# Budget Duck – Konzept

Inhalt

[Budget Duck – Konzept 1](#_Toc112328041)

[Sinn der App 1](#_Toc112328042)

[Zielgruppe 1](#_Toc112328043)

[Anforderungen 1](#_Toc112328044)

[Mockups 2](#_Toc112328045)

[Funktionen der App 2](#_Toc112328046)

[Architektur 2](#_Toc112328047)

[Verwendete Technologien 2](#_Toc112328048)

[Diagramme 3](#_Toc112328049)

[UML 3](#_Toc112328050)

[2. Diagramm 3](#_Toc112328051)

[Testkonzept 3](#_Toc112328052)

## Sinn der App

Die App, welche ich Entwickeln möchte, ist ein Budget Verwaltungs-App. Auf die Idee bin ich gekommen, als ich ein paar Tage vor dem üK meine Mutter und Schwester nach Ideen gefragt hatte. Meine Schwester ist Studentin und meinte, dass sie es toll fände, wenn es ein App gäbe, mit welchem man sein Budget planen und verwalten kann. Da man als Student aber kein fixes Einkommen hat, muss das App sehr flexibel sein.

## Zielgruppe

Budget Verwaltungs-Apps werden in fast allen Alterskategorien genutzt. Mein App soll sich aber mehr an etwa 18 - 30-Jährige richten. Die Hauptsprache der App soll daher auch Englisch sein, aber ich möchte auch eine Funktion einbauen, um die Sprache zumindest auf Deutsch umstellen zu können.

## Anforderungen

Es sollen die funktionalen als auch nicht funktionalen Anforderungen erläutert werden

## Mockups

Für die einzelnen Ansichten der App müssen Mockups erstellt werden. Dazu sind sämtliche Tools erlaubt

## Funktionen der App

In meinem App hat man die Möglichkeit, sein Budget zu verwalten. Man kann Ausgaben und Einkommen notieren und verschiedenen Kategorien zuteilen. Man wird auch selbst Kategorien erstellen können. Geplant ist auch ein Feature, mit welchem man Einstellen kann, dass immer wieder kommende Zahlungen und/oder Einkommen, automatisch jeden Monat ab- oder zugerechnet werden. Man soll 3 verschiedene Konten / Budgets haben. Man kann diese Konten / Budgets nach eigenen belieben umbenennen. Man soll einen Verlauf der jeweils 50 Letzten Zahlungen und Einnahmen haben.

## Architektur

Ich werde MVC verwenden, weil es die einzige Architektur ist, die ich kenne und wir diese auch im Geschäft benutzten. MVC steht für Model View Controller. Das Model beinhaltet alle Daten, bzw. es weiss, wo sich diese befinden. Die View stellt alle Daten dar. Im Controller befindet sich die Logik und er bearbeitet die anstehenden Ereignisse. Durch diese Aufteilung werden spätere Änderungen und Erweiterungen vereinfacht.

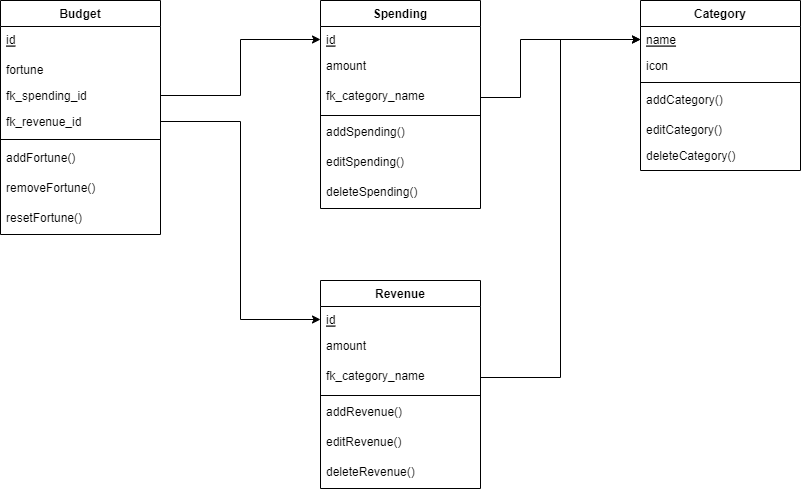
## Verwendete Technologien

Kotlin,

Die verwendeten Technologien (Programmiersprachen, Frameworks, etc.) sollen aufgezeigt werden. Ausserdem soll kurz erklärt werden, wieso genau diese eingesetzt werden.

## Diagramme

### UML



### 2. Diagramm

(Bild)

## Testkonzept

Im Testkonzept werden die Tests für die fertige App definiert. Darin muss jeder Anwendungsfall mit einem aussagekräftigen Testfall abgedeckt werden. Für jeden Testfall müssen folgende Dinge angegeben werden: · Identifikation · Vorbedingungen · Vorgehen beim Test (Schritt für Schritt) · Erwartetes Resultat