Prototyping Projektdokumentation

Name: Leon Wink

E-Mail: winkleo1@students.zhaw.ch

URL der deployten Anwendung: https://683b27c851d080e52dd68b51--tangerine-rabanadas-39ff7f.netlify.app/

# Einleitung

Project Manager ist eine Webanwendung zur Verwaltung von Projekten und ihren zugehörigen Aufgaben (Tasks). Benutzer können Projekte mit Titel und Beschreibung anlegen, und innerhalb jedes Projekts beliebig viele Aufgaben erfassen. Jede Aufgabe enthält Informationen wie Beschreibung, Impact-Level, Fälligkeitsdatum und Status (offen oder erledigt). Die Anwendung ist mit SvelteKit und Bootstrap umgesetzt, nutzt MongoDB zur Datenspeicherung und implementiert serverseitige Logik über load()- und action()-Funktionen.

# Datenmodell

*Ein Bild, das Text, Screenshot, Diagramm, Schrift enthält.

KI-generierte Inhalte können fehlerhaft sein.*

# Beschreibung der Anwendung

## Landingpage

Route: /

Ein Bild, das Text, Screenshot, Software, Betriebssystem enthält.

KI-generierte Inhalte können fehlerhaft sein.

Das ist die Landingpage mit einem kurzen Willkommensvideo. Mit klick auf den Button «Start Now» wird man zur Projekt-übersicht geleitet. Über die Navigation kann direkt auf /projects/create navigiert werden, um ein Projekt zu erfassen. Ebenfalls kann via dem Menüpunkt «Tasks» direkt auf die Übersicht aller Tasks navigiert werden.

## Projects

Route: /projects

Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

KI-generierte Inhalte können fehlerhaft sein.

Auf dieser Seite werden alle aktuellen Projekte mit ihren dazugehörigen Tasks dargestellt. Ebenfalls bietet sich der Button «Add new project» an, um ein neues Projekt zu erfassen. Ebenfalls ist es möglich mit Klick auf den Projekttitel, auf die Detailseite des entsprechenden Projektes zu gelangen. Ebenfalls ist es möglich einer der erfassten Task anzuklicken, um direkt auf die Detailseite des Tasks zu gelangen.

## Create Project

Route: /projects/create

Ein Bild, das Screenshot, Text, Software, Display enthält.

KI-generierte Inhalte können fehlerhaft sein.

Bei klick auf «Add new Project» wird die Maske für das Erfassen eines neuen Projekts geöffnet. Hier kann mit der Eingabe der entsprechenden Felder ein neues Projekt angelegt werden.

## Project Detailseite

Route: /projects/[id]

Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

KI-generierte Inhalte können fehlerhaft sein.

Die Detailseite zeigt eine weiter Informationen auf dem Projekt wie aber auch die auf dem entsprechendem Projekt erfassten Tasks. Mit dem Button «Add task» kann auf das Projekt ein neuer Task eröffnet werden. Mit dem Button «Delete project» kann das Projekt gelöst werden. Ebenfalls ist es möglich, mit Klick auf den Tasktitel auf die Detailseite des Tasks zu gelangen.

## Add Task

Route: /tasks/create?projectId=[project\_id]

Ein Bild, das Screenshot, Text, Software, Computersymbol enthält.

KI-generierte Inhalte können fehlerhaft sein.

Auf dieser Seite kann ein neuer Task erfasst und in die DB gespeichert werden. Dieser Task bekommt die Projekt ID des entsprechenden Projekts zugeordnet. Dies passiert via Weitergabe der ID in der URL.

## Task Detailseite

Route: /tasks/[task\_id]

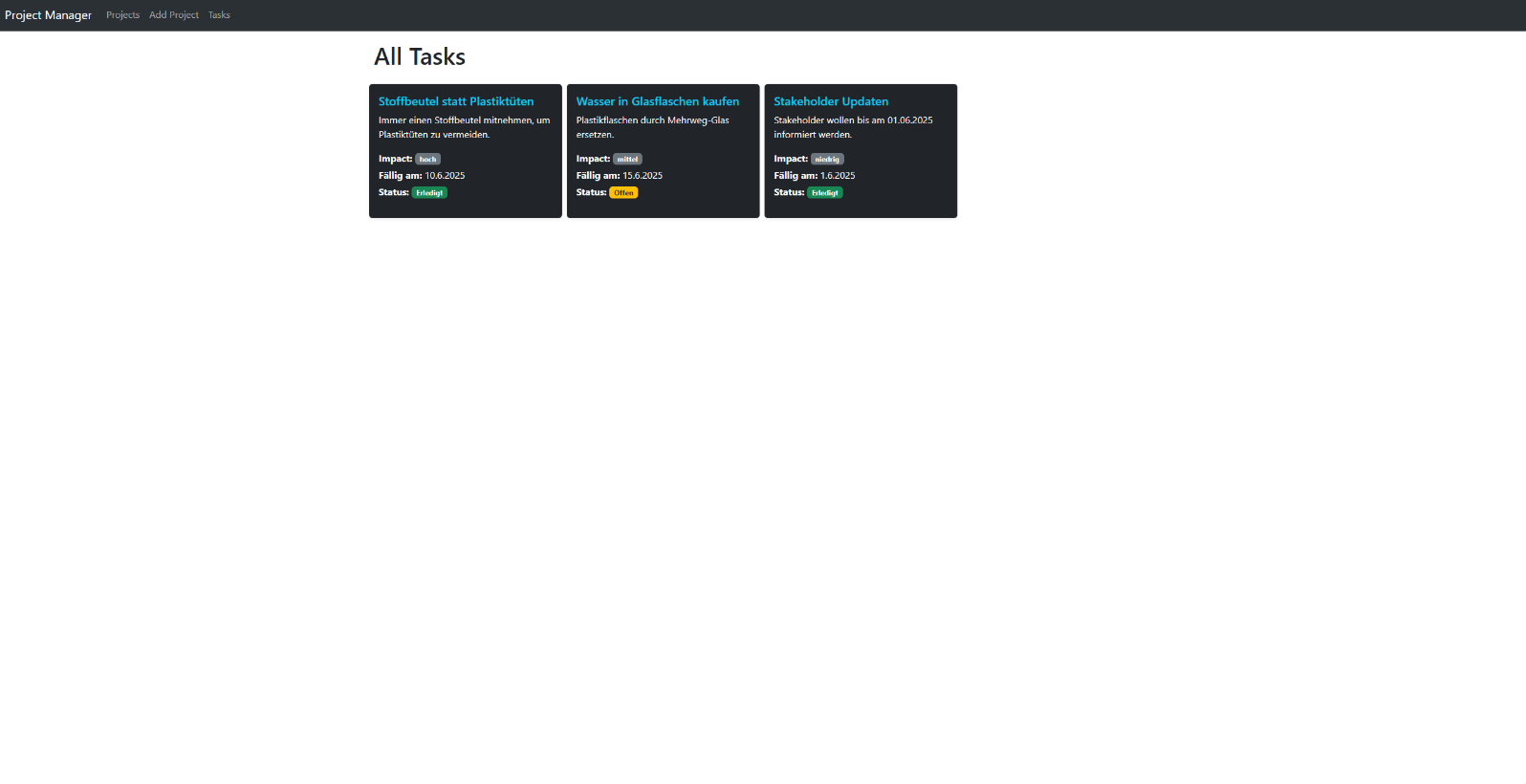
Ein Bild, das Text, Screenshot, Software, Computer enthält.

KI-generierte Inhalte können fehlerhaft sein.

Die Detailseite eines Tasks gibt auskunft über die entsprechenden Daten. Mit dem Button «Edit Task» kann der Task bearbeitet werden. Es öffnet erneut dieselbe Maske wie das Create Task. Mit dem Button «Delete Task» kann der Task gelöst werden.

## All Tasks

Route: /Tasks



Auf dieser Seite sind alle erfassten Tasks gelistet. Ebenfalls gibt es wieder die Option mit Klick auf den Tasktitel, auf die jeweilige Detailseite zu gelangen.

## Dateistruktur

Die Dateistruktur ist durchgängig und kongruent gehalten. Entsprechend der Routes sind die Files auch angeordnet.

1. lib

* components

ProjectCard.svelte

TaskCard.svelte

* db.js
* index.js

1. routes

* +layout.svelte
* +page.svelte
* projects

+page.server.js

+page.svelte

[project\_id]

+page.server.js

+page.svelte

create

+page.server.js

+page.svelte

* styles.css
* tasks

+page.server.js

+page.svelte

[task\_id]

+page.server.js

+page.svelte

edit

+page.server.js

+page.svelte

create

+page.server.js

+page.svelte

# Erweiterungen

## Entitäts-Verknüpfung

Es können Tasks zu dazugehören Projects erfasst werden. Dies wird beim erstellen des Task mit der Übergabe der Project\_Id via URL bewerkstelligt.

Dateien:

* src\routes\tasks\create\+page.server.js
* src\routes\tasks\create\+page.svelte

## Umfang

Es wurden insgesammt acht +page.svelte Seiten erstellt wie auch zwei Templates jeweils für die TaskCard und die ProjectCard

Dateien:

* src\routes\projects\+page.svelte
* src\routes\tasks\[task\_id]\edit\+page.svelte
* src\routes\tasks\[task\_id]\+page.svelte
* src\routes\tasks\create\+page.svelte
* src\routes\tasks\+page.svelte
* src\routes\+page.svelte
* src\routes\projects\create\+page.svelte
* src\routes\projects\[project\_id]\+page.svelte
* src\lib\components\ProjectCard.svelte
* src\lib\components\TaskCard.svelte