Frontend – Leonard Muth

Inhaltsverzeichnis

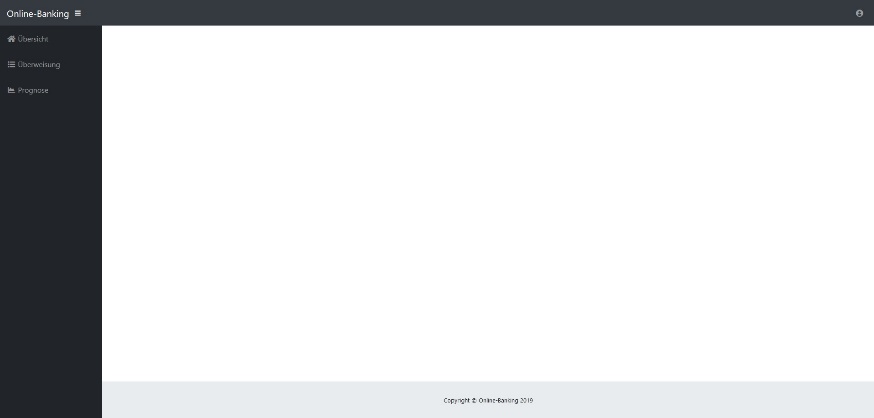
1. Allgemeines
   1. Zielsetzung
   2. Übersicht
   3. Einrichtung
2. Funktion
   1. Index
   2. Überweisung
   3. Admin
   4. Umsetzung
3. Architekturelle Entscheidungen

**1. Allgemeines**

1.1. Zielsetzung:

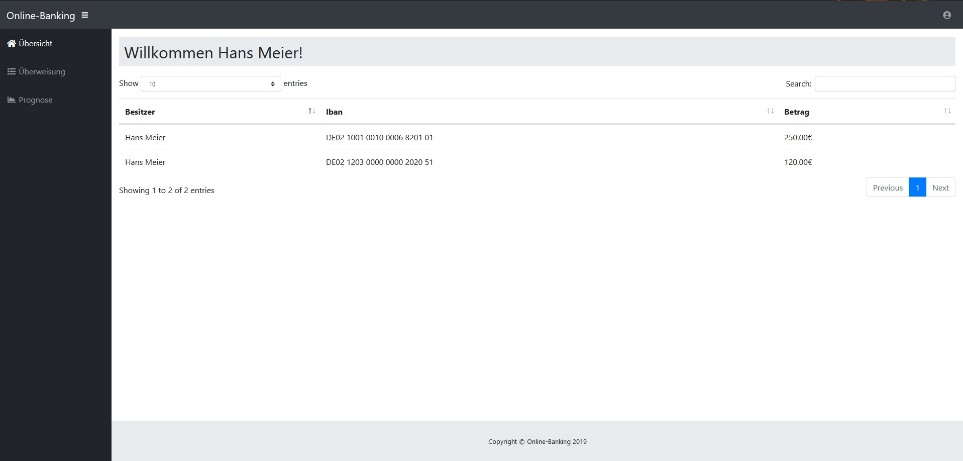
Die Ziele, die wir an diesen Teil des Frontends gesetzt haben, waren ziemlich eindeutig. Es soll eine Übersichtsseite geben, eine Seite, um Überweisungen zu tätigen und einzusehen und eine Art Admin-Bereich zur Verwaltung der Daten innerhalb der Applikation. Dazu kommt noch eine Seite, um Prognosen anzulegen und eine Login-Verwaltung. Auf Diese werde ich hier aber nicht eingehen. Insgesamt standen die Übersichtlichkeit der Seiten und die einfache Bedienung im Vordergrund. Außerdem war klar, dass die Daten, die in dieses Frontend eingespeist werden von der Datenbank bezogen werden sollten. Daher musste sichergestellt werden, dass diese Interaktion reibungslos funktioniert und vom Nutzer keine falschen Werte in die Datenbank gelangen. Genauer beschrieben, sollte die Übersicht aus den für den Kunden wichtigen Informationen bestehen. Dazu gehört die Anzeige aller Konten des Kunden, die jeweilige IBAN des Kontos und den zur Verfügung stehenden Betrag. Die Überweisungsseite sollte eine Möglichkeit haben sowohl Ziel- als auch Ausgangs-Konto auszuwählen und den zu überweisenden Betrag anzugeben. Der Admin-Bereich sollte die üblichen Funktionen (Create, Read, Update, Delete) für Konten ermöglichen und dabei möglichst einfach gehalten sein.

1.2. Übersicht:

Unser Projekt ist grundsätzlich so organisiert, dass sich alles auf einem virtuellen Laufwerk der Hochschule abspielt. Dort wird sowohl der Code, als auch die Datenbank gespeichert. Das Frontend besteht dabei aus einer HTML-Seite, die über einen Link aufgerufen wird. Die Daten müssen aber auch auf der VM bzw. auf dem Server der Hochschule gespeichert werden, da nur so sichergestellt werden kann, dass unsere Seite durchgängig und von überall erreichbar ist. Bei der Funktionalität der Anzeige des Frontends haben wir mit Bootstrap gearbeitet, da uns so viel Arbeit abgenommen wurde. Die Schnittstelle zur Datenbank haben wir mit php realisiert, da es für solche, kleinere Anwendungen am übersichtlichsten ist und am schnellsten zum Ergebnis führt. Zur Datenbankverwaltung verwenden wir MySQL. Zur Anzeige haben wir uns ein vordefiniertes Template ausgewählt, das gerade zur übersichtlichen Darstellung von Zahlen geeignet ist. So konnten wir die Funktionalität von Tabellen und Graphen von dort übernehmen. Trotz dieser Erleichterung habe ich bei diesem Teil nur das Grundgerüst des Templates verwendet, sodass eine erste Struktur auf der Seite zu sehen ist. Dadurch sind die ersten Schritte einfacher und übersichtlicher. Die eigentliche Bearbeitung erfolgte dann durch Anpassen der jeweiligen HTML-Seiten, abfragen der Daten aus der Datenbank über php und einsetzen Dieser an die richtige Stelle im Dokument. Alle optischen Veränderungen wurden, wie auf Websites üblich mit CSS vorgenommen.

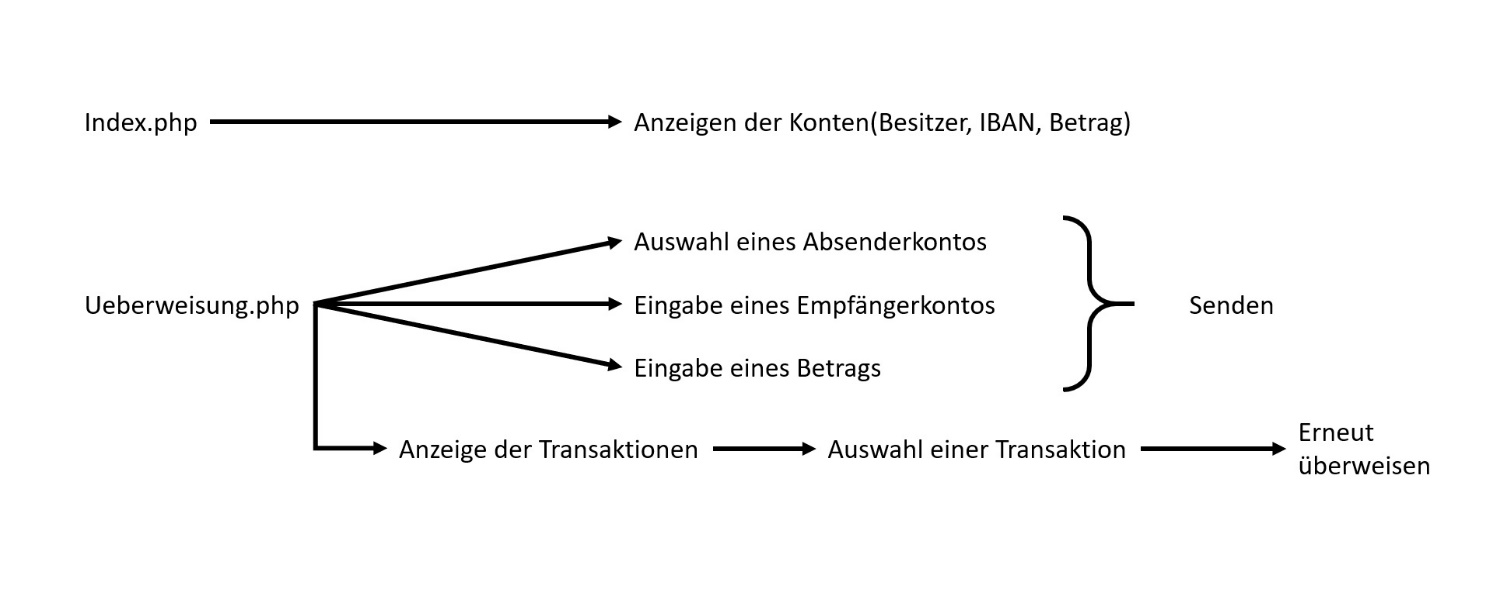
Anzeige des leeren Templates

1.3. Einrichtung:

Dadurch, dass die Online-Banking-Applikation über eine Webseite erreichbar ist, ist der Aufruf sehr einfach. Mit dem richtigen Link kann von überall über das Internet darauf zugegriffen werden. Dadurch ist auch die Übertragung und Installation unser Applikation ist sehr einfach. Dafür sind grundsätzlich nur zwei Dinge nötig. Zum einen benötigt man den Projektordner, in dem die Dateien (hauptsächlich HTML, CSS, JS, PHP) gespeichert sind, und eine Datenbank. Nach Anpassung der Datei, die die Zugangsdaten der Datenbank speichert (incl/frontend/dbconnect.php), wird die Anwendung bereits funktionsfähig sein.

Startseite

**2. Funktion:**

2.1. Index:

Hier werde ich die einzelnen Funktionen, der von mir erstellten Seiten genauer beschreiben. Die Grafik zeigt übersichtlich, was jeweils die Hauptaufgaben sind. Die index.php bzw. Übersichtsseite stellt alle Konten des im Moment angemeldeten Nutzers auf. Dies erfolgt mit Hilfe eines Datatables, einer vorgefertigten Klasse aus Bootstrap. Dort sind dann bereits viele Funktionen vordefiniert. Dazu gehört zum Beispiel die Möglichkeit zu Suchen oder nach Spalten zu sortieren. Diese Datatables finden in der ganzen Online-Banking-Application des Öfteren Verwendung, sodass es auch ein einheitliches Bild innerhalb der Webseite ergibt.

Funktionen der Index.php und Ueberweisung.php

2.2. Ueberweisung:

Die Überweisungsseite hat natürlich als Hauptaufgabe eine Transaktion von einem Konto, auf ein Anderes zu tätigen. Dazu gehört also Kontostand des Ausgangskontos um den Betrag zu verringern, den Kontostand des Zielkontos um den Betrag zu erhöhen und in die Transaktionstabelle diese Informationen zuzüglich dem Zeitpunkt des sendens zu schreiben. Um das Ausgangskonto auszuwählen, habe ich mich aus Gründen der Übersichtlichkeit und um falsche Eingaben zu vermeiden für ein select-Feld entschieden. Dort findet der angemeldete Kunde alle Konten, die ihm zugehörig sind, deren IBAN und den aktuellen Kontostand.

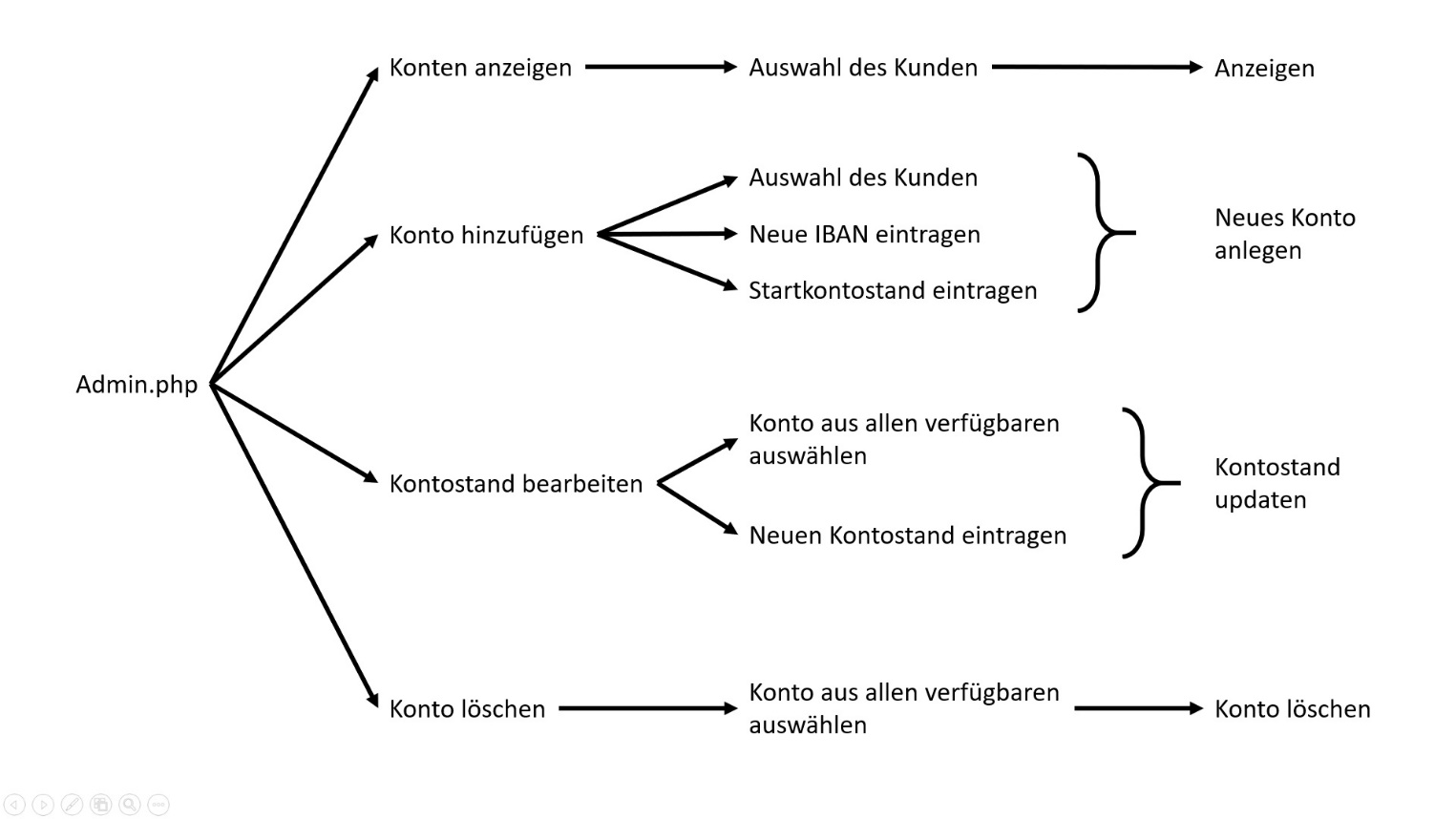
Wenn eine IBAN eingegeben wird, wird sie hinsichtlich drei Kriterien geprüft. Bei der ersten Prüfung wird die Absender IBAN mit der Empfänger IBAN verglichen. Sollten diese identisch sein, handelt es um keine korrekte Transaktion und wird daher nicht weiter ausgeführt. Die Zweite achtet auf die Nutzereingabe. Sollte der Nutzer sich verschrieben haben oder an eine falsche IBAN überweisen wollen, wird an dieser Stelle die Überweisung abgebrochen und eine Fehlermeldung angezeigt. Die dritte Prüfung ist speziell für unseren Entwicklungsraum. Dort wird die eingegebene IBAN mit denen der Datenbank verglichen. Nur, wenn Sie dort vorkommt, wird die Transaktion ausgeführt. Diese kleine Sperre ist eingebaut, da sonst Anomalien in der Transaktionstabelle bzw. Fehler entstehen könnten. Sollten die Transaktionen nicht nur über unsere Datenbank laufen, sondern an eine tatsächliche Zieladresse geschickt werden können, kann diese Abfrage entfernt werden, da durch die Korrektheit der IBAN im Normalfall sichergestellt ist, dass auch ein Konto oder zumindest ein Adressat hinter der Nummer steht.

IBAN Prüfung



Eine weitere Funktion der Überweisungsseite ist der „Erneut überweisen“-Button. Hinter diesem versteckt sich die Möglichkeit eine in der Transaktionstabelle ausgewählte Überweisung erneut zu tätigen. Dazu gehören Absender-IBAN, Empfänger-IBAN und der angezeigte Betrag. Als Zeitpunkt wird natürlich wieder die aktuelle Zeit der Transaktion genommen. Die Funktion wurde mit Hilfe von JavaScript realisiert. Dabei wird auf der Seite zuerst nach einer Zeile innerhalb einer Datatable gesucht, die angewählt ist. Das an- und abwählen von Zeilen ist eine vordefinierte Funktion der Bootstrap-Klasse und wurde sich hier zu Nutze gemacht. Sollte keine aktive Zeile gefunden werden, heißt das, dass keine Transaktion ausgewählt wurde und somit wird eine Fehlermeldung ausgegeben und abgebrochen. Sollte jedoch eine Zeile ausgewählt sein werden die Werte aus der dritten, vierten und fünften Spalte in Variablen geschrieben. Spalte eins ist unsichtbar und beinhaltet die Überweisungs\_ID, die für die Weiterverarbeitung der Transaktion wichtig ist. In der zweiten Spalte steht der Absender beziehungsweise der angemeldete Kunde. Diese Variablen werden dann umgeformt, sodass Sie für das nächste PHP-Skript lesbar sind und verschickt.

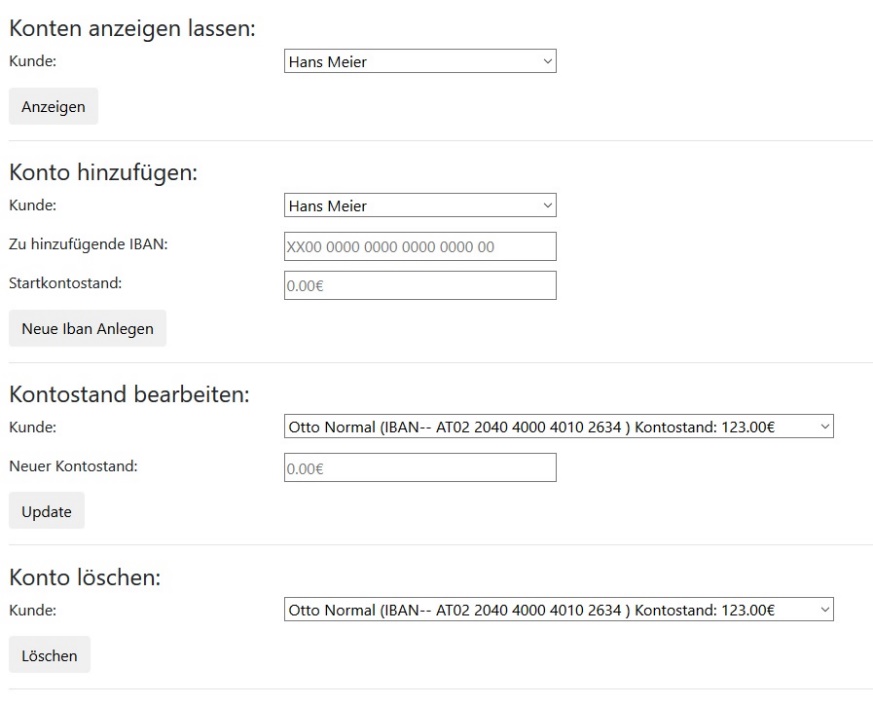
JavaScript hinter „Erneut überweisen“

2.3. Admin:

Die Admin.php ist grundsätzlich für keinen anderen Kunden, als den Admin zu erreichen. Sollte man versuchen unangemeldet auf die Seite zuzugreifen, wird man zurück zur Login Seite geleitet. Sollte man als Nicht-Admin versuchen darauf zuzugreifen, wird man auf die Startseite geleitet. Sicherheitsansprüche kann man mit dieser Lösung wahrscheinlich nicht erreichen, für diese kleine Anwendung erfüllt es aber seinen Zweck.

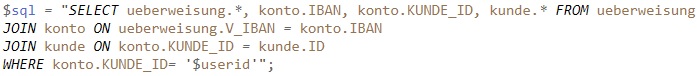
Funktionen der Admin.php

Der richtige User hat von dort dann einige Möglichkeiten. Für eine reale Banking-Application, wären diese Funktionen natürlich undenkbar, für unsere Anwendung erleichtern Sie allerdings das verwalten der Daten enorm. Bei den Bearbeitungsmöglichkeiten habe ich mich an den typischen CRUD-Funktionen orientiert. So können alle Konten eines Kunden angezeigt werden. Der Kunde wird über ein Select-Feld ausgefählt, das aus der Datenbank gespeist wird. Außerdem können Konten hinzugefügt werden. Dafür wird das gleiche Select-Feld verwendet, um den Kunden zu bestimmen. Danach kann eine IBAN eingetragen werden, die ebenfalls geprüft wird, ob Sie korrekt ist. Ich habe mich außerdem bewusst dafür entschieden, den Startkontostand nicht auf Negativität zu prüfen, da ein Konto vom Admin auch mit Schulden angelegt werden kann.

Kontostand bearbeiten verfährt ähnlich. Nur wird hier aus der Liste aller gelisteten Konten in der Datenbank ein Select-Feld erstellt, sodass falsche Eingaben durch den Nutzer von vornherein ausgeschlossen sind. Um ein Konto zu löschen wird das Select-Feld verwendet, um eindeutig Eines auszuwählen und dieses zu löschen.

Admin.php

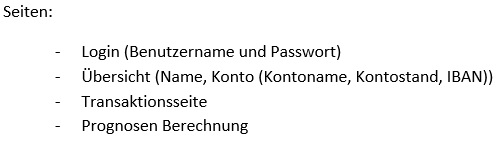
2.4. Umsetzung:

Alle Anfragen wurden so programmiert, dass die Eingabe, welche als HTML-Formular realisiert wurde, beim Betätigen der unterschiedlichen Button an das jeweilige PHP-Skript geschickt wird. Von dort werden diese Daten zuerst bearbeitet, dass Sie leichter weiterzuverarbeiten sind. Dazu gehört beispielsweise das Entfernen des €-Zeichens beim Anlegen einer neuen Überweisung, da der Datentyp in der Datenbank ein DECIMAL ist, also eine Zahl mit Nachkommastellen und eine Zahl mit einem Euro- oder Dollar-Zeichen als String behandelt werden müsste. Danach werden die benötigten Daten per SQL-Abfrage aus der Datenbank gelesen. Hier ist zum Beispiel die Anfrage zu sehen, die alle nötigen Daten ausliest, um die Transaktionstabelle auf der Überweisungsseite darzustellen. Danach folgt die Ausgabe, von der ein Ausschnitt zu sehen ist. Dabei ist es so, dass die Tabellenstruktur durch einfache Echo-Befehle erzeugt wird. Die Daten werden nach und nach an die zuerst erzeugte Kopf-Zeile angefügt. Dazu kommen noch einige Zwischenbearbeitungen, wie zum Beispiel die Korrektur der Zeit-Ausgabe. Dabei wird für jeden Datensatz das Ergebnis der Spalte „TIME“ in ein neues Layout gebracht, sodass die Anzeige in der Transaktionstabelle übersichtlich ist.

SQL-Abfrage zur Transaktionsanzeige

Ausgabe zur  
Transaktionsanzeige

**3. Architekturelle Entscheidungen:**

Ausgehend von der nebenstehenden ersten Anforderungsdefinition hat sich unser Projektentwurf entwickelt. Daher war klar, dass keine sehr komplexen Interaktionen notwendig werden. Daher konnten wir zum Ersten das Template relativ frei wählen. Wichtig war jedoch, dass es eine übersichtliche Bedienoberfläche bietet und eventuell schon Möglichkeiten der Zahlendarstellung beinhaltet. Das von uns ausgesuchte Template hat uns sowohl Tabellen geboten, wie die Datatable-Klasse, die mehrfach bei uns zum Einsatz kam, als auch eine einfache Graphen Implementierung ermöglichte, was wir vielleicht noch für die Prognose verwenden konnten. Bei der Datenbank haben wir uns für eine MySQL-Datenbank entschieden, da wir damit bereits erste Erfahrungen gesammelt haben und die Bedienung sehr leicht ist. Die letzte Entscheidung, die wir treffen mussten war während des Projektes und betraf unsere Lösung bezüglich PHP. Dabei war die Frage, ob es sich für uns lohnt ein Framework zu verwenden, um uns die Arbeit zu erleichtern. Dabei haben wir sowohl Laravel als auch Symfony in Erwägung gezogen und uns mit den ersten Schritten auseinandergesetzt. Währenddessen sind wir allerdings zu dem Schluss gekommen, dass für unsere Ansprüche noch kein Framework nötig ist und wir diese auch so realisieren können.

Erste Anforderungsdefinition