

Duale Hochschule Baden-Württemberg Mannheim

# **Webprogrammierung Projekt**

## **Instrumentenverleih**

### **Dokumentation**

**Marcel Mildenberger (Mr. Nr. 9011160)**

**&**

**Frederick Orschiedt (Mr. Nr. 2215312)**

Kurs: WWI20SEB

Studiengangsleitung: Prof. Dr. Thomas Holey

Prüfer: Prof. Dr. Konrad Preiser

Abgabedatum: 03.07.2022

# **Inhaltsverzeichnis**

Anforderungen.....	1
Systementwurf.....	1
Entity Relationship Diagram .....	1
High Level Architecture.....	3
Funktionen .....	4
Login.....	4
Profilseite .....	6
Bestellprozess .....	6
Navigation .....	9
Responsive Design .....	9
Unique Selling Point – Welches Instrument passt zu dir? .....	12
Verwendete Technologien.....	13
NextJS.....	13
PostgreSQL.....	14
Prisma .....	14
React .....	14
Tailwind CSS.....	15
NextAuth .....	15
Weiterführende Ideen .....	15
Fazit.....	16
Arbeitsteilung.....	17

# **Anforderungen**

In der Aufgabenstellung sollte es darum gehen, einen Onlineverleih für Instrumente zu entwickeln. Es reicht aus, wenn die Seite lokal auf dem Rechner läuft. Man soll als Kunde eine Auswahl aus verschiedenen Instrumenten haben, die man buchen und dann ausleihen kann. Dabei sollten auf der Webseite Bilder, Videos, Töne oder Ähnliches eingebaut sein. Außerdem wäre ein USP (Unique Selling Point) wünschenswert, aber nicht verpflichtend. Das Design der ganzen Seite sollte schön anzuschauen sein und stabil sein, also auch funktionieren, wenn die Seite auf einem Tablet oder Smartphone geöffnet wird. Als Programmiersprachen bieten sich HTML, CSS und JavaScript an. Verwendet werden soll ein Framework wie Vue, React oder Angular. Aus Zeitgründen ist ein Backend nicht verpflichtend.

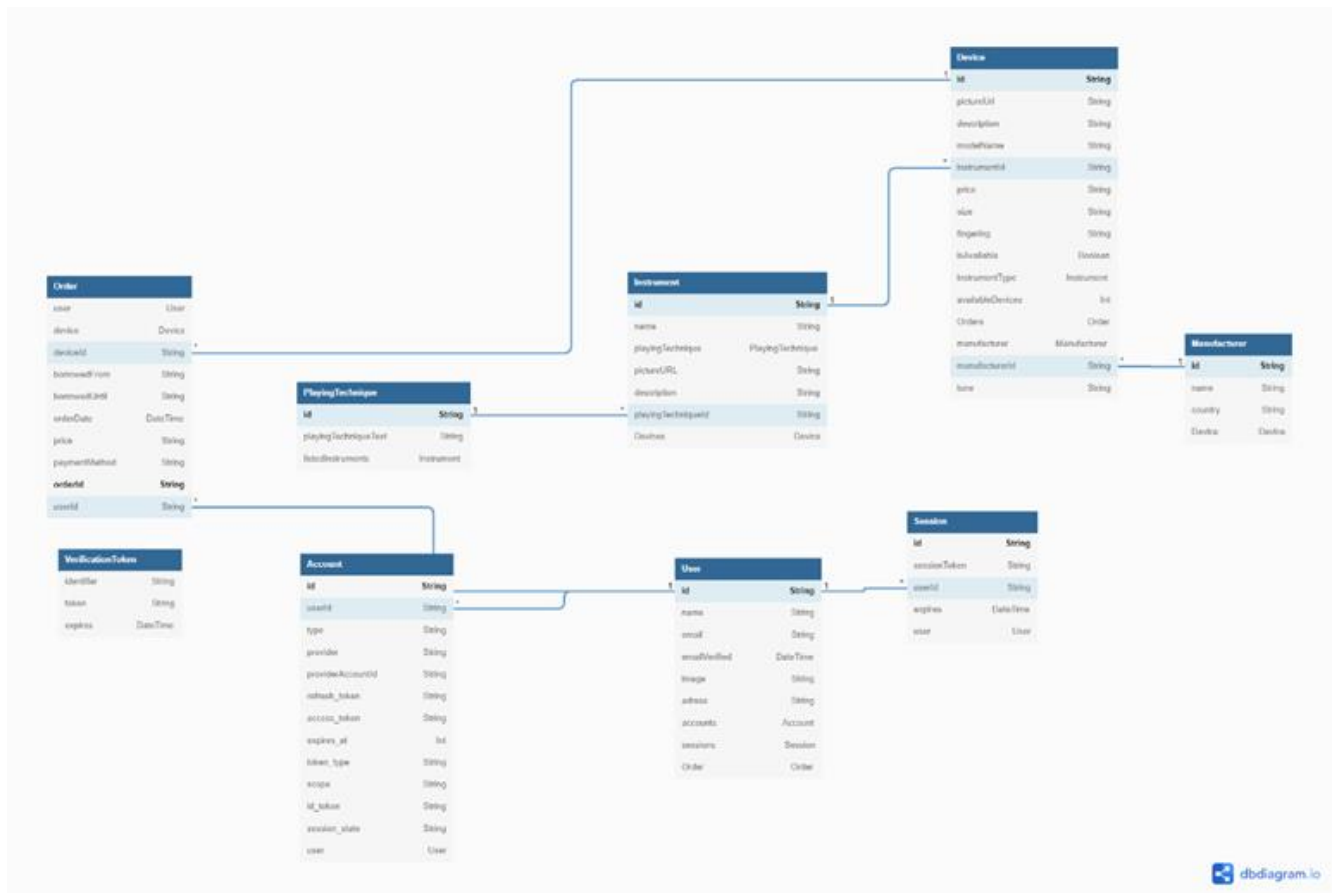
## **Systementwurf**

### **Entity Relationship Diagram**

Da wir uns für unsere Anwendung dazu entschieden haben ein Backend (Next.js API Pages) zu entwickeln, um RESTFUL-API Calls zu machen haben wir uns auch dazu entschieden eine richtige Datenbasis aufzubauen. Hierbei haben wir uns für eine PostgreSQL Datenbank entschieden. Unser aktuelles Datenmodell hat sich hierbei aus einer Wiederholung von „Trial-and-Error“ zusammengesetzt. Anfangs waren unsere ersten Entwürfe nicht ausreichend genug, um die Thematik des Instrumentenverleihs wahrheitsgetreu abzubilden. Gegen Ende hat sich dann aber eine Datenstruktur auskristallisiert, mit der wir arbeiten konnten. Dabei ist zu erwähnen, dass das Datenschema für das User Management hierbei durch „Next Authentication“ vorgegeben wird (Dazu später mehr). Die Tabellen „VerificationToken“, „Account“, „User“ und „Session“ sind die hierbei automatisch erstellten Tabelle, um User Informationen zu speichern und einen sicheren Login und Sessions zu ermöglichen. Ergänzt wurden diese Tabellen dabei um Felder wie „Adresse“ bei „User“. Hierbei haben wir eine 1..\* Relation zwischen „User“ und „Session“ sowie eine 1..\* Relation zwischen „User“ und „Account“. Dies hängt damit zusammen, dass ein User verschiedene Login-Methoden (Provider) verwenden kann, solange es jedoch beispielweise über die gleiche E-Mail läuft einem User zugeordnet ist. Unsere selbst konstruierte Datenbasis beginnt dann ab der 1..\*

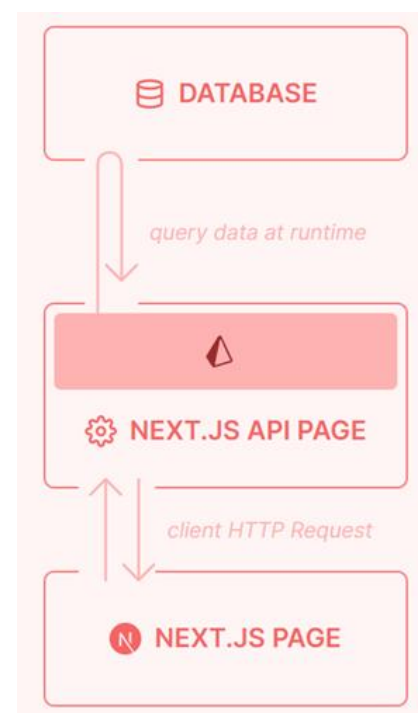
Beziehung zwischen „User“ und „Order“. Ein User kann mehrere Bestellungen haben. Bestellungen sind hierbei die Ausleihen für ein spezifisches Instrument. Dies erklärt auch die \*...1 Beziehung zwischen „Order“ und „Device“. Ein Device ist hierbei ein bestimmtes Instrument (beispielsweise Die Violine Opus 6). Dieses Device kann mehrmals vorhanden sein. Dies wird gepflegt über das Feld „availableDevices“ in der Tabelle „Device“. Eine Order enthält nun immer genau einen User und ein Device (Peter leiht die Violine Opus 6 aus). Da das Device (spezifisches Instrumentenmodell) öfters vorhanden ist, kann es in mehreren Orders auftauchen. Ein „Device“ hat nun eine \*..1 Beziehung zu der Tabelle „Manufacturer“. Hier werden lediglich die verschiedenen Instrumentenhersteller gepflegt. Ein Device ist hierbei immer genau von einem Hersteller. Die \*..1 Beziehung zwischen „Instrument“ und „Device“ ist nun für die Anwendung maßgeblich. Die Tabelle „Instrument“ bezieht sich hierbei auf die übergeordnete Instrumenten Art (beispielsweise Geige) und Device auf ein spezifisches Modell (Violine Opus 6). Deswegen kann ein Instrument (Geige) viele Devices (verschiedene Modelle) haben. Ein Device ist dabei immer genau einem Instrument zugeordnet. Um eine noch feinere Unterteilung vorzunehmen, gibt es nun die 1..\* Beziehung zwischen „PlayingTechnique“ und „Instrument“. PlayingTechnique ist hierbei die Spielweise (beispielsweise Streichinstrumente). Hierzu gehören mehrere Instrumente (Beispielweise Geigen, Bratschen). Und zu diesem Instrument gehören verschiedene Ausprägungen/Modelle bzw. „Devices“ (Beispielweise Violine Opus 6).

So baut sich unser Datenmodell vom Groben ins feine auf: Unterteilung in Spielweise => Unterteilung in Instrumente dieser Spielweise => Unterteilung in die verschiedenen Modelle, welche dann ausgeliehen werden können.



## High Level Architecture

Für unsere Webanwendung haben wir uns entschieden eine drei Schichten Architektur zu wählen. Dies steht im Einklang mit Next.js, welches die Möglichkeit bietet sowohl Frontend (Mit React) als auch Backend (Mit Node.js) zu entwickeln. Die GUI Schicht ist hierbei die Next.js Pages. Next.js ist hierbei ein Framework für React, weshalb die Pages mit React entwickelt wurden (Dazu später mehr). Zusätzlich bietet Next.js nun die Möglichkeit API Routes zu nutzen. Diese API Routes sind das Backend (Node.js) unserer Anwendung und sind auch mit JavaScript geschrieben, wodurch es eine reine Node Anwendung ist. Das Frontend sendet hierbei, wie sich in der Abbildung erkennen lässt, HTTP Requests an die API Pages der Anwendung. In diesen von uns entwickelten API Pages nutzen wir nun Prisma als



Ergänzung (oder auch Adapter) um Datenbankzugriffe (3. Schicht) auf unsere auf Heroku gehosteten PostgreSQL vorzunehmen (Prisma ist hierbei durch eine Pyramide symbolisiert). Durch diesen Aufbau konnten wir effizient und schnell eine vollständige REST-API entwickeln, welches für unsere Webanwendung zur Verfügung steht und mit der Datenbasis verbunden ist. Durch diese logische Aufteilung in drei Schichten war es uns möglich gezielt Arbeitspakete für verschiedene Schichten zu definieren. Dabei haben wir uns für das drei Schichten Modell entschieden, da wir dies bereits aus der Theorie kannten. Dies ist jedoch unsere erste Praktische Umsetzung eines solchen Modells.

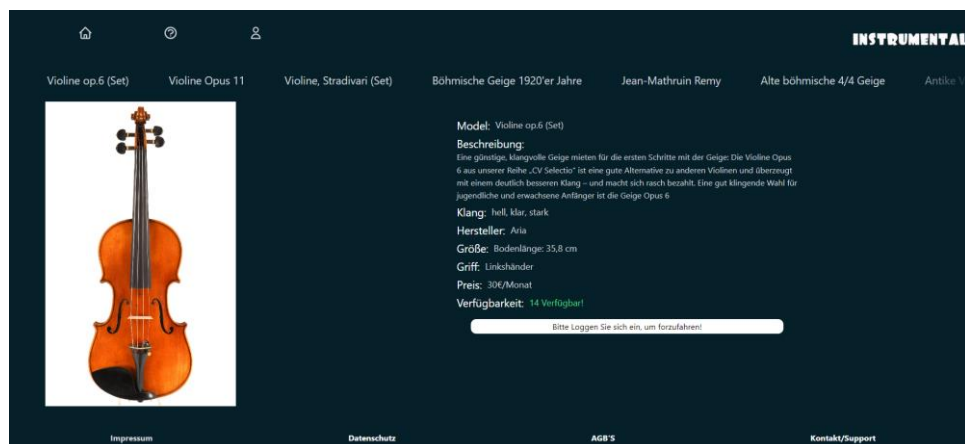
## Funktionen

### Login

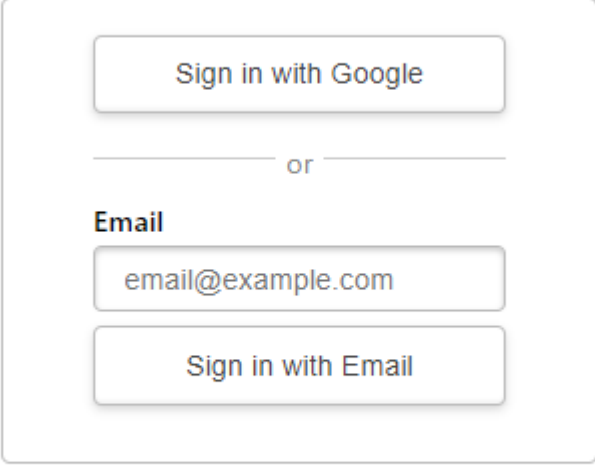
Wenn ein User unsere Website betritt und nicht eingeloggt ist (beispielsweise beim ersten mal) so findet er den Header folgendermaßen vor:



Er kann nun die Funktionalitäten unserer Website im vollen Umfang nutzen, bis er ein Instrument ausleihen will. Dabei findet er folgenden Bildschirm vor:



Der User wird dazu aufgefordert sich Einzuloggen, um fortfahren zu können. Dies kann er nun, indem er entweder den Button „Bitte loggen Sie sich ein!“ oder das User-Icon im Header drückt. Durch beide verfahren gelangt er nun auf folgende Ansicht:



The image shows a login form with a light gray border. At the top is a button labeled "Sign in with Google". Below it is a horizontal line with the word "or" in the center. Under the line is the label "Email" in bold. Below the label is a text input field containing the placeholder text "email@example.com". At the bottom is a button labeled "Sign in with Email".

Hier hat der User nun die Möglichkeit sich mit Google einzuloggen oder sich per E-Mail zu registrieren. Dabei wird jeweils kein Passwort benötigt (Beim „Sign in with Google“ muss er sich lediglich in sein Google Konto einloggen und bei dem Verfahren mit E-Mail erhält er einen einmaligen Login Link). Hier können wir dem Instrumentenverleih anbieten auf Wunsch verschiedene weitere Login Provider zur Verfügung zu stellen. Loggt er sich entweder mit E-Mail oder Google ein gelangt er wieder auf die Startseite und am Header kann man erkennen, dass er nun eingeloggt ist:



Das Icon mit dem "M" in diesem Beispiel ist das Profilbild auf Google (wenn diese Login Methode gewählt wurde). Durch drücken dieses Accounts Button gelangt der Benutzer nun auf seine Profilseite. Mit dem Logout Symbol nebendran kann der User sich jederzeit Ausloggen.

Zu erwähnen ist noch, dass ein User, der sich das erste Mal anmeldet auf folgende Ansicht geleitet wird, um seine Accountinformationen zu vervollständigen:

The screenshot shows a dark-themed registration form for 'INSTRUMENTAL'. At the top, there is a navigation bar with icons for home, search, user profile, and cart. The main heading is 'Bitte vervollständigen Sie folgende Informationen'. Below this, there are five input fields: 'Name', 'Straße', 'Hausnummer', 'Stadt', and 'Postleitzahl'. At the bottom of the form, there is a checkbox labeled 'Datenschutz & AGB's akzeptieren'.

## Profilseite

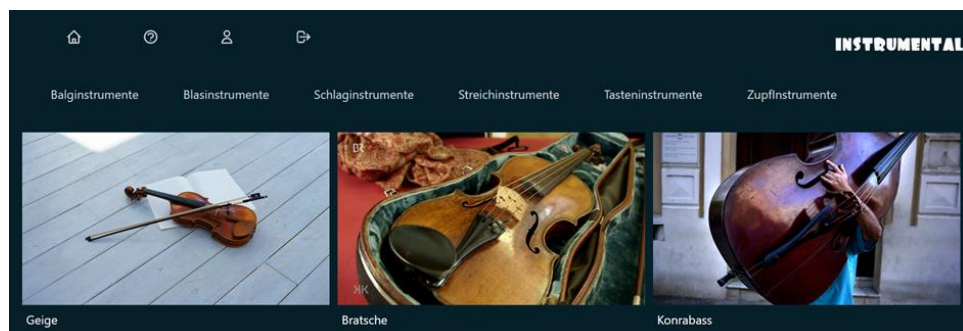
Auf der Profilseite hat der User die Möglichkeit seine User Informationen zu bearbeiten und seine Bestellungen inklusive Bestellnummern und Informationen über diese einzusehen:

The screenshot shows the user profile page for 'INSTRUMENTAL'. The page has a dark theme. At the top, there is a navigation bar with icons for home, search, user profile, and cart. The main heading is 'Ihre Accountinformationen' with an edit icon. Below this, the user's information is displayed: 'Name: Max Mustermann', 'Email: test@papierkorb.me', and 'Adresse: Musterstraße 10, 68199 Mannheim'. Below the account information, there is a section titled 'Ihre Bestellungen'. A specific order is highlighted in a box with the following details: '#lgYO84Cg9G8xm6', 'Instrument: Violine op.6 (Set)', 'Bestelldatum: 02.07.22', 'Zeitraum: 02.07.22 - 23.07.22', 'Preis: 21.00€', and 'Bezahlmethode: Paypal'. At the bottom of the page, there are links for 'Impressum', 'Datenschutz', 'AGB's', and 'Kontakt/Support'.

Durch den Stift Button gelangt der User auf eine Seite zur Bearbeitung seiner Profilinformationen

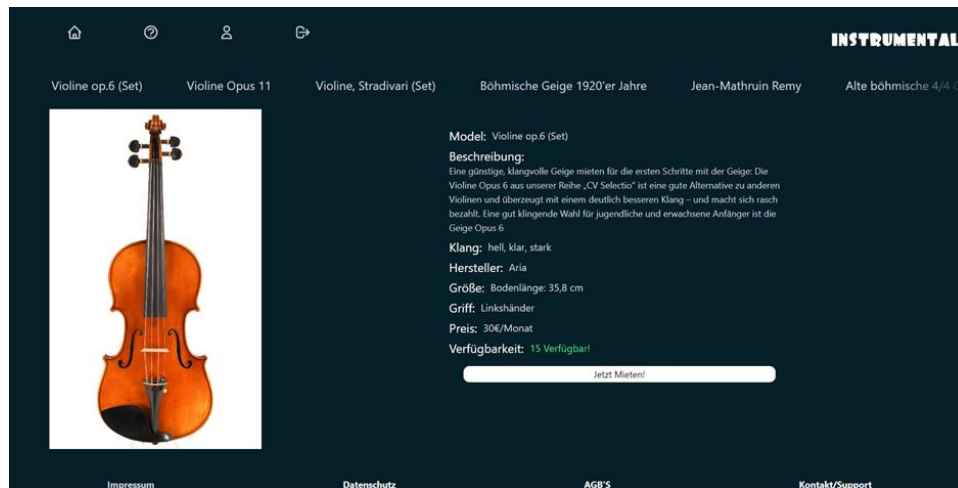
## Bestellprozess

Der Bestellprozess ist simple und intuitiv gestaltet. Wenn der User eingeloggt ist findet er die Seite wie folgt beim Aufrufen vor:

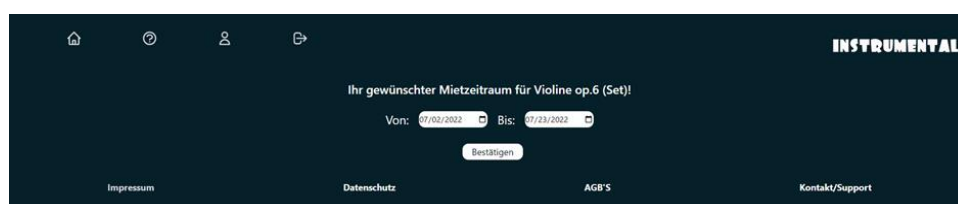




Nun kann er sich in der Navigation Bar für eine der Spielweisen entscheiden. In diesem Fall werden Standardmäßig die Streichinstrumente geladen. Nun kann der User, wenn er eine Geige ausleihen möchte, einfach über das Geigenbild Hovern und sich eine kurze Beschreibung dazu durchlesen und anschließend draufklicken, um auf die nächste Seite zu gelangen:



Auf dieser Seite wird nun Standardmäßig das günstigste Instrument geladen. Die Navigationsbar hat sich nun in die verfügbaren Geigen Modelle verändert und sind aufsteigend nach dem Preis sortiert. Nun kann der User sich die verschiedenen Geigen anschauen und vergleichen, um sich anschließend für eine zu Entscheiden. Hat er sich entschieden drückt er anschließend „Jetzt Mieten!“ und gelang auf folgenden Bildschirm:



Hier wählt der User nun den gewünschten Zeitraum zum Ausleihen aus. Dabei muss dieser mindestens eine Woche betragen. Hat er sich dazu Entschieden, wie lange er das Instrument ausleihen will, klickt er auf Bestätigen um zur Bestellübersicht zu gelangen:

Warenkorb

Instrument: Violine op.6 (Set)  
Zeitraum: 02.07.22 - 23.07.22  
Preis: 21.00€

Accountinformationen

Name: Max Mustermann  
Email: max.mustermann@muell.monster  
Adresse: Musterstraße 10, 68199 Mannheim

Bezahlmethode

☒ Paypal ☐ Klarna ☐ Kreditkarte

☒ Kaufbedingungen akzeptieren

Jetzt Zahlungspflichtig bestellen

Impressum Datenschutz AGB'S Kontakt/Support

Hier kann der User nun nochmal seinen Warenkorb einsehen und überprüfen, welches Instrument er für wie lange zu welchem Preis ausleihen möchte. Zudem kann er seine Accountinformationen überprüfen, ob alles stimmt und anschließend die Bezahlmethode auswählen und die Kaufbedingungen akzeptieren. Wenn er nun „Zahlungspflichtig bestellen“ auswählt gelangt er zur folgenden Übersicht:

Vielen Dank für ihre Bestellung!

Bestellnummer: #oRMovvy7USj2KN4

Sie können nun zur Startseite zurückkehren oder in ihrem Profil ihre Bestellungen einsehen und verwalten. Das Instrument können sie sich an unserem Standort, sobald ihr Mietzeitraum beginnt, abholen.

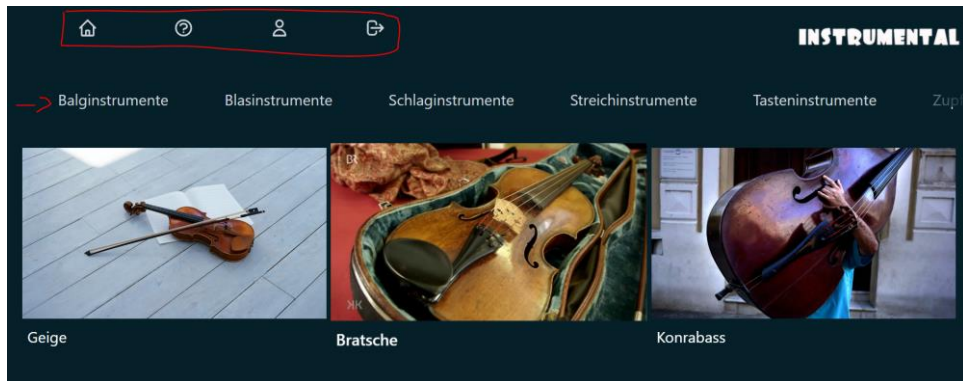
Zurück zur Startseite!

Impressum Datenschutz AGB'S Kontakt/Support

Wenn die Bestellung erfolgreich war gelangt er nun auf diese Bestätigungsseite. Hierbei kriegt der Benutzer seine Bestellnummer mitgeteilt und kann nun den Bestellprozess durch „Zurück zur Startseite“ abschließen. Ob der Instrumentenverleih das Instrument nun verschicken will an die Adresse des Users oder nur zur Abholung anbieten möchte, kann hier gepflegt und dem User mitgeteilt werden.

## Navigation

Die Navigation wird ermöglicht durch Buttons im Header oder der Dynamischen Navigationsbar, wo die verschiedenen Spielweisen der Instrumente gerendert werden:



## Responsive Design

Um ein Responsive Design zu ermöglichen wurde auf Tailwind CSS zurückgegriffen. Tailwind bietet die Möglichkeit mit Breakpoints von verschiedenen Bildschirmgrößen zu Arbeiten. Dabei verlaufen diese von klein zu groß und das CSS wird dynamisch für verschiedene Breakpoints angepasst. Dabei gibt es folgende Breakpoints (Bildschirmgrößen):

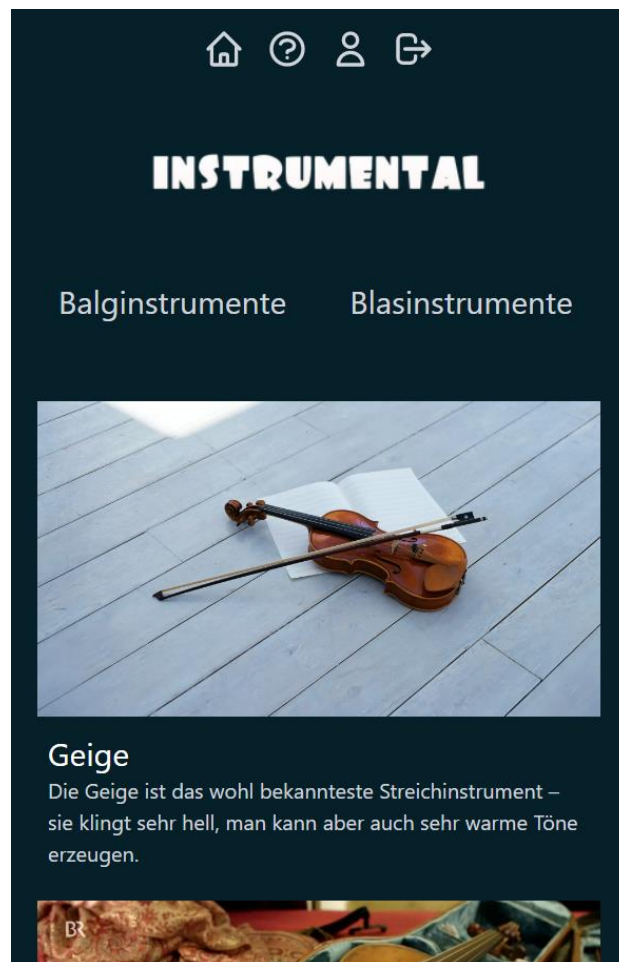
Small => Medium => Large => XL => 2XL

Unsere Anwendung wurde unter dem Motto "Mobile First" entwickelt. Das heißt, dass wir erst die Mobilansicht entwickelt haben und dann nach und nach über die Breakpoints das CSS an verschiedene Bildschirmgrößen angepasst haben. Dabei ist zu erwähnen, dass durch diesen Approach die „Mobile Expierence“ am besten ist.

Wir haben dies für jede Seite unsere Anwendung und jedes Element vorgenommen. Um einen Überblick zu verschaffen, wie dies aussehen könnte, vergleichen wir die Startseite auf verschiedenen Bildschirmgrößen:

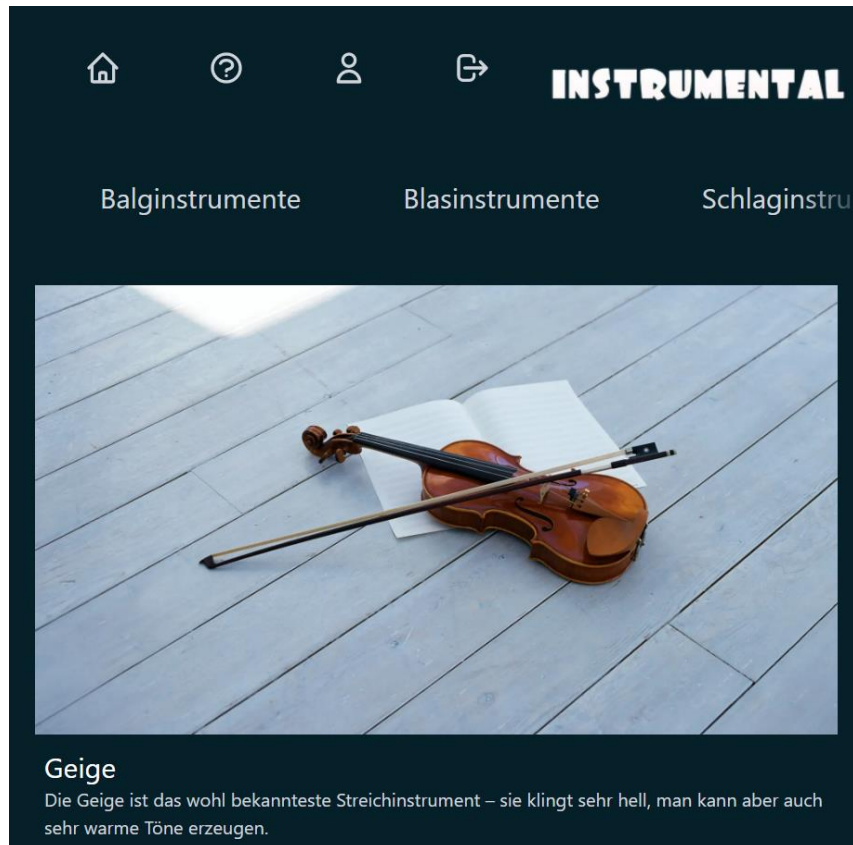
## Small Screen:

(Flex-Col)



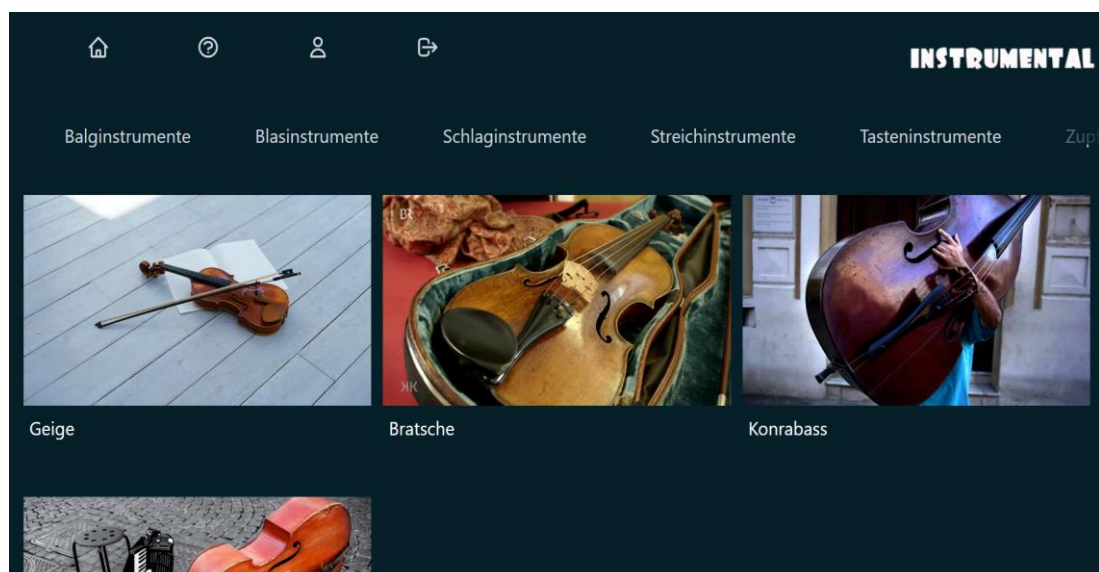
## Medium Screen:

(Flex-Col, Header passt sich an)



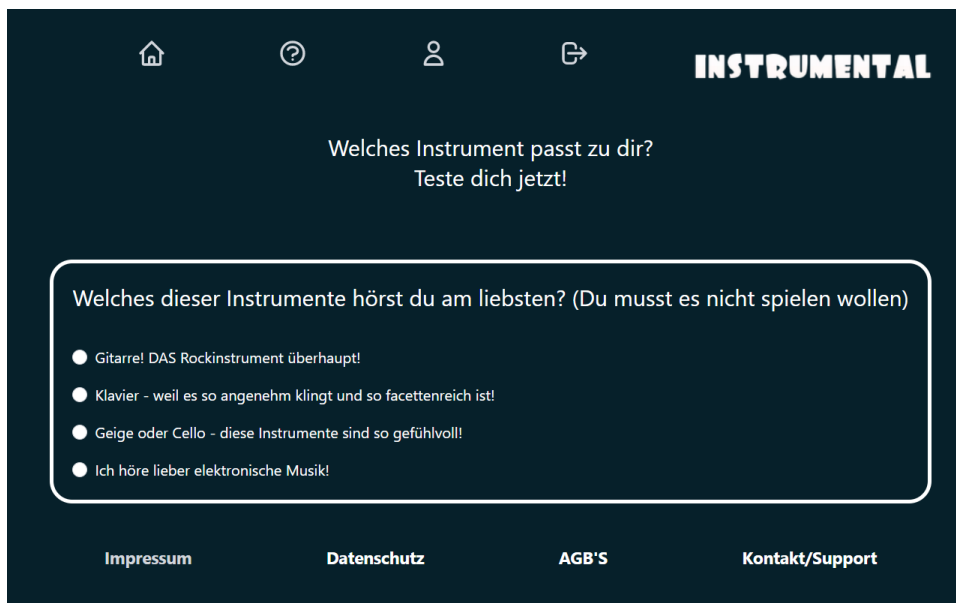
## Large Screen:

(Grid mit 3 Spalten)



## Unique Selling Point – Welches Instrument passt zu dir?

Als Unique Selling Point bietet die Website ein Feature, in dem der Benutzer sehen kann, wie gut ein Instrument zu ihm passt. Dazu gibt es ein Quiz, welches der Benutzer durchführen kann. In dem Quiz werden verschiedene Fragen zu den Interessen des Benutzers, worauf die Webseite am Ende ein Instrument berechnet, welches gut zu dem Benutzer passen würde. So ist die Webseite nicht nur für Benutzer, die bereits genau wissen welches Instrument sie ausleihen wollen, sondern eben auch für Benutzer, die etwas Neues ausprobieren wollen und dafür eine Inspiration benötigen. Dieses vorgeschlagene Instrument kann dann auf der gleichen Webseite direkt ausgeliehen werden. Dabei besteht das Quiz aus 12 Fragen, welche folgendermaßen aufgebaut sind:

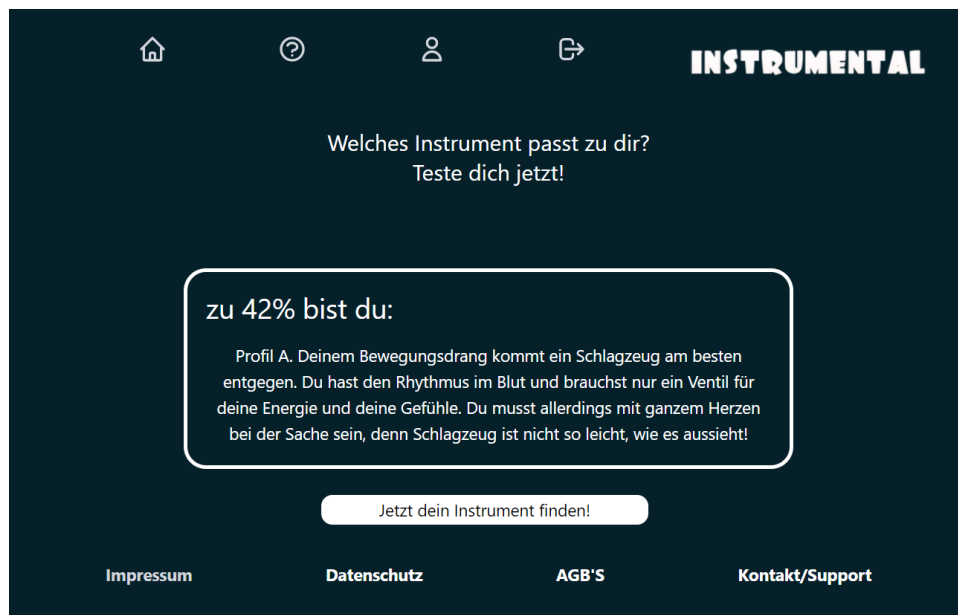


The screenshot shows a dark-themed web interface for a quiz titled "Welches Instrument passt zu dir? Teste dich jetzt!". At the top, there is a navigation bar with icons for home, help, user profile, and share, followed by the "INSTRUMENTAL" logo. The main content area contains a question: "Welches dieser Instrumente hörst du am liebsten? (Du musst es nicht spielen wollen)". Below the question are four radio button options:

- ☐ Gitarre! DAS Rockinstrument überhaupt!
- ☐ Klavier - weil es so angenehm klingt und so facettenreich ist!
- ☐ Geige oder Cello - diese Instrumente sind so gefühlvoll!
- ☐ Ich höre lieber elektronische Musik!

At the bottom of the page, there is a footer with links for "Impressum", "Datenschutz", "AGB'S", and "Kontakt/Support".

Dabei hat der User immer die Möglichkeit zwischen 4 Antwortmöglichkeiten auszuwählen. Der User wird dann am Ende des Quiz in eines von vier Kategorien eingeteilt (Profil A, B, C und D). Jede Kategorie ist hierbei einem Instrument und Persönlichkeit zugeordnet. Der Ausgang könnte beispielsweise nach den 12 Fragen wie folgt aussehen:



## Verwendete Technologien

### NextJS

NextJS ist ein React Framework auf dem neusten Stand der Technik. Es bietet Möglichkeiten Fullstack, also sowohl im Frontend als auch Backend, zu entwickeln. In unserem Fall wird im Frontend React benutzt und im Backend NodeJS. NextJS bietet dabei die Möglichkeit API Routes zu nutzen um die Kommunikation zwischen Frontend und Backend zu ermöglichen. Eine Besonderheit von NextJS ist, dass die Webseite sowohl clientseitig, als auch serverseitig gerendert werden kann, wodurch NextJS so flexibel gegenüber anderen Frameworks ist. Außerdem macht es standardmäßig pre Rendering, also generiert schon im Vorhinein HTML für eine Seite. Da diese vor dem Aufruf schon gerendert ist, ist sie für den Nutzer besonders schnell verfügbar. Bei der Statischen Generation wird alles im Build einmal gerendert und bei jedem Request darauf zugegriffen, bei dem serverseitigen Rendering wird bei jedem Request neu gerendert. NextJS bietet hier die Möglichkeit eines hybriden Systems, also zum Beispiel aus Performancegründen die meisten Seiten statisch zu generieren und aus Präzisionsgründen ein paar Seiten serverseitig zu rendern. Des Weiteren bietet NextJS standardmäßig die Möglichkeit des „Fast Refresh“, welches sich auf Änderungen im Code bezieht. Ändert man ein einzelnes File, welches nur React Komponenten enthält, muss auch nur dieses File neu gerendert werden. Das vereinfacht zusätzlich das Entwickeln mit NextJS.

## **PostgreSQL**

Als Datenbank Managemant System benutzten wir PostgreSQL. Es ist ein zuverlässiges und modernes Open Source Projekt, welches eine gute kompatibilität mit den anderen Komponenten aufweist. Dadurch, dass es sich um ein Open Source Projekt handelt, ist es transparent und befindet sich immer auf dem neusten Standard. Datenbankmanagementsysteme unterliegen dem CAP Theorem. Dieses besagt, dass aus den folgenden drei Merkmalen nur zwei wirklich erfüllt werden können: Konsistenz (Consistence = C), Verfügbarkeit (Availability = A) und Ausfalltoleranz (Tolerance of Network Partitions = P). PostgreSQL ist grundsätzlich ein CA-System, also konsistent und hoch verfügbar. Eine Besonderheit an PostgreSQL ist, dass man es umformen kann, um auch ein AP-System zu haben. In vielen Fällen ist das zwar nicht relevant, jedoch macht es PostgreSQL zu einer besonders flexiblen Lösung. Des weiteren ist PostreSQL kostenlos für jedermann zu benutzen. Andere Gruppen von Kommilitonen machten in anderen Programmierprojekten bereits gute Erfahrungen mit PostgreSQL, weshalb wir es nun damit auch probiert haben.

## **Prisma**

Prisma ist ein ORM (object relational mapping) Tool um einfacher mit Datenbanken arbeiten zu können. Dabei wird viel manuelle Arbeit abgenommen, die das Schreiben von vielen SQL Statements in Anspruch genommen hätte. Wir selbst hatten zuvor noch keinerlei Erfahrung mit diesem Tool gemacht. Mit dem verwendeten Datenbank Management System PostgreSQL ist Prisma sehr gut kompatibel. Wie viele andere hier verwendete Technologien ist Prisma ein Open Source Projekt und kostenlos, weshalb es sich gut für dieses Projekt eignete.



## **React**

Um die Implementierung zu vereinfachen, wurde zudem Gebrauch von der JavaScript-Softwarebibliothek React gemacht. React ermöglicht es, effizient GUIs zu bauen. Dabei werden (wiederverwendbare) Komponenten definiert, welche immer wieder innerhalb der App verwendet werden können. Zudem ermöglicht React die Nutzung von JSX (JavaScript XML). JSX wird dazu verwendet, Komponenten in JavaScript zu definieren mit einer Syntax, die sehr ähnlich zu HTML ist. JSX ist eine Erweiterung der JavaScript-Sprachsyntax. React kann man also als eine Art Verbindung von JavaScript und HTML Elementen sehen, wobei die Vorteile und Anwendungsbereiche aus beiden Sprachen übernommen werden. Der React Code ist dabei vergleichsweise simpel gehalten, wodurch man einen besseren Überblick hat und leichter Änderungen vornehmen kann. Mit den anderen hier verwendeten Technologien ist React gut kompatibel.

## **Tailwind CSS**

Tailwind CSS ist ein modernes CSS Framework. Die Besonderheit besteht darin, wie gut dieses sich für Responsive Designs eignet. Es bietet Möglichkeiten im Design verschiedene Breakpoints zu setzen, an welchen sich dann im Responsive Design orientiert wird. Im Kapitel „Responsive Design“ wurde bereits näher darauf eingegangen. Da wir den Fokus stärker auf die mobile Anwendung gesetzt haben war Tailwind CSS hier die richtige Wahl.

Zudem bietet Tailwind CSS die Möglichkeit Inline CSS zu verwenden und zwar in eigener Form. Dadurch lässt sich CSS sehr effizient schreiben und Änderungen werden durch den Fastrefresh sofort erkennbar. Somit Tailwind CSS maßgeblich unsere Entwicklungsgeschwindigkeit gesteigert auch wenn wir uns erst in diese neue Technologie einarbeiten mussten.

## **NextAuth**

Um bei der Anmeldung die Authentifizierung zu ermöglichen wird das Tool NextAuth benutzt. Als Authentifizierung sind verschiedene OAuth Provider wie Github, Twitter, Google oder Apple möglich. In unserem Fall ist die Anmeldung mit einem Google Konto möglich. Dabei schickt unsere Webseite einen Callback an Google, worauf Google die Authentifizierung ermöglicht. Eine Registrierung und Anmeldung mit E-

Mail ist mit NextAuth ebenfalls möglich. Es ist kompatibel mit den gängigsten Datenbank Management Systemen, so auch das hier verwendete PostgreSQL und harmonisiert mit Prisma als Adapter. Wenn ein User sich über Google anmeldet so reagiert der Prisma Adapter und schreibt automatisch die User, Account und Session Informationen in unsere PostgreSQL Datenbank.

## Weiterführende Ideen

Auch wenn unsere Anwendung bereits viele Aspekte abdeckt, so gibt es doch auch hier noch weiterführende Ideen, welche man implementieren könnte. Wir haben uns aufgrund von Zeit und Ressourcen dazu entschieden diese Aspekte nicht mit in unsere Anwendung einzuarbeiten. Weiterführende Ideen für die Anwendung wären:

- Die Funktionalität eines Warenkorbes zu implementieren, sodass User gegeben falls mehrere Instrumente auf einmal Ausleihen können (Für Bands, Orchester etc.)
- Auf der Detail Seite eines Instruments die Möglichkeit geben sich Hörproben anzuhören (Müsste man Sound-Dateien für jedes Instrument besorgen)
- Eine Admin-Page erstellen, um Admins ein leichteres Pflegen der Daten zu ermöglichen anstatt es über „Prisma Studio“ oder ähnlichen DB-Viewer zu machen.
- Das Quiz so erweitern, dass man auf spezifisches Instrument zum Ausleihen direkt weitergeleitet wird.
- Einen Chatbot implementieren, um einfache User Anfragen entgegenzunehmen
- Usability Tests durchführen und an der Usability der Anwendung mit Methoden des Usability Engineerings arbeiten
- Die Möglichkeit für User schaffen, Ausleihen in ihrem Profil zu stornieren oder zu verlängern
- Mehrere Login-Provider wie Apple, Twitter, Facebook oder ähnliches implementieren (Wobei dies nach dem Aufsetzen von NextAuth (was bereits geschehen ist) sehr schnell und einfach ist)
- Einen Marktplatz implementieren, wo User untereinander Instrumente ausleihen können und dafür Gebühren verlangen
- Eine Landing Page implementieren, wo Informationen über die Ausleihe, Werbevideos oder ähnliches sein können (Wurde darauf verzichtet, da es nicht maßgeblich für die Anwendung selbst ist)
- Eine App Entwickeln, die statt der mobilen Ansicht im Browser genutzt werden kann

## **Fazit**

Obwohl wir bereits in verschiedenen Vorlesungen und Praxisphasen Kontakt zu vielen Technologien hatten, haben wir uns dazu entschieden für dieses Projekt vieles neues zu lernen und vor allem die neuen Technologien zu lernen die gerade „State of the Art“ sind. Wir haben uns beispielsweise als Grundstruktur dazu entschieden Next.js zu lernen. Es waren zwar bereits Programmierkenntnisse in JavaScript und dem JavaScript Framework React vorhanden, dennoch mussten diese Basiskenntnisse nicht nur vertieft werden sondern auch um Next.js Funktionalitäten (ServerSideRendering, API-Routes...) erweitert werden. Dazu kommt, dass wir beide keinerlei Backend Erfahrung hatten und noch nie eine REST-API gebaut haben. Genauso haben wir auch noch nie direkt mit einer Datenbank gearbeitet, in der wie die Daten persistieren mussten (Haben meist andere Gruppenmitglieder in den anderen Projekten übernommen). Also haben wir, um es kurz zu fassen, das erste Mal einen Fullstack entworfen und dabei in jeder der drei Schichten aktiv gearbeitet und uns mit den neuesten Technologien dabei auseinander gesetzt. Dementsprechend ist das Ergebnis nicht so atemberaubend wie vielleicht bei anderen Gruppen. Dennoch glauben wir, dass die Lernkurve die wir durchlebt haben enorm war. Diese neuen Erkenntnisse in JavaScript, Next.js, Node.js, Heroku, TailwindCSS, GitHub, PostgreSQL, Prisma ORM und vieles mehr werden maßgeblich Einfluss auf unsere zukünftigen Arbeiten und Projekte haben. Mit der Entscheidung die Aktuell neuesten und gefragtesten Technologien zu lernen haben wir zwar in Kauf genommen, dass das Ergebnis schlechter wird. als wenn wir auf alte Bekannte Technologien zurückgegriffen hätten, hatten aber dafür die Chance aktiv mit den neuen Technologien mal gearbeitet zu arbeiten.

## **Arbeitsteilung**

Anwendung + Dokumentation + Präsentation = 100%

Marcel Mildemberger (Mr. Nr. 9011160): 80%

Frederick Orschiedt (Mr. Nr. 2215312): 20%