, StatButtonOwn.js, StatButtonOwnOld.js, SubmitButton.js und TitleInput.js.

### 3. Node.js Backend:

- Das Backend ist verantwortlich für die Verarbeitung von Anfragen, die von React Native gesendet werden.
- Es stellt RESTful APIs bereit und interagiert mit der Datenbank, um Daten abzurufen und zu speichern.
- Sie verfügt über die Funktion generateID und die Komponenten: answerGivenRouter.js, answerOptionRouter.js, mixRoute.js, questionRouter.js und userRouter.js.
- Das Entity Relationship (ER)-Diagramm ist der folgenden Grafik zu entnehmen:



### 4. Datenbank (MariaDB):

- Die Datenbank speichert Umfragedaten und Benutzerinformationen.

- Das Backend kommuniziert mit der MariaDB Datenbank, um Daten zu speichern und abzurufen.

Die Komponenten bilden das Gerüst, auf dem unsere App aufgebaut ist, während die Funktionen den eigentlichen Mehrwert für unsere Nutzer liefern. Die wesentlichen Funktionen lassen sich hierbei in Umfrageverwaltung, Benutzerverwaltung und Push-Benachrichtigungen zuordnen.

1. Umfrageverwaltung: Hier werden Umfragen erstellt, aktualisiert und gelöscht. Die Verarbeitung von Umfrageergebnissen kann ebenfalls stattfinden.
2. Benutzerverwaltung: Benutzerverwaltungsfunktionen wie Registrierung, Anmeldung und Aktualisierung von Benutzerprofilen.
3. Push-Benachrichtigungen: Diese Komponente sendet Push-Benachrichtigungen an das Frontend, um Benutzer über z.B. neu zugewiesene Umfragen zu informieren.

## 6 Technologien

Die Auswahl der richtigen Technologien ist sehr wichtig, wenn wir eine App planen und erstellen. Die Technologien, die wir auswählen, sind wie das Grundgerüst, auf dem das ganze Projekt basiert. Sie haben einen großen Einfluss darauf, wie gut die entwickelte App funktioniert, wie leicht sie erweitert werden kann und wie einfach sie für die Benutzer ist. In diesem Kapitel wird genau erläutert, wie wir die besten Technologien für den Teil der Software wählen, den die Benutzer sehen und für den Teil, den sie nicht sehen, aber der im Hintergrund arbeitet.

### 6.1 Frontend

Die Auswahl der Frontend-Technologie React Native für die Entwicklung unserer Querify-App zusammen mit den entsprechenden Bibliotheken und JavaScript, kann auf verschiedenen Ebenen begründet werden. Im Folgenden werden die Gründe für die Auswahl von React Native als Frontend-Framework, die damit verbundenen Bibliotheken und die Verwendung von JavaScript detailliert erläutert. Die folgende Begründung führt zur Architekturentscheidung, die die Grundlage für die App-Entwicklung darstellt.

#### **React Native als Frontend-Framework:**

Die Wahl von React Native als Frontend-Framework für unsere App basiert auf mehreren entscheidenden Überlegungen, die in die Architektur der Anwendung einfließen. Zunächst einmal ermöglicht React Native die Entwicklung von mobilen Anwendungen für sowohl iOS als auch Android unter Verwendung einer gemeinsamen Codebasis. Dieser plattformübergreifende Ansatz reduziert den Entwicklungsaufwand erheblich und führt zu konsistenten Ergebnissen auf verschiedenen Plattformen. Eine einheitliche Codebasis bildet die Grundlage für eine effiziente und gut wartbare Architektur.



Darüber hinaus bietet React Native eine herausragende Leistung und eine nativ wirkende Benutzeroberfläche, da es nativen UI-Komponenten verwendet. Die Architekturentscheidung für React Native gewährleistet somit eine reibungslose Benutzererfahrung und schnelle Reaktionszeiten, was in einer Umfrage-App von entscheidender Bedeutung ist.

Die Architekturwahl von React Native wird weiter gestärkt durch die Verfügbarkeit einer aktiven Entwickler-Community, die kontinuierlich neue Funktionen und Bibliotheken entwickelt. Diese Ressourcen tragen dazu bei, die Architektur der Umfrage-App durch bewährte Lösungen und bewährte Methoden zu unterstützen.

### **Bibliotheken für React Native:**

Die Integration von bestimmten Bibliotheken in die Architektur der Umfrage-App stellt sicher, dass die Anforderungen der Benutzer effektiv erfüllt werden. Einige der wichtigsten Bibliotheken und ihre Begründungen im Kontext der Architektur sind:

1. **Axios:** Die Einbindung von Axios in die Architektur ermöglicht eine zuverlässige Kommunikation zwischen der Umfrage-App und dem Backend-Server, was einen zentralen Aspekt der Architektur darstellt.
2. **'expo-linear-gradient':** Dies ist eine Bibliothek, die die Erstellung von linearen Verläufen in React Native ermöglicht.
3. **MaterialCommunityIcons:** Eine Bibliothek zur Verwendung von Design-Icons innerhalb der React Native-Anwendung.
4. **'@expo-google-fonts/manrope':** Dies ist eine Bibliothek, die die Verwendung von Google Fonts ermöglicht. In unserer Anwendung wird für Standardtext die Schriftart Manrope verwendet.

### **JavaScript als Programmiersprache:**

Die Entscheidung, JavaScript als die primäre Programmiersprache für die Architektur der Umfrage-App zu verwenden, wurde aufgrund einer Reihe von Überlegungen getroffen, die die Architektur auf verschiedene Weisen positiv beeinflussen:

- **Schnelle Entwicklung:** Die Verwendung von JavaScript ermöglicht die zügige Entwicklung von Prototypen und Anwendungen, was die Agilität und Anpassungsfähigkeit der Architektur unterstreicht. Die Architektur kann somit schnell auf geänderte Anforderungen reagieren.
- **Große Entwicklergemeinschaft:** Die breite Unterstützung und Verfügbarkeit von Ressourcen in der JavaScript-Entwicklergemeinschaft tragen dazu bei, dass die Architektur auf bewährten Lösungen und bewährten Praktiken basiert.
- **Plattformunabhängigkeit:** Die Verwendung von JavaScript in der Architektur eröffnet die Möglichkeit, die Umfrage-App auf verschiedenen Plattformen bereitzustellen und Webtechnologien in die Architektur zu integrieren.

## Architekturentscheidung Frontend:

Insgesamt führen diese begründeten Entscheidungen zur Auswahl von React Native als Frontend-Framework, der Integration spezifischer Bibliotheken und der Verwendung von JavaScript in der Architektur der Umfrage-App zu einer zentralen Architekturentscheidung. Diese Architekturwahl bildet das solide Fundament, auf dem die gesamte Anwendung aufbaut. Sie ermöglicht eine plattformübergreifende, leistungsstarke und benutzerfreundliche Umfrage-App, die den Anforderungen der Benutzer entspricht und zukünftige Erweiterungen und Wartungen erleichtert. Die Architekturentscheidung stellt sicher, dass die Umfrage-App nicht nur technisch funktioniert, sondern auch skalierbar und gut wartbar ist, was letztendlich zum Erfolg des Projekts beiträgt.

## Interviewtranskript: Frontend

Interviewer	Welche Technologien und Frameworks werden im Frontend verwendet? Gibt es bestimmte Gründe für die Auswahl von Frameworks oder Bibliotheken
Interviewpartner	<i>Hauptsächlich React Native. Es ist ein Framework mit dem Apps programmiert werden können. Der Vorteil daran ist, dass man es in JavaScript programmiert. Man kann sich dadurch eine native App generieren lassen. Bei iOS heißt es Objective C was nachher herauskommt und bei Java heißt es glaube ich Kotlin, d.h. man programmiert es quasi einmal und hat eine native App für beide Plattformen. Der Vorteil ist, dass sie eben leistungstechnisch besser sind als Apps, die man in HTML, CSS usw. schreibt und die dann einfach ablaufen lässt. Außerdem gibt es in React Native natürlich Bibliotheken und Packages, wo wir auch ein paar verwenden. Zum Beispiel verwenden wir den integrierten Kalender, d.h. es ist ein vorgefertigter Kalender, wo man dann noch selbst das Design und Schriftarten anpassen kann.</i>
Interviewer	Worauf habt ihr bei der Umsetzung der Benutzeroberfläche besonders wertgelegt?
Interviewpartner	<i>Einerseits das natürlich das Design so gut wie möglich umgesetzt wird und andererseits auf Intuitivität. Ganz banales Beispiel war, dass bei der Erstellung einer Mehrfachumfrage der Knopf zur Erstellung einer neuen Antwortmöglichkeit unter dem Textfeld war. Wir haben es dann neben dem Textfeld gepackt haben, damit es einfach in der Bedienung ist und man so schneller Wählantwortmöglichkeiten hinzufügen kann. Diese 2 Punkte waren unsere Hauptfokusse.</i>
Interviewer	Wie wird sichergestellt, dass die Anwendung auf verschiedenen Geräten und Bildschirmgrößen optimal funktioniert?
	<i>Man kann mit verschiedenen Techniken hier arbeiten. Man kann z.B. die Bildschirmauflösung auslesen lassen und dementsprechend skalieren, sodass z.B. Knöpfe immer 80% von der Bildschirmbreite sind und 20% von der Bildschirmhöhe. So schafft man, dass es auf allen Geräten gleich aussieht, wobei das natürlich nur optimiert ist für Handys. Für Tablets würde man die Benutzeroberfläche eher noch ein bisschen anders</i>

	<i>gestalten. Es sieht natürlich doof aus, wenn da ein Riesenknopf auf dem Tablet ist, der auf dem Handy eine gute Größe hat aber auf dem Tablet ist es dann viel zu groß. Hier kann man den Bildschirmplatz ganz anders nutzen. Unsere App ist gerade für Handys optimiert – da aber für die verschiedenen Bildschirmgrößen, d.h. ob man unsere App auf einem iPhone 11 oder iPhone 11 Pro Max laufen lässt, würde keinen Unterschied darstellen.</i>
Interviewer	Wie erfolgt die Kommunikation zwischen Frontend und Backend? Welche API-Endpunkte werden aufgerufen?
Interviewpartner	<i>Es läuft über eine Rest-API. Ich denke wir machen das über Axios, so heißt die Bibliothek mit der die Calls gemacht werden. Es gibt halt die verschiedenen Calls, die das Backend initiiert, die man braucht um bestimmte Informationen abzurufen oder zu schreiben. Und mit Axios kann man dann die Calls machen, also ganz normal z.B. erstmal getCall.</i>
Interviewer	Werden bestimmte Standards beachtet, um sicherzustellen, dass die Anwendung für alle Benutzer zugänglich ist?
Interviewpartner	<i>Statt Richtung Barrierefreiheit, haben wir eher unsere Standards beim Coden gesetzt – Design Patterns und einheitlicher Flow zum Beispiel. Beim Coden gibt es Design Patterns. Beispielsweise werden Funktionen klein geschrieben oder wenn du z.B. ein zweiteiliges Wort hast, kannst du es ja mit Unterstrich trennen oder mit CamelCase arbeiten und dann haben wir uns halt auf eins geeinigt und das ist dann unser Design Pattern und danach haben wir dann gearbeitet. Wir haben da einige Design Pattern genommen, die es auch schon bereits gab.</i>
Interviewer	Wie werden Frontend-Komponenten und Funktionen getestet, um eine hohe Code-Qualität sicherzustellen?
Interviewpartner	<i>Durch Expo und React Native haben wir die Möglichkeit die App direkt zu testen. Wir sehen die App direkt auf unserem Handy vor uns in einer Container App, Expo Go heißt sie, und sehen alle Elemente die wir coden in Echtzeit auf dem Handy. Man programmiert also was, sieht es auf dem Handy und testet sich dann quasi durch, ob alles so funktioniert wie es soll.</i>
Interviewer	Wie erfolgt die Zusammenarbeit mit dem Backend-Team bei der Definition von API-Anforderungen und -Formaten?
Interviewpartner	<i>Wir haben eine WhatsApp-Gruppe, da kommt dann rein, was wir noch brauchen. Wenn wir halt merken, wir müssen hier eine Frontend Funktion umsetzen und es gibt noch nicht den passenden Call dazu – Beispielsweise brauchen wir irgendwelche Nutzerdaten für die es gerade noch keinen Call gibt, also auf die man vom Frontend aus noch nicht zugreifen kann. In diesem Fall schreiben wir das in der Gruppe, dass wir einen Getcall brauchen mit den und den Parametern und das Backend-Team setzt es dann um.</i>
Interviewer	Habt ihr Vorschläge für zukünftige Erweiterungen oder Verbesserungen des Frontends?
Interviewpartner	<i>Klar, Richtung Barrierefreiheit kann sich noch Gedanken machen – ist immer gut natürlich. Ansonsten haben wir noch in Richtung Animation relativ wenig. Hier haben wir halt Standardanimationen, die aber auch natürlich schon was her machen. Aber diese sind nicht aufwendig zu</i>

	<p><i>implementieren, in dieser Hinsicht könnte man glaube ich noch mehr mit Animationen arbeiten.</i></p> <p><i>Später könnte man natürlich auch noch Spielereien wie verschiedene Farbthemen, dass man also z.B. Dark Mode hat und Light Mode hat, einbauen und mehr in Richtung Personalisierung umsetzen – aber das wärs auch schon mal fürs Erste.</i></p>
--	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## 6.2 Backend

Die Wahl der Backend-Technologie für die Entwicklung von Querify einem von entscheidender Bedeutung, da sie die Leistung, Skalierbarkeit, Sicherheit und die Gesamteffizienz der Anwendung beeinflusst. Im Folgenden werden die Gründe für die Auswahl von Node.js/Express.js und MariaDB als Backend-Technologien näher erläutert.

### 1. Node.js/Express.js als serverseitiges Framework:

Node.js ist eine JavaScript-Laufzeitumgebung, die eine ereignisgesteuerte Architektur bietet mit folgenden Vorteilen:

- *Effizienz und Geschwindigkeit:*  
Node.js ist für seine Leistungsfähigkeit und Geschwindigkeit bekannt. Durch die Verwendung von JavaScript auf sowohl der Client- als auch der Serverseite kann Code zwischen beiden leichter geteilt werden, was die Entwicklung beschleunigt.
- *Skalierbarkeit:*  
Node.js ermöglicht eine einfache horizontale Skalierung durch den Einsatz von Cluster-Modulen und Load-Balancern. Dies ist wichtig, um sicherzustellen, dass die App auch bei steigender Nutzerzahl reibungslos funktioniert.
- *Reichhaltiges Ökosystem:*  
Node.js verfügt über eine umfangreiche Auswahl an Open-Source-Bibliotheken und Modulen, die die Entwicklung beschleunigen und die Integration von Drittanbieterdiensten erleichtern.
- *Einheitliche Programmiersprache:*  
Die Verwendung von JavaScript sowohl auf der Client- als auch auf der Serverseite erleichtert die Kommunikation zwischen Frontend- und Backend-Entwicklern.

### 2. MariaDB als relationales Datenbanksystem:

Gründe für die Auswahl von MariaDB sind wie folgt:

- *Relationale Datenstruktur:*  
MariaDB verwendet das bewährte relationale Datenbankmodell, das gut strukturierte und verknüpfte Daten ermöglicht. Dies ist entscheidend, um Umfrageergebnisse effizient zu speichern und abzurufen.
- *ACID-Konformität:*

MariaDB garantiert die ACID-Eigenschaften (Atomicity, Consistency, Isolation, Durability), die für Transaktionsintegrität und Datenkonsistenz entscheidend sind. Dies ist besonders wichtig, wenn mehrere Benutzer gleichzeitig auf die Datenbank zugreifen.

- *Skalierbarkeit:*  
MariaDB bietet verschiedene Optionen für die Skalierung, darunter Master-Slave-Replikation und Sharding, was es ermöglicht, die Datenbankleistung an die Anforderungen der Anwendung anzupassen.
- *Open Source:*  
MariaDB ist eine Open-Source-Datenbank, was bedeutet, dass sie kostenlos verwendet werden kann und eine aktive Entwicklergemeinschaft hat, die kontinuierlich an ihrer Verbesserung arbeitet.

### 3. Verwendung von Postman für API-Tests:

- *Benutzerfreundliche Oberfläche:*  
Postman bietet eine intuitive Benutzeroberfläche, die es Entwicklern erleichtert, HTTP-Anfragen zu erstellen, zu senden und zu überprüfen. Diese Benutzerfreundlichkeit ermöglicht es unkompliziert API-Tests durchzuführen.
- *Automatisierte Tests:*  
Postman unterstützt die Automatisierung von API-Tests. Durch das Erstellen von Testskripten können wir sicherstellen, dass die Backend-Calls konsistent und zuverlässig funktionieren.
- *Collections:*  
Mit Postman können Collections erstellt werden, die alle erforderlichen API-Endpunkte und Testfälle zusammenfassen. Dies erleichtert die Verwaltung und Ausführung von Testszenarien für verschiedene Teile unserer App.
- *Dokumentation und Zusammenarbeit:*  
Postman ermöglicht die Erstellung von API-Dokumentationen und das Teilen von Testsammlungen mit anderen Teammitgliedern. Dies fördert die Zusammenarbeit und erleichtert die Kommunikation zwischen Backend- und Frontend-Team.
- *Flexibilität:*  
Postman unterstützt verschiedene Authentifizierungsmethoden, Parameterübergaben und Testumgebungen. Dies ermöglicht es uns, verschiedene Testbedingungen und Szenarien zu simulieren, um sicherzustellen, dass Querify in verschiedenen Situationen robust funktioniert.
- *Integration:*  
Postman bietet Integrationen mit anderen Tools wie Continuous Integration (CI)-Systemen und Versionskontrollsystemen. Dadurch können wir API-Tests nahtlos in unseren Entwicklungsworkflow integrieren.

### Architekturentscheidung Backend:

Zusammenfassend lässt sich sagen, dass die Auswahl von Node.js/Express.js als Backend-Technologie und MariaDB als Datenbank im Rahmen unseres Projekts aufgrund ihrer Leistungsfähigkeit, Skalierbarkeit, Sicherheit und des umfangreichen Ökosystems die beste Wahl ist. Diese Technologien bieten die erforderliche Flexibilität, um die App effizient zu entwickeln und sicherzustellen, dass sie den Anforderungen der Benutzer gerecht wird. Die Verwendung von Postman als Werkzeug zum Testen der Backend-Calls ergänzt die Auswahl von Node.js/Express.js und MariaDB als Backend-Technologien

### Interviewtranskript: Backend

Interviewer	Welche Programmiersprachen und Frameworks werden im Backend eingesetzt?
Interviewpartner	<i>Wir nutzen Node.js bzw. die Erweiterung Express.js.</i>
Interviewer	Worauf habt ihr bei der Umsetzung der Benutzeroberfläche besonders wertgelegt?
Interviewpartner 1	<i>Für mich war das so, weil ich das schon kannte. Ich hatte mit Node.js schon bisschen gearbeitet gehabt im Private. Ist ein sehr populäres Framework mit gutem Support und einer guten Community. Ist eine gängige Lösung. Und weil die Sprache so populär ist, haben sie auch immer den neuen Sicherheitsstand.</i>
Interviewer	Welche Art von Datenbanken werden verwendet?
Interviewpartner 1	<i>Wir nutzen MariaDB, welches eine SQL-Datenbank ist.</i>
Interviewpartner 2	<i>Wir haben das veranschaulicht mit DBeaver, da kann man die Datenbank drauf laufen lassen und hat eine visuelle Komponente dazu.</i>
Interviewer	Welche Maßnahmen werden ergriffen, um die Datenbankleistung und -skalierbarkeit zu gewährleisten?
Interviewpartner 2	<i>An sich ist Node.js schon relativ gut skalierbar und leistungsfähig, welches auch ein Grund unserer Wahl von der Sprache ist.</i>
Interviewpartner 1	<i>Wir haben es in die 3.Normalform gebracht, wodurch wir keine Redundanzen in den Daten hat. Dadurch haben wir eine bessere Performance und Skalierbarkeit</i>
Interviewer	Wie ist die API-Struktur aufgebaut?
Interviewpartner 1	<i>Wir haben es so aufgeteilt, wie wir auch die Datenbank aufgeteilt haben, d.h. wir haben einmal den User - zum Erstellen, zum Einloggen usw. Dann haben wir die jeweiligen Umfragen, wo einfach nur die Fragen beschrieben werden, wer sie erstellt, wo es auch eine Beschreibung gibt, was für eine Art von Frage es ist. Darauf kommt dann der 3. Baustein, welches die Antwortmöglichkeiten sind. Diese sind entweder Ja/Nein, Mehrwachaussagen, Freitext. Das sind nur die Optionen, d.h. es sind nicht die Antworten, die gegeben wurden. Das wäre unser 4. Teil, wo tatsächlich gespeichert wird, was die jeweiligen Beantworter auch tatsächlich geantwortet haben. Das sind so die 4 Teile, die sich sowohl in unserer API-Struktur als auch in unserer Datenbankstruktur widerspiegeln.</i>
Interviewer	Welche Technologien oder Ansätze werden verwendet, um die Backend-Performance zu optimieren?

Interviewpartner 2	<i>Wir haben unsere Backend-Calls mit Postman getestet.</i>
Interviewpartner 1	<i>Für das Testen und auch um eine Dokumentation der Calls zu haben, die wir ans Frontend weitergeben können. Damit diese sehen, was für Parameter diese bereitstellen müssen.</i>
Interviewpartner 2	<i>Genau, für das Frontend und auch für die Kommunikation zwischen uns, damit wir immer die aktuelle Version von unseren Calls haben.</i>
Interviewer	Wie werden Fehler verfolgt, protokolliert und behoben?
Interviewpartner 2	<i>Wir arbeiten halt mit Try-Catch-Blöcken und überprüfen dann auch immer die übergebenen Parameter. Wenn da eben was undefined ist oder nicht stimmt, geben wir das vor der Bearbeitung wieder zurück ans Frontend, dass die Anfrage ungültig ist. 500 geben wir zurück, wenn ein unvorhergesehener Server-Error passiert. Der Code 400 ist Bad-Request, welches wir immer am Anfang prüfen. Da soll dann Name und Passwort String sein und wenn dort z.B. eins ein Integer ist, dann geben wir das als Bad-Request zurück.</i>
Interviewer	Gibt es bereits hinsichtlich des Backends bereits eine Dokumentation?
Interviewpartner 1	<i>Wir haben eine ReadMe erstellt, welches die Basis ist, wie man überhaupt zu unserem Backend kommt. Die Installation usw. für das Frontend, damit diese wissen wie man vorgehen müssen. Über Postmann haben wir unsere Postman Collection, also unsere Calls mehr oder weniger dokumentiert.</i>
Interviewpartner 2	<i>Außerdem haben wir unser ER-Modell erstellt. Da haben wir die 4 Bausteine, die wir vorhin erklärt haben, haben wir in einem Diagramm aufgeschrieben.</i>
Interviewpartner 1	<i>Ja, da sind die ganzen Daten mit Fremd- und Primärschlüssel drin.</i>
Interviewer	In welchen Aspekten musstet ihr mit dem Frontend-Team arbeiten und wie lief die Zusammenarbeit?
Interviewpartner 1	<i>Eigentlich mussten wir immer mit dem Frontend zusammenarbeiten, weil wir die Calls, die sie brauchen, aufbauen.</i>
Interviewpartner 2	<i>Wir haben anfangs immer so 1-2mal wöchentlich Meetings mit dem Frontend gehabt. Und auch über WhatsApp lief die Kommunikation auch immer. Da kam die Anforderung vom Frontend, die wir dann umgesetzt haben.</i>
Interviewer	Habt ihr Vorschläge für zukünftige Erweiterungen oder Verbesserungen des Backends?
Interviewpartner 2	Wenn die App irgendwann Richtung App Store wandern sollte, dann passt die Sicherheit vermutlich noch nicht vollständig – Richtung Passwort hashen.
Interviewpartner 1	Ja, eine richtige Authentifizierung gibt es nicht, welches dafür dann notwendig ist. Bis jetzt sind Passwort und User normale Einträge wie auch eine Umfrage, was aber so nicht sein dürfte. Vielleicht auch noch Richtung besseres Error-Handling.
Interviewpartner 2	Wir haben schon geschaut, dass die gängigen Fälle bei Fehlern abgefangen werden. Aber da gibt es bestimmt noch ein paar verzwickte Fälle. Aber die gängigen Fälle sind bereits abgedeckt.

## 7 Lessons Learned

Während des Verlaufs dieses Projekts wurden zahlreiche wertvolle Erkenntnisse gesammelt, die sowohl technischen Fähigkeiten als auch das Verständnis für die Zusammenarbeit in Teams erweitert haben. In den folgenden Abschnitten werden einige der wichtigsten Lehren, die aus Sichten des Projektmanagements, Frontends, Backends und Designs gezogen wurden, näher beschrieben.

### 7.1 Technische Herausforderungen und ihre Bewältigung

- *Projektmanagement*  
Digitale Abstimmungen, das Finden von geeigneten Zeiten für Treffen und die Verwaltung von Zeitzoneverschiebungen stellen hier die technischen Herausforderungen dar. Diese wurden aber durch die Terminplanung und Abwesenheitsplanung im geteilten Kalender (Notion) bewältigt.
- *Frontend*  
Eines der zentralen Anliegen bei der Zusammenarbeit von Backend und Frontend betrifft die Datenbank, konkret MariaDB, und damit verbundene Netzwerkprobleme. Es ist von entscheidender Bedeutung, keine falschen Eingaben abzufangen. Auch bei der Umsetzung des Frontenddesigns gab es Schwierigkeiten. Allerdings wurden diese mithilfe der Figma-Screens als Vorlage mit Farben und Größen, Implementierung der Funktionen mit den Bibliotheken von React Native und einer ausführlichen Recherche, Test und Auswahl an geeigneter Bibliotheken bewältigt.
- *Backend*  
Die größte Herausforderung bestand darin, das ERM-Modell und die Datenbank so zu gestalten, dass sie den Anforderungen unserer App gerecht werden. Dies erforderte eine gründliche Planung. Diese Herausforderung wurde gemeistert, indem wir intensiv recherchiert, iterative Modellierung durchgeführt und unsere Lösungen ständig verbessert haben.
- *Design*  
Wie auch später im Berufsleben erhielten wir für die Umsetzung des Projekts zwar zunächst grobe Vorgaben, hatten aber relativ viel Freiheit bei der Realisierung unserer App. Im Rahmen des Designs fiel die Auswahl auf das Tool Figma. Allerdings war dies ein bis dahin unbekanntes Tool. Die Herausforderung lag hier also im Kennenlernen des Tools. Durch das Lernen aus vielen Versuchen und Fehlern wurde diese Herausforderung allerdings gut bewältigt.

### 7.2 Nützliche Tools und Bibliotheken



- *Projektmanagement*  
Nützliche Tools, wie ein gemeinsamer Kalender und ein Kanban-Board, erleichterten die Organisation und Zusammenarbeit im Team erheblich. Sie ermöglichten eine effiziente Terminplanung und Aufgabenverwaltung. Die Problematik lag allerdings darin, ein gleiches Verständnis für die Dateiablagen bei Nutzung von mehr als einem Tool (Kanban und Notion) zu gewinnen.
- *Frontend*  
Besonders vorteilhaft war die Nutzung von React Native Paper, Expo und KeyboardawareScrollView, mit welcher die Verschiebung des Screens mit Textinput bei Erscheinen des Keyboards möglich war. Schwieriger war hier das Debugging sowie die Menüleiste.
- *Backend*  
DBeaver und Postman waren unverzichtbar für die Entwicklung und das Testen der Datenbank und der APIs. Darüber hinaus hat sich die Verwendung von Git und GitHub als essenziell für die Versionskontrolle und die Zusammenarbeit im Team erwiesen. Wir haben auch von der Integration des Projektmanagement-Tools Notion profitiert, um Aufgaben und Fortschritte zu verfolgen.
- *Design*  
Die Nutzung von Figma bewährte sich vor allem darin, dass sie es mehreren Teammitgliedern ermöglichte, gleichzeitig am Entwurf zu arbeiten. Außerdem war es von Vorteil, dass keine Installation oder Aktualisierung erforderlich war, da Figma selbst in der Cloud läuft. Dennoch gab es auch Schwierigkeiten, vor allem in Bezug auf die Leistung des Programms und die Zeit, die man brauchte, um sich in das Programm einzuarbeiten. Bei der Verwendung vieler Bilder schien die Leistung des Programms etwas beeinträchtigt zu sein. Außerdem, auch wenn die Benutzerfreundlichkeit der grafischen Benutzeroberfläche grundsätzlich gegeben ist, gibt es Funktionen und Verfahren, die zeitlich gewöhnungsbedürftig sind.

## 7.3 Bewährte Praktiken

- *Projektmanagement*  
Die Einführung wöchentlicher Besprechungen hat sich im Projektverlauf als bewährte Praxis erwiesen. Diese regelmäßigen Treffen haben eine wichtige Gelegenheit für das Team geboten, sich auf dem Laufenden zu halten, Herausforderungen zu besprechen und Fortschritte zu verfolgen. Die wöchentliche Frequenz ermöglichte es, schnell auf auftretende Probleme zu reagieren und sicherzustellen, dass das Projekt auf Kurs bleibt.
- *Frontend*  
Die Verwendung des JavaScript Inspectors in Google Chrome hat sich als hervorragendes Debugging-Tool erwiesen. Hierdurch konnten effizient Fehler identifiziert und behoben werden, was die Entwicklungszeit verkürzte und die Qualität des Codes verbesserte. Darüber hinaus bewährte sich auch die Versionsverwaltung

und Zusammenarbeit über GitHub, da sie es ermöglichte den Code effizient zu verwalten und Änderungen besser nachzuverfolgen.

- *Backend*  
Best Practices, die sich im Laufe des Projekts im Backend bewährt haben, sind die regelmäßigen Gespräche mit dem Frontend und die Aktualisierung von Dokumentationen gewesen. So konnten Hindernisse oder der Bedarf vom Frontend besser identifiziert werden. Außerdem haben wir unsere Erkenntnisse und Entscheidungen in Dokumenten bzw. im Code festgehalten, was dazu beigetragen hat, ein gemeinsames Verständnis im Team aufrechtzuerhalten.
- *Design*  
Die Kommunikation innerhalb des Teams hat sich als wesentliche Best Practice erwiesen. Je mehr wir uns untereinander abgesprochen haben, desto besser und genauer waren die erstellten Screens.

## 7.4 Wissenslücken und Engpässe

- *Projektmanagement*  
Einen Engpass gab es im Projektmanagement hinsichtlich dem Klären von Konflikten. Diese wurden aber durch Rücksprache mit dem Kunden / Auftraggeber behoben.
- *Frontend*  
Durch das Recherchieren von unbekanntem Wissen im Internet, konnten Wissenslücken geschlossen werden.
- *Backend*  
Während des Projekts sind Wissenslücken und Engpässe aufgetreten, insbesondere beim Übergang von Frontend zu Backend. Um diese zu überwinden, haben wir regelmäßige Treffen und Wissenstransfers innerhalb des Teams durchgeführt. Um weiteres fehlendes Wissen aufzufüllen haben wir hauptsächlich externe Ressourcen wie Online-Tutorials und Dokumentationen genutzt.
- *Design*  
Die Wissenslücken in Bezug auf das Kennenlernen und Einarbeiten von Figma wurden durch hilfreiche Informationen von Teammitgliedern und Recherchen im Internet beseitigt.

## 7.5 Persönliche Lernergebnisse

- *Projektmanagement*  
Im Rahmen des Projekts wurden die Kompetenzen Strategisches planen, Agilität, Konfliktmanagement und Kundenorientierung angeeignet.

- *Frontend*  
Die wesentlichen Lernergebnisse liegen in der App-Entwicklung für iOS mit React Native samt Bibliotheken, JavaScript, Nutzung von GitHub sowie agile Vorgehensmodelle.
- *Backend*  
Dieses Projekt hat uns viele wertvolle Fähigkeiten und Erfahrungen vermittelt, die sich in Zukunft als nützlich erweisen werden. Nicht nur technischen Fähigkeiten in den Bereichen Backend-Entwicklung und Datenbankmanagement wurden verbessert, sondern wurde auch gelernt, wie wichtig die enge Zusammenarbeit im Team und die klare Kommunikation sind. Darüber hinaus haben wir gelernt, wie man sich an wechselnde Anforderungen und unvorhergesehene Herausforderungen anpassen kann, um erfolgreiche Lösungen zu finden.
- *Design*  
Im Laufe dieses Projekts wurde uns bewusst, dass es eine enorme Anzahl von Techniken und Möglichkeiten zur Softwareentwicklung gibt. Neben dem Kennenlernen vom konkreten Tool Figma, konnten wir auch an unserer Teamarbeits- und Problemlösungskompetenz arbeiten.

## 8 Nützliche Links

- Github: <https://github.com/blubfix/querify>
- Wireframes:  
<https://www.figma.com/file/l5xxB4rJCpDQJBumdWZY04/querify?node-id=107%3A931&mode=dev>