|  |  |
| --- | --- |
|  |  |
| **M151 Doku**  **ShortURL** |  |
|  |  |
|  | *18.6.2022*  *M151* |
|  | *Ramon, Sven M.* |

Contents

[Einleitung 3](#_Toc106460454)

[Zieldefinition 4](#_Toc106460455)

[Meilensteine 5](#_Toc106460456)

[Benutzung von Frameworks 6](#_Toc106460457)

[Arbeitsjournal 7](#_Toc106460458)

## Einleitung

Wir haben im Modul 151 den Auftrag bekommen, eine Webapplikation zu realisieren. Diese Sollte mindestens eine 4-Tier Architektur besitzen, eine verschlüsselte Verbindung besitzen und speziell auf Datensicherheit achten. Dieser Auftrag wird in einer Gruppe erledigt.

Nach dem wir uns organisiert haben, besprachen wir, was für eine Webseite wir machen sollten. Wir hatten zuerst eher weniger Ideen, aber schlussendlich einigten wir uns auf einen URL-Shortener Service. Das ist eine Website, mit der man lange URLs (z.B. <https://dies-ist-eine-sehr-lange-domain.com/noch-mehr-text-hier>) kürzen kann, so dass sie nur noch ein Bruchteil ihrer Originallänge waren. So eine gekürzte URL würde etwa so aussehen: <https://short.cc/AHftl9>

## Zieldefinition

Damit wir uns immer schön Orientieren können während dem Entwickeln setzen wir uns vor dem Beginn ein paar Ziele, welche wir mit unserer Website erreichen möchten. Somit wissen wir immer klar, woran noch gearbeitet werden muss.

#### Ziel 1

Die Webseite soll eingegebene URLs kürzen und dem User die gekürzte URL zurückgeben.

#### Ziel 2

Es sollte keine Short-URL zwei Mal existieren.

#### Ziel 3

Es darf keine SQL-Injection, XSS oder Request-Forgery möglich sein.

## Meilensteine

Anhand der Meilensteine können wir unseren Fortschritt messen. Da wir in Etappen arbeiten, wäre es gut, wenn wir jede Woche einen Meilenstein erreichen können. Meilenstein 1 steht hierbei für die 1. Woche, Meilenstein 2 für die 2. und so weiter.

#### Meilenstein 1

Die Planung des Projektes ist abgeschlossen

#### Meilenstein 2

Das Frontend steht.

#### Meilenstein 3

Die Datenbank ist fertig gestellt und es sind Funktionen im Backend, die mit der DB interagieren.

#### Meilenstein 4

Das Backend ist fertig.

#### Meilenstein 5

Die Dokumentation ist abgeschlossen und somit ist das Projekt fertig.

## Benutzung von Frameworks

Wir haben für unser Programm PHP als Back-end verwendet, da es in JavaScript (ohne ein Framework) unmöglich gewesen wäre, die Serversoftware umzusetzen. Ansonsten wurde kein anderes Framework verwendet. Wir haben uns für PHP entschieden, da wir bereits damit vertraut waren und es auch einfach zum Hosten ist.

## Arbeitsjournal

#### Woche

##### Ramon & Sven

In der ersten Woche haben wir zusammen das Projekt definiert. Wir einigten uns auf einen URL-Shortener, da es einfach zum Entwickeln ist, man aber dennoch auf gewisse Sicherheitsmassnahmen achten muss. Ausserdem haben wir in der 1. Woche noch die Grobplanung gemacht, also festgelegt welche Meilensteine wir in jeder Woche erreichen wollen.

#### Woche

##### Ramon

Das Frontend samt design habe ich in dieser Woche gemacht. Allerdings noch ohne JavaScript, das kommt später dazu, denn dafür muss zuerst das Backend stehen.

##### Sven

bla

#### Woche

##### Ramon

In dieser Woche habe ich die Datenbank entworfen und realisiert. Die gekürzten URLs sollten schliesslich gespeichert werden. Ausserdem habe ich noch ein hCaptcha hinzugefügt auf dem Frontend, da so eine Seite bestimmt Bots anlocken wird und man sich mit einem Captcha, welches zum Teil nicht mal Menschen können, einfach schützen kann.

##### Sven

Bli

#### Woche

##### R**amon & Sven**

Es war schulfrei, dementsprechend wurde an diesem Tag eher weniger erledigt. Das war auch so eingeplant und sollte im Zeitplan aufgehen, da wir sehr gut mit dabei sind.

#### Woche

##### Ramon

In der 5. Woche habe ich das Backend fast fertig gemacht. Es steht noch nicht ganz aber das sollte nächstes Mal fertig werden. Ausserdem habe ich das JS beim Frontend hinzugefügt, mit welchem man über jQuery und Ajax mit dem Backend interagiert.

## Vermeiden von Angriffen

Eine Website ist ein beliebtes ziel von Hackern und anderweitigen Angreiffern, daher muss der Entwickler sich darum kümmern, dass die Sicherheit und Integrität der Systeme und Besucher Daten gewahrt bleibt. Unter diesem Kapitel beschreiben wir kurz, was für Sicherheitsmassnahmen wir getroffen haben, um typische Angriffe zu verhindern.

#### SQL-Injection

Jeder Input vom User, der in irgendeinem Weg Kontakt mit der Datenbank hat, muss «escaped» werden. Das bedeutet, dass Spezielle im Zeichen wie ‘ und ` modifiziert werden, damit sie nicht von der Datenbanksoftware als SQL Syntax erkennt werden können.

#### XSS

Cross-Site Scripting, kurz XSS, ist eine Vulnerability, mit der man auf dem Webserver JavaScript speichern kann. Wenn danach ein anderer User die Webseite aufruft, wird dieses JS Script ausgeführt und kann potenziell die Login Daten stehlen oder im Hintergrund Crypto Minen.   
Das kann man auch relativ einfach verhindern, in dem Jeder Userinput, der auf der Webseite geladen/gerendert werden könnte, durch die Funktion *htmlspecialchars(«input»)* geschickt wird.