Herausgeber Prof. Dr. Barbara Dörsam

Schriftreihe Bachelor-Resümee

Forschungsbereich **Softwareentwicklung / JavaScript**

Überprüfung von Web-Auftritten

Konzeption und Realisierung einer prototypischen Anwendung zur automatisierten Evaluierung von HTML- und CSS-basiertem Quellcode

Roland Gassen

Impressum

Hochschule der Medien

Nobelstrasse 10

70569 Stuttgart

www.hdm-stuttgart.de

0711 8923-0

Autor

Roland Gassen

Betreuer

Prof. Dr. Barbara Dörsam

Datum

24.08.2022

Wirtschaftsingenieurwesen Medien

www.hdm-stuttgart.de/wing

hitzges@hdm-stuttgart.de

0711/8923-2634

Layout

Jochen Riegg

Fotos und Illustrationen

Innenteil: Roland Gassen

Bachelor-Resümee

Überprüfung von Web-Auftritten

Konzeption und Realisierung einer prototypischen Anwendung zur automatisierten Evaluierung von HTML- und CSS-basiertem Quellcode

Roland Gassen

24.08.2022

Der Autor

Roland Gassen absolvierte 2015 seine Ausbildung zum Mediengestalter für Digital- und Printmedien. Nach dem Abitur 2017, das er unter anderem mit dem Informatik-Preis der Wirtschaftsoberschule Stuttgart an eben dieser abschloss, studierte er Druck- und Medientechnologie mit dem Schwerpunkt Digital-Publishing an der HdM Stuttgart.

Inhaltsverzeichnis

1. Problemstellung und Zielsetzung

Im Rahmen der Vorlesung „Web-Technologien“ an der Hochschule der Medien werden Studierende des Studiengangs „Wirtschaftsingenieurwesen Medien“ in die Entwicklung und Gestaltung von Webauftritten eingeführt. Dabei ist die Prüfungsleistung der Vorlesung die Abgabe eines Webauftritts, der je mindestens 5 HTML- und die dazugehörigen CSS-Dateien enthalten muss. Diese Webauftritte werden dann zur Bewertung anhand von über 40 Kriterien überprüft. Da in dieser Vorlesung bis zu über hundert Studierende eingeschrieben sind, entsteht hier ein sehr großer Zeitaufwand.

Ziel dieser Bachelorarbeit ist es, eine prototypische Anwendung zu entwickeln, die möglichst viele dieser Kriterien automatisiert überprüfen kann.

Um dieses Ziel umzusetzen ist folgende Frage zu klären:

- Mit welchen Methoden können HTML- und CSS-Dateien sowohl nach syntaktischen als auch inhaltlichen Kriterien automatisiert überprüft werden?

2. Anforderungen

An die prototypische Anwendung wurden folgende Kriterien gestellt:

- Alle überprüften Kriterien sollen mit einer **Bewertung** (1, 0.5 oder 0 Punkte) bewertet werden. Wird ein Kriterium nicht vollständig eingehalten, soll ein **Kommentar** den Punktabzug erklären.
- Die Anwendung soll **konfigurierbar** sein. Es soll möglich sein, einzelne Faktoren der Kriterien, wie beispielsweise die Menge der erforderlichen HTML-Dateien eines Webauftritts, mit Hilfe einer Konfigurationsdatei anzupassen
- Die Anwendung soll **erweiterbar** sein. Es soll möglich sein, die Anwendung um weitere Funktionen zu erweitern, ohne dass in den bestehenden Code eingegriffen werden muss.
- Die Anwendung soll insgesamt 22 Kriterien möglichst vollständig bewerten. In Tabelle 1 sind beispielhaft einige der Kriterien aufgelistet:

Nr.	Titel	Beschreibung
1	CSS- und HTML-Dateien sind valide	Alle HTML- und CSS-Dateien des Webauftritts müssen nach den Kriterien des W3C als „valide“ eingestuft werden.
2	Verzeichnisstruktur	Die Verzeichnisstruktur der Webauftritte muss den Anforderungen aus der Aufgabenstellung entsprechen.
3	Mindestens ein h2-Element	Im gesamten Webauftritt muss mindestens ein h2-Element verwendet werden.
4	Mobile first	Alle CSS-Dateien sollen mit dem „Mobile First“-Prinzip geschrieben sein.

Tabelle 1: Anforderungen an die prototypische Anwendung

3. Grundlagen

Grundsätzlich wird das automatisierte Überprüfen von Software eingesetzt, um die Qualität eines Codes und dessen Konformität zu Standards oder individuellen Programmierrichtlinien zu beurteilen (Dorninger und Ziebermayr 2021, S. 315). In dieser Bachelorarbeit werden Werkzeuge zur statischen Codeanalyse verwendet. Diese kontrollieren den zu überprüfenden Code - hier die Webauftritte - nach festgelegten Kriterien, ohne diesen auszuführen. Das heißt, alle Analysen können prinzipiell ohne Computerunterstützung durchgeführt werden (Liggesmeyer 2009, S. 57).

Zum Überprüfen von HTML- und CSS-Code werden im Wesentlichen zwei Werkzeugtypen verwendet:

- **Validatoren**: Hierbei handelt es sich um Programme, die die Gültigkeit, also die syntaktische Korrektheit eines Codefragments oder Dokuments überprüfen.
- **Linting-Systeme**: Dies sind Programme, die Code-Abschnitte nach vorher definierten Regeln oder Kriterien überprüfen können.

Linting-Systeme wie beispielsweise „stylelint“ verwenden eine **Konfigurationsdatei**, in der Einstellungen festgehalten werden können, die das Verhalten der Anwendung beeinflussen. Beispielsweise können alle Überprüfungen ein- oder ausgeschaltet werden. Außerdem besteht das Programm selbst neben einem **Controller**, der sich um den Ablauf und die interne Logik kümmert, aus je einem **Testskript** pro Kriterium, das überprüft werden soll (stylelint.io 2022) und (Phabricator 2022). Ein sehr ähnlicher Aufbau wurde in der prototypischen Anwendung dieser Bachelorarbeit verwendet.

4. Umsetzung

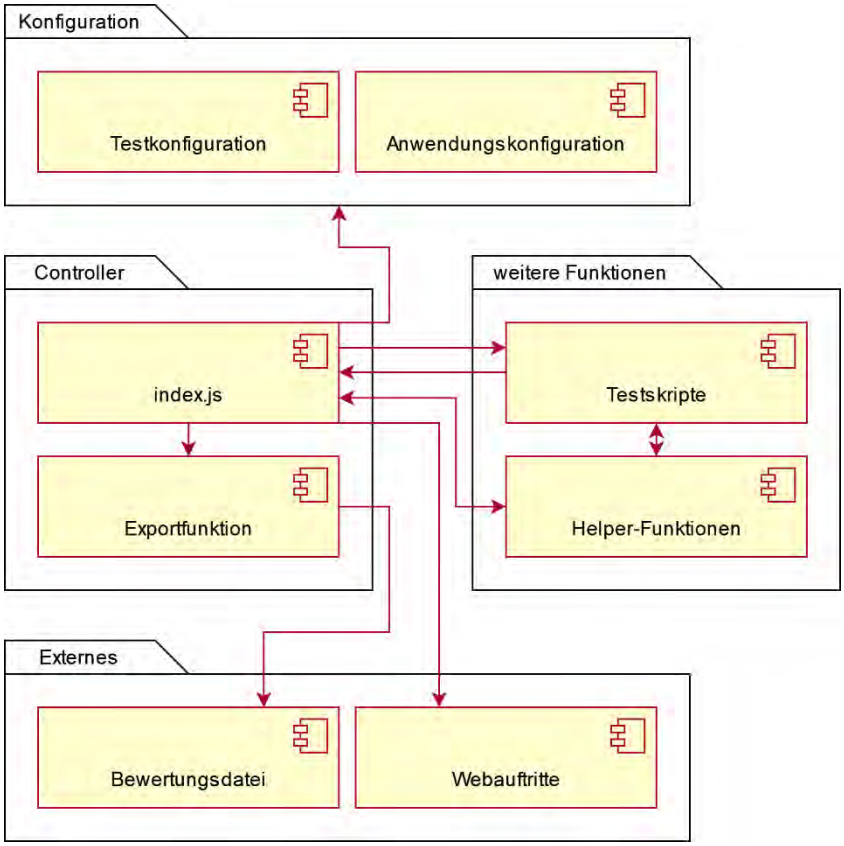

Abbildung 1: Architekturentwurf des Prototyps

Die Architektur des Prototyps ist, wie in Abbildung 1 dargestellt, an die der oben erwähnten Linting-Systeme angelehnt. Im Wesentlichen besteht sie aus vier Elementen:

1. **Controller**: Als Kern der Anwendung dient ein Controller, in dem die interne Logik abgewickelt wird. Der Controller liest nacheinander alle benötigten Dateien ein: die Konfiguration, die Testskripte zum Überprüfen der Kriterien und die Webauftritte, die überprüft werden sollen. Anschließend werden die Testskripte ausgeführt und die Bewertungen, die diese zurückgeben, gesammelt. Am Ende eines Programmdurchlaufs werden diese dann in einer CSV-Datei auf dem Betriebssystem gespeichert.
2. **Konfigurationsdateien**: Die Konfigurationsdateien enthalten Einstellungen, die Einfluss auf den Ablauf des Prototyps haben. Dabei werden sowohl Einstellungen für den Controller, als auch für die Testskripte eingelesen. Die Konfiguration ist so konzipiert, dass eine einfache Erweiterung um weitere Kriterien möglich ist, ohne in den Code des Controllers eingreifen zu müssen.

Abbildung 2: Konfigurationsbeispiel des Tests zur Überprüfung der Validität von CSS-Dateien.

Wie im Beispiel in Abbildung 2 in der dritten Zeile zu sehen, könnten beispielsweise die Anzahl der Punkte, die pro Fehler abgezogen werden in der Konfiguration eingestellt werden.

3. **Weitere Hilfsfunktionen**: Unter den Hilfsfunktionen sind Funktionen zusammengefasst, die kleinere Aufgaben übernehmen. Diese können sowohl von dem Controller als auch in den einzelnen Testskripten verwendet werden.
4. **Externes**: Als vierter Bereich können die externen Dateien gesehen werden. Dazu gehören zum einen die Dateien, die von der Anwendung überprüft werden sollen und zum anderen die auf dem System gespeicherte Bewertungsdatei, die von dem Prototyp erstellt werden kann. Technisch gesehen sind die externen Dateien kein Teil der Anwendung selbst.

5. Evaluierung

Von den eingangs gestellten Grundanforderungen konnten alle erfüllt werden:

- Die prototypische Anwendung ist in der Lage, die Kriterien, die automatisiert überprüft werden können, zu bewerten und kommentieren.
- Der Prototyp ist durch die verwendete Architektur bei sich ändernden Kriterien einfach an diese Änderungen anzupassen.
- Die einfache Erweiterbarkeit des Prototyps ist gewährleistet, da die Anwendung keine Änderung an bestehendem Code benötigt, um sinnvoll erweitert zu werden.

Von den 22 Kriterien, die zur automatisierten Bewertung umgesetzt wurden, sind 18 ohne weiteres so verwendbar. Die restlichen vier brauchen nach aktuellem Stand stets eine manuelle Nachbearbeitung.

Wie in Abbildung 3 dargestellt sind die Webauftritte durch die manuelle Überprüfung in den meisten Fällen wesentlich besser bewertet als durch die automatisierte Überprüfung. Dies liegt hauptsächlich an vier Gründen:

- Wiederholungsfehler, also Fehler, die mehrfach von der automatisierten Überprüfung gefunden werden, führen zu mehreren Punktabzügen.
- Einzelne der Überprüfungsfunktionen wiesen zwischenzeitlich Technik- oder Logik-Fehler auf, die ebenfalls zu Punktabzug führten.
- Anders als bei einer menschlichen Überprüfung kann die Automatisierte keine Fehler übersehen oder als nicht gravierend einordnen. Die Bewertung erfolgt stets mit voller Strenge.
- Außerdem ist zu beachten, dass die manuelle Überprüfung eine Maximalpunktzahl von 23 Punkten ermöglicht, die Automatisierte nur 21 Punkten. Der Unterschied ist in der Realität also etwas kleiner als die Grafik vermuten lässt.

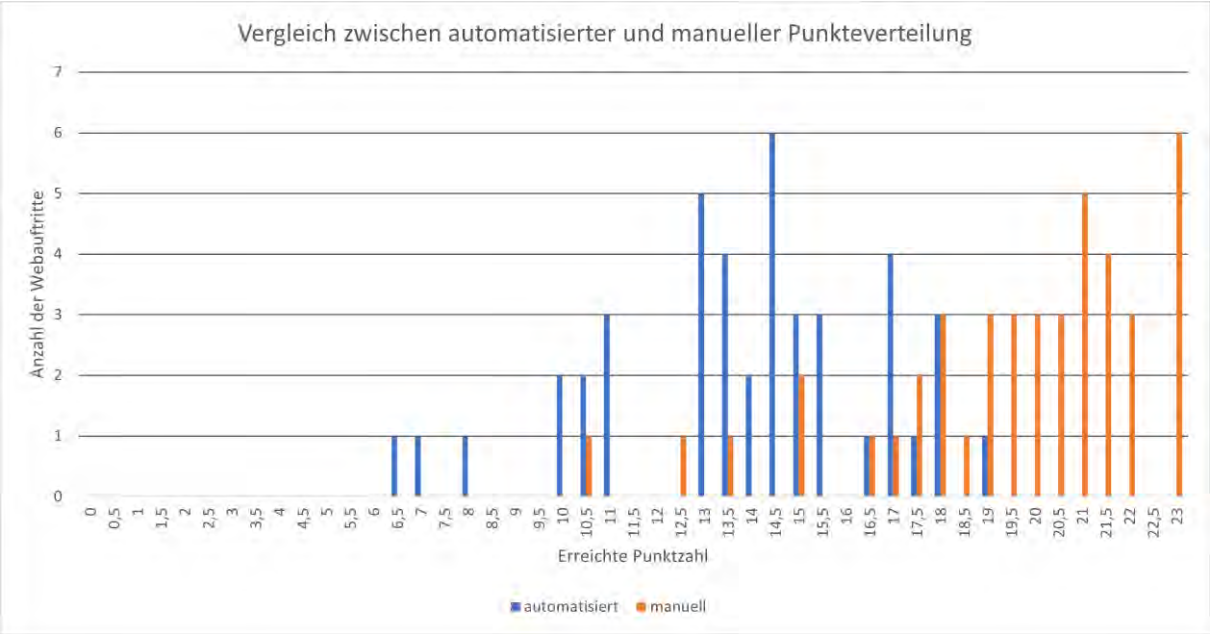

Abbildung 3: Vergleich zwischen manueller und automatisierter Punkteverteilung

6. Fazit / Ausblick

Ziel der Arbeit war es, zu untersuchen, ob mithilfe eines Programms einfache Webauftritte nach einer Reihe von definierten Kriterien überprüft und bewertet werden können. Zu diesem Zweck wurde ein Prototyp entwickelt, der mithilfe von statischen Analyseverfahren HTML- und CSS-Code nach selbst definierten Kriterien überprüft.

Durch die Umsetzung des Prototyps konnte aufgezeigt werden, dass eine Unterstützung der manuellen Bewertung durch diesen Prototyp sinnvoll und zeitersparend sein kann. Die automatisierte Bewertung kann überdies dazu beitragen die manuelle Bewertung langfristig zu verbessern, da auf Bewertungsfehler oder Ungenauigkeiten hingewiesen werden kann.

Durch die konsequente Umsetzung einer konfigurier- und erweiterbaren Software ist der Prototyp auch bei sich ändernden Kriterien weiterhin einsetzbar.

Dennoch ist durch die Analyse der Bewertungen ebenfalls klar geworden, dass es zwischen der manuellen und der automatisierten Bewertung für einzelne Kriterien zu großen Unterschieden kommen kann. Es wurde deutlich sichtbar, an welchen Stellen eine manuelle Überprüfung derzeit noch unabdingbar ist.

Insgesamt bietet der Protoptyp schon in der ersten Version aufgrund der dadurch entstehenden Zeitersparnis ein praktisches Hilfsmittel, dass zur objektiven Überprüfung der Webauftritte der Studierenden einen sinnvollen Beitrag leisten kann.

7. Referenzen

Dorninger, Bernhard; Ziebermayr, Thomas (2021): Software-Qualitätssicherung im Maschinen- und Anlagenbau: automatisierte Bewertung der technischen Qualität von SPS-Code. In: *Elektrotech. Inftech.* 138 (6), S. 315–320. DOI: 10.1007/s00502-021-00917-x.

Liggesmeyer, Peter (2009): Software-Qualität. Testen, Analysieren und Verifizieren von Software. 2. Auflage. Heidelberg: Spektrum Akademischer Verlag.

Phabricator (2022): Arcanist User Guide: Lint. Online verfügbar unter https://secure.phabricator.com/book/phabricator/article/arcanist_lint/, zuletzt geprüft am 02.04.2022.

stylelint.io (2022): StyleLint Configuration. Online verfügbar unter https://github.com/stylelint/stylelint/blob/main/docs/user-guide/configure.md, zuletzt aktualisiert am 04.02.2022.