**Application Concept Document (ACD) für das Film-Matching-System**

Inhalt

[Einleitung 2](#_Toc187951072)

[Ziele der Anwendung 2](#_Toc187951073)

[Systemübersicht 3](#_Toc187951074)

[Wichtige Features: 3](#_Toc187951075)

[Technologie 3](#_Toc187951076)

[Datenbankmodellierung 4](#_Toc187951077)

[Tabellenstruktur 4](#_Toc187951078)

[API Design 5](#_Toc187951079)

[Login 5](#_Toc187951080)

[Filmauswahl abrufen 5](#_Toc187951081)

[Filme auswählen 5](#_Toc187951082)

[Matching 5](#_Toc187951083)

# Einleitung

Das Ziel dieses Projekts ist es, eine Anwendung zu entwickeln, bei der zwei Nutzer Filme auswählen und das System die Übereinstimmungen zwischen ihren Auswahlentscheidungen berechnet. Die Anwendung soll aus einer zentralen Datenbank Filme abrufen, die verschiedenen Kategorien zugeordnet sind, und die Möglichkeit bieten, dass die Nutzer ihre bevorzugten Filme auswählen.

# Ziele der Anwendung

Das Ziel dieser Anwendung ist die Bereitstellung einer Plattform, auf der Nutzer gemeinsam mit Ihrem Partner Filme aus einer Datenbank wählen können. Das Ziel hierbei ist es zu bestimmen, auf welchen Film beide Benutzer Lust haben. Im Rahmen der Arbeit im Fach Entwicklung verteilter Systeme wird zusätzlich eine API, die die Kommunikation zwischen Frontend und Backend ermöglicht als Ziel der Anwendung gesetzt.

# Systemübersicht

Die Anwendung besteht aus drei Hauptkomponenten:

* **Frontend:** Die Benutzeroberfläche, die es den Nutzern ermöglicht, sich einzuloggen und Filme auszuwählen.
* **Backend:** Die API, die die Logik der Filmauswahl, das Matching und die Kommunikation mit der Datenbank übernimmt.
* **Datenbank:** Eine relationale Datenbank zur Speicherung von Filmen, Nutzerdaten und Auswahlen.

## Wichtige Features:

Benutzerregistrierung und Login (Authentifizierung)

Anzeige von Filmen aus einer vordefinierten Auswahl

Möglichkeit, Filme auszuwählen, diese zu speichern

Matching-Algorithmus, der übereinstimmende Filme zwischen den beiden Nutzern findet.

# Technologie

**Frontend:** React.js (für eine dynamische, benutzerfreundliche Oberfläche).

**Backend:** Python mit Flask (leichtgewichtig und gut geeignet für APIs).

**Datenbank:** PostgreSQL (für strukturierte Daten und komplexe Abfragen).

**Authentifizierung:** JWT (JSON Web Tokens) für die sichere Benutzer-Authentifizierung.

# Datenbankmodellierung

## Tabellenstruktur

**User:**

id

username

passwort

**Movies:**

id

title

category

**selections:**

id

user\_id

movie\_id

# API Design

## Login

Endpoint

Input

Output

## Filmauswahl abrufen

Entpoint

Output: (Die 10 Filme)

## Filme auswählen

Endpoint

Input: (Auswahl der übereinstimmenden Filme)

## Matching

Endpoint:

Input:

Output:

# Systemarchitektur

## Frontend

## Backend

## Datenbank

# Matching-Algorithmus

Hier sollten wir erklären, wie wir den Algorithmus umgesetzt haben.