

LLCE

Dokumentation zur Projektarbeit

Projekt „LLCE“

**Lern App zur Vorbereitung auf LPIC-1 Prüfungen**

Abgabedatum: Dortmund, den 06.04.2023

**Ersteller der Dokumentation:**

Selim Sadik Tülümen

**Inhaltsverzeichnis**

Inhaltsverzeichnis

[Abbildungsverzeichnis IV](#bookmark49" \t "Current Document)

[Verzeichnis der Listings V](#bookmark53" \t "Current Document)

[Tabellenverzeichnis VI](#bookmark57" \t "Current Document)

[Abkürzungsverzeichnis VII](#bookmark61" \t "Current Document)

1. [Einleitung 1](#bookmark65" \t "Current Document)
   1. [Projektumfeld 1](#bookmark70" \t "Current Document)
   2. [Projektziel 1](#bookmark75" \t "Current Document)
   3. [Projektbegründung 1](#bookmark80" \t "Current Document)
   4. [Projektschnittstellen 1](#bookmark85" \t "Current Document)
   5. [Zielgruppen 2](#bookmark90" \t "Current Document)
   6. [Projektabgrenzung 2](#bookmark95" \t "Current Document)
2. [Projektplanung 2](#bookmark101" \t "Current Document)
   1. [Projektphasen 2](#bookmark105" \t "Current Document)
   2. [Ressourcenplanung 3](#bookmark111" \t "Current Document)
   3. [Entwicklungsprozess 3](#bookmark116" \t "Current Document)
3. [Analysephase 3](#bookmark122" \t "Current Document)
   1. [Ist-Analyse 3](#bookmark126" \t "Current Document)
   2. [Anwendungsfälle 3](#bookmark131" \t "Current Document)
   3. [Qualitätsanforderungen 4](#bookmark137" \t "Current Document)
   4. [Lastenheft 4](#bookmark142" \t "Current Document)
4. [Entwurfsphase 5](#bookmark147" \t "Current Document)
   1. [Zielplattform 5](#bookmark152" \t "Current Document)
   2. [Architekturdesign 5](#bookmark157" \t "Current Document)
   3. [Entwurf der Benutzeroberfläche 5](#bookmark162" \t "Current Document)
   4. [Datenmodell 6](#bookmark167" \t "Current Document)
   5. [Maßnahmen zur Qualitätssicherung 6](#bookmark172" \t "Current Document)
   6. [Pflichtenheft 6](#bookmark177" \t "Current Document)
5. [Implementierungsphase 7](#bookmark183" \t "Current Document)
   1. [Implementierung der Datenstrukturen 7](#bookmark187" \t "Current Document)
   2. [Implementierung der Benutzeroberfläche 7](#bookmark192" \t "Current Document)
   3. [Implementierung der Geschäftslogik 7](#bookmark197" \t "Current Document)
6. [Benutzerdokumentation 8](#bookmark202" \t "Current Document)
7. [Fazit 8](#bookmark207" \t "Current Document)
   1. [Soll-/Ist-Vergleich 8](#bookmark212" \t "Current Document)
   2. [Lessons Learned 8](#bookmark216" \t "Current Document)
   3. [Ausblick 9](#bookmark221" \t "Current Document)

[Literaturverzeichnis 10](#bookmark233" \t "Current Document)

[Anhang i](#bookmark237" \t "Current Document)

[A1 Use-Case-Diagramm i](#bookmark241" \t "Current Document)

[A2 Lastenheft (Auszug) ii](#bookmark246" \t "Current Document)

1. Zweck und Ziel dieses Dokuments ii
2. Ziel des Projektes ii
3. App Laufzeitumgebung ii

[A3 Pflichtenheft (Auszug) iii](#bookmark255" \t "Current Document)

* 1. [Musskriterien iii](#bookmark258" \t "Current Document)
  2. [Wunschkriterien iii](#bookmark266" \t "Current Document)

[A4 Ereignisgesteuerte Prozesskette Hauptnavigation iv](#bookmark284" \t "Current Document)

[A5 Ereignisgesteuerte Prozesskette Einzelfrage v](#bookmark289" \t "Current Document)

[A6 Screenshot der Startseite vi](#bookmark293" \t "Current Document)

[A7 Screenshot Startseite Dropdown Katalogauswahl vi](#bookmark298" \t "Current Document)

[A8 Screenshot Fragenliste vii](#bookmark305" \t "Current Document)

[A9 Screenshot Einzelfrage (Fill-Inn-Frage) vii](#bookmark310" \t "Current Document)

[A10 Screenshot Einzelfrage (Multiple-Choice-Frage) viii](#bookmark314" \t "Current Document)

[A11 Screenshot Einzelfrage (Single-Choice-Frage) viii](#bookmark319" \t "Current Document)

[A12 Screenshot Einzelfrage (Single-Choice-Frage) ix](#bookmark323" \t "Current Document)

[A13 Screenshot Einzelfrage (Fill-In-Frage) ix](#bookmark328" \t "Current Document)

[A14 Haupt-Ordnerstruktur Angular x](#bookmark332" \t "Current Document)

[A15 Listing app.module.ts xi](#bookmark337" \t "Current Document)

[A16 app-routing.module.ts xii](#bookmark343" \t "Current Document)

[A17 app.component.ts xiii](#bookmark348" \t "Current Document)

[A18 app.component.html xiii](#bookmark353" \t "Current Document)

[A19 home.component.ts xiv](#bookmark357" \t "Current Document)

[A20 home.component.html xv](#bookmark368" \t "Current Document)

[A21 home.component.css xv](#bookmark373" \t "Current Document)

[A22 check-all.component.ts xvi](#bookmark378" \t "Current Document)

[A23 check-all.component.html xvii](#bookmark388" \t "Current Document)

[A24 check-all.component.css xviii](#bookmark393" \t "Current Document)

[A25 learn-mc-list.component.html xix](#bookmark404" \t "Current Document)

[A26 learn-mc-list.component.ts xx](#bookmark409" \t "Current Document)

[A27 learn-mc-list.component.css xxi](#bookmark420" \t "Current Document)

[A28 exam-mc.component.ts xxii](#bookmark425" \t "Current Document)

[A29 exam-mc.component.html xxii](#bookmark430" \t "Current Document)

[A30 exam-mc.component.css xxiii](#bookmark435" \t "Current Document)

[A31 mc-query.ts xxiv](#bookmark443" \t "Current Document)

[A32 statistics.ts xxiv](#bookmark448" \t "Current Document)

[A33 stats.servicets xxv](#bookmark453" \t "Current Document)

[A34 lpic101.serice.ts xxv](#bookmark458" \t "Current Document)

A35 question-service.ts xxvi

**Abbildungsverzeichnis**

Abbildungsverzeichnis

Abbildung 1: Use-Case-Diagramm i

Abbildung 2: Prozess Hauptnavigation iv

Abbildung 3: Prozess Einzelfrage v

Abbildung 4: Startseite von LLCE vi

Abbildung 5: Dropdown-Liste der Kat vi

Abbildung 6: Fragenliste vii

Abbildung 7: Fill-In-Frage vii

Abbildung 8: Multiple-Choice viii

Abbildung 9: Single-Choice viii

Abbildung 10: SC-Frage mit grün markierter Antwort ix

Abbildung 11: Fill-In-Frage-falsche Antwort mit Pop-Up-Fenster ix

Abbildung 12: Hauptstruktur der App x

Abbildung 13: Auszug aus der Benutzerdokumentation Seite1 ...................... xxviii

Abbildung 14: Auszug aus der Benutzerdokumentation Seite2 xxix

**Verzeichnis der Listings**

Verzeichnis der Listings

Listing 1: app.module.ts xi

Listing 2: app-routing.module.ts xii

Listing 3: app.component.ts xiii

Listing 4: app.component.html xiii

Listing 5: home.component.ts xiv

Listing 6: home.component.html xv

Listing 7: home.component.css xv

Listing 8: check-all.component.ts xvi

Listing 9: check-all.component.html xvii

Listing 10: check-all.component.css xix

Listing 11: learn-mc-list.component.html xix

Listing 12: learn-mc-list.component.ts xxi

Listing 13: learn-mc-list.component.css xxi

Listing 14: exam-mc.component.ts xxii

Listing 15: exam-mc.component.html xxii

Listing 16: exam-mc.component.css xxiii

Listing 17: mc-query.ts xxiv

Listing 18: statistics.ts xxiv

Listing 19: stats.service.ts xxv

Listing 20: lpic101service.ts xxvL

**Tabellenverzeichnis**

Tabellenverzeichnis

[Tabelle 1: grobe Zeitplanung 2](#bookmark108" \t "Current Document)

[Tabelle 2: Qualitätsanforderungen 4](#bookmark135" \t "Current Document)

**Abkürzungsverzeichnis**

Abkürzungsverzeichnis

App *Application*

CSS3 *Cascading Style Sheets Version 3*

ERM *Entity Relationship Model*

GB *Gigabyte*

GUI *Graphical User Interface*

HTML5 *Hyper Text Markup Language Version 5*

JSON *Java Script Object Notation*

LINUX *Unix basierte Betriebssysteme*

LPIC *Linux Professional Institute Central Europe*

PC *Personal Computer*

UI *User Interface*

VM *Virtuelle Maschine*

**Einleitung**

1. Einleitun
   1. Projektumfeld

Das Projekt fand im Rahmen des Kurses "Enterprise Development mit HTML5, CSS3, Node.JS, JavaScript und Angular“ statt.

Die Teilnehmer hatten zum Einstieg des Kurses die App "BookMonkey" aus dem Buch Angular[[1]](#footnote-0) erfolgreich umgesetzt.

Das hier beschriebene Projekt wurde dann als Folgeprojekt inklusive Erstellung von Las­ten- und Pflichtenheft von den Teilnehmern umgesetzt.

* 1. Projektziel

Zur Festigung des mit der Umsetzung der App "BookMonkey" erworbenen Wissens sollte eine App programmiert werden, mit der es möglich ist, sich auf Zertifizierungsprüfungen vorzubereiten.

Sie sollte in der Lage sein, Multiple-Choice-, Single-Choice- und Fill-In-Fragen aus den LPIC-1-Fragen-Katalogen zu verarbeiten.

Die App sollte dabei sowohl die Möglichkeit zum reinen Lernen des Stoffs als auch zur Überprüfung des bereits erworbenen Wissens bieten. Sie sollte den Nutzer aber dazu er­ziehen, den Stoff zunächst intensiv zu lernen und erst dann sein Wissen zu überprüfen.

* 1. Projektbegründung

Die meisten vorhandenen Tools zur Prüfungsvorbereitung stellen eine reine Prüfungssi­mulation dar.

Der Benutzer hat keine Möglichkeit den Stoff zunächst zu lernen, sondern muss direkt in die Prüfung einsteigen. Dadurch besteht die Gefahr, dass sich bei falsch beantworteten Fragen dieses falsche Wissen verfestigt und es dem Nutzer in der Folge noch schwerer fällt, die korrekten Antworten zu lernen.

Die App sollte daher die Möglichkeit bieten, sich die korrekten Antworten anzeigen zu las­sen, ohne vorher selbst eine Antwort auswählen zu müssen. Darüber hinaus sollte der Nutzer bei der Überprüfung seines Wissens nur bei falscher Beantwortung ein (negatives) Feedback bekommen, um ihn noch mehr zu motivieren, den Stoff zunächst intensiv zu lernen.

* 1. Projektschnittstellen

Als Datengrundlage dienen der App die aus anderen Tools bekannten LPIC-1-Fragen-Ka- taloge. Diese wurden allerdings bereits im Vorfeld mit einem bash-Skript in ein passendes JSON-Format gewandelt und sind in der App selbst gespeichert.

**Projektplanung**

* 1. Zielgruppen

Als Benutzer der App kommen alle Personen in Frage, die sich auf Zertifizierungsprüfun­gen vorbereiten wollen, egal ob sie dies in einem schulischen Umfeld, am Arbeitsplatz oder rein privat machen wollen.

* 1. Projektabgrenzung

Der Zugriff auf Fragen-Kataloge von einer externen Quelle wie z.B. einem Webserver ist nicht Bestandteil der App. Alle Fragen sind direkt in der App hinterlegt.

1. Projektplanun
   1. Projektphasen

Die Umsetzung des Projekts erfolgte mit zwei Team-Mitgliedern an 16 Tagen im Zeitraum vom 20.03.2023 bis zum 07.04.2023 bei einer täglichen Arbeitszeit von acht Stunden.

Tabelle 1: grobe Zeitplanung

|  |  |
| --- | --- |
| **Projektphase** | **Geplante Zeit** |
| Erstellung des Lastenheftes | 14 h |
| Erstellung des Pflichtenheftes | 14 h |
| Planung der App | 5 h |
| Implementierung der App | 68 h |
| Testen der App | 5 h |
| Erstellung der Dokumentation | 24 h |
| **Gesamt** | **130 h** |

**Analysephase**

* 1. Ressourcenplanung

Für jedes Team-Mitglied wurden folgende Ressourcen eingeplant:

**Software:** CentOS7 64bit-Betriebssystem in einer VMware Workstation 15 Pro Pro-Umge­bung[[2]](#footnote-1), MS Visual-Studio-Code 1.76.2[[3]](#footnote-2), mit Entwicklungsumgebung für Angular[[4]](#footnote-3) 15.2.0, NodeJS®[[5]](#footnote-4) 16.14.2, HTML5, CSS3, aktueller Internet-Browser,

**Hardware:** Workstation mit Monitor, Tastatur und Maus

Darüber hinaus waren sowohl der Dozent als auch die anderen Kursteilnehmer als An­sprechpartner bei Problemen eingeplant.

* 1. Entwicklungsprozess

Bei der Durchführung des Projekts orientierte sich das Team am Wasserfall-Modell, d.h. die einzelnen Phasen der Umsetzung erfolgten linear nacheinander (siehe 2.1 Projektpha­sen).

1. Analysephase
   1. Ist-Analyse

Die bisher verfügbaren Apps zur Vorbereitung auf Zertifizierungsprüfungen bieten lediglich die Möglichkeit das bereits vorhandene Wissen zu prüfen. Der Nutzer muss sich also direkt einer Prüfungssimulation unterziehen, was zur Folge hat, dass sich ggf. falsches Wissen bei der Beantwortung der Fragen verfestigt.

Es fehlt bisher eine Möglichkeit den Stoff, ohne den Druck einer Prüfungssimulation mit der App zu erlernen. Der Benutzer ist gezwungen auf andere Medien wie Bücher, das Internet oder Vorbereitungs-Kurse zurückzugreifen..

* 1. Anwendungsfälle

Für die App ergeben sich daraus zwei konkrete Anwendungsfälle:

Zum einen das Erlernen des Prüfungs-Stoffs, ohne sich einer Prüfungs-Situation zu stel­len. Der Benutzer soll sich zu den Fragen jederzeit auch ohne Abgabe einer eigenen Ant­wort die korrekten Lösungen anzeigen lassen können.

**Analysephase**

Zum anderen die Überprüfung des bereits erworbenen Wissens. Wenn der Benutzer sich im Lernstoff fit genug fühlt, soll die App die von ihm gegebenen Antworten bewerten kön­nen.

Ein Beispiel für das Use-Case-Diagramm der App findet sich im Anhang [Abbildung 1.](#bookmark243" \t "Current Document)

* 1. Qualitätsanforderungen

Tabelle 2: Qualitätsanforderungen

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **sehr wichtig** | **wichtig** | **weniger wichtig** | **irrelevant** |
| **Robustheit** | **x** |  |  |  |
| **Zuverlässigkeit** | **x** |  |  |  |
| **Korrektheit** |  | **x** |  |  |
| **Benutzungsfreund­lichkeit** |  | **x** |  |  |
| **Effizienz** |  | **x** |  |  |
| **Portierbarkeit** |  |  |  | **x** |
| **Kompatibilität** |  |  |  | **x** |

* 1. Lastenheft

Laut Vorgabe aus dem Lastenheft soll die App als Prüfungs- und Zertifizierungssimulator dienen. Zielvorgaben sind u.a. Lauffähigkeit in einem aktuellen, beliebigen Web-Browser, Eingabe durch Tastatur und/oder Maus.

Im Auszug aus dem Pflichtenheft (siehe Abschnitt [4.6)](#bookmark175" \t "Current Document) wurde die Vorgabe um den Punkt „Lernen“ ergänzt.

Ein Auszug aus dem Lastenheft findet sich im Anhang, Absatz [A2](#bookmark244" \t "Current Document)

**Entwurfsphase**

1. Entwurfsphase
   1. Zielplattform

Die App sollte als Single-Page-Applikation in TypeScript mit dem Angular-Framework um­gesetzt werden.

Ziel war es, eine App zu erstellen, die von nahezu jedem Gerät mit einem modernen Inter­net-Browser, wie z.B. PC, Tablet oder Smartphone, aufgerufen werden kann.

Dabei sollten Netzwerkzugriffe und Wartezeiten für den Nutzer möglichst gering gehalten werden.

* 1. Architekturdesign

Gemäß dem Lernziel des Kurses wurde Angular als Framework gewählt. Angular bietet ein Rundum-Paket und Funktionalitäten wie Routing, Validierung und Kommunikation mit dem Backend, können mit einfachen Mitteln umgesetzt werden.

Durch die Transpilierung der fertigen App in einen gewünschten JavaScript-Standard, bei­spielsweise ECMA-Script 2020, ist die Entwicklung unabhängig vom Browser. Inzwi­schen unterstützen alle Browser die ECMA-Script 2020.

Darüber hinaus erlaubt es der modulare Ansatz von Angular auch komplexere Webseiten in kleinere überschaubare Komponenten zu gliedern.

* 1. Entwurf der Benutzeroberfläche

Beim Entwurf der Oberfläche wurde besonders auf eine übersichtliche und intuitiv zu be­dienende Steuerung geachtet.

Die App bietet dem Benutzer nur eine minimale Anzahl an Steuerungsmöglichkeiten, schränkt ihn dabei aber in keiner Weise ein. Der Benutzer kann sich innerhalb der Fragen­Kataloge völlig frei bewegen und ist nicht an eine bestimmte Bearbeitungs-Reihenfolge gebunden.

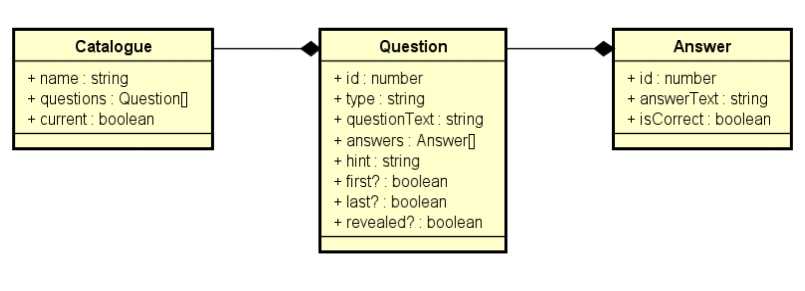
Für die Gestaltung der Oberflächen wurde das Semantic-UI-CSS eingesetzt, da es sich hierbei um ein sehr variables und bereits beim "BookMonkey"-Projekt erprobtes CSS­Framework handelt.

Beispielentwürfe finden sich im Anhang [Abbildung 4: Startseite von LLCE](#bookmark296" \t "Current Document) und nachfol­gende Seiten bis Abbildung 9 einschließlich.

**Entwurfsphase**

* 1. Datenmodell

Die Basis der Anwendung im nachfolgenden Klassendiagramm dargestellt:



Die EPK im Anhang, [Abbildung 2](#bookmark286" \t "Current Document) zeigt die Hauptnavigation der App, die EPK Abbildung 3 zeigt den Prozess mit dem einzelne Fragen abgearbeitet werden.

* 1. Maßnahmen zur Qualitätssicherung

Zur weiteren Absicherung wurden die oben genannten Browser mit der Anwendung auf den Linux-Distributionen:

CentOS[[6]](#footnote-5) [[7]](#footnote-6) [[8]](#footnote-7) [[9]](#footnote-8), Ubuntu und Fedora, MS-Windows107 und MacOS10.148 getestet.

Zur Qualitätssicherung wurde die Anwendung auf MS-Edge9(nicht für Linux), Opera[[10]](#footnote-9), Fire­fox[[11]](#footnote-10), Safari[[12]](#footnote-11) (unter Ubuntu) und Google-Chrome[[13]](#footnote-12) getestet insofern diese für die oben genannten Betriebssysteme zur Verfügung standen.

* 1. Pflichtenheft

Im Pflichtenheft wurden die Vorgaben aus dem Lastenheft auf Muss- und Wunschkriterien aufgesplittet.

Die Mussvorgaben wurden vollständig umgesetzt, ebenso die Punkte Responsives De­sign, Single-Choice-Fragetypen und Fill-In-Fragetypen aus den Wunschkriterien.

Ein Auszug aus dem Pflichtenheft ist im Anhang [A3](#bookmark253" \t "Current Document) zu finden.

**Implementierungsphase**

1. Implementierungsphase
   1. Implementierung der Datenstrukturen

Für die Verarbeitung der Fragen innerhalb der App wurde die Klasse *question.ts* entwickelt. Die für jede Frage ihre ID, den Typ, den Fragetext, die möglichen Antworten sowie die Information ob es die erste oder die letzte Frage im Katalog ist abbildet.

Die Struktur wurde so gewählt, dass alle drei Typen (Multiple-Choice, Single-Choice, Fill- In) in der gleichen Klasse abgebildet werden können. Über den in der Frage gespeicherten Typ kann die App die für die Frage passende Darstellung wählen. Die Information, ob die Frage die erste oder letzte im Katalog ist, wird bei Auswahl eines Katalogs jedes Mal neu generiert.

Dadurch kann später eine zufällige Reihenfolge der Fragen leicht implementiert werden.

Die möglichen Antworten zu einer Frage bilden einen Array der Klasse *answer.ts.* Diese Klasse bildet für jede Antwortmöglichkeit eine ID und den Antworttext ab, sowie die Infor­mation ob diese Antwort korrekt ist oder nicht.

Da jede Antwortmöglichkeit selbst die Information beinhaltet ob sie korrekt ist oder nicht, kann später eine zufällige Reihenfolge der Antwortmöglichkeiten problemlos implementiert werden.

* 1. Implementierung der Benutzeroberfläche

Bei der Implementierung der Benutzeroberfläche wurde auf eine einfache und selbsterklä­rende Struktur geachtet.

Es gibt drei Ansichten, zwischen denen der Nutzer wechseln kann:

Eine Startseite zur Auswahl eines der verfügbaren Kataloge, eine Übersichtsliste mit allen Fragen des gewählten Katalogs und eine Einzelansicht zur Bearbeitung der Fragen. Zum Wechsel zwischen den Ansichten wurde ein Routing-Modul implementiert.

Farbige Kennzeichnung:

Korrekte Antworten werden grün markiert (siehe [Abbildung 10)](#bookmark326" \t "Current Document).

Wurde eine Frage falsch beantwortet erscheint ein Pop-Up-Fenster mit einem entspre­chenden Hinweis und rot markierter Antwort (siehe Abbildung 11).

Die Screenshots der Anwendung in der Entwicklungsphase befinden sich im Anhang ab Abschnitt [A6.](#bookmark291" \t "Current Document)

* 1. Implementierung der Geschäftslogik

Die Anwendung nutzt zur Bereitstellung der Fragen die Service-Klasse *question.service.ts.* Diese kann einen gesamten Fragenkatalog, eine einzelne Frage sowie eine Liste der ver­fügbaren Fragenkataloge liefern.

Die Navigation zwischen den einzelnen Fragen geschieht anhand der Position im Katalog und nicht anhand der in der Frage gespeicherten ID.

**Benutzerdokumentation**

Dadurch sollte eine spätere Implementierung einer zufälligen Reihenfolge der Fragen leichter zu realisieren sein.

Für die Darstellung der einzelnen Frage-Typen ist der in der Frage gespeicherte Typ rele­vant. Um die verschiedenen Logiken bei der Anzeige und Bewertung der einzelnen Typen sauber trennen zu können, verwendet die App dazu drei Unterkomponenten der Detailan­sicht.

Die Buttons "Lernen" und "Prüfen" sind niemals gleichzeitig aktiv. Sie orientieren sich da­ran, ob der Nutzer bereits eine Antwortmöglichkeit angeklickt bzw. eingetragen hat.

Ist dies der Fall, kann der Nutzer nur den Button "Prüfen" anklicken, der Button "Lernen" ist inaktiv. Hat der Nutzer noch nichts unternommen, ist es genau umgekehrt.

Die Listings zu den Modulen befinden sich im Anhang ab [0](#bookmark340" \t "Current Document)

1. Benutzerdokumentation

Die Benutzerdokumentation zur App konnte kurz gehalten werden, da die meisten Funkti­onen selbsterklärend sind.

Ein Ausschnitt aus der erstellten Benutzerdokumentation befindet sich im Anhang [A36.](#bookmark470" \t "Current Document)

1. Fazit

7.1 Soll-/Ist-Vergleich

Das im Vorfeld festgelegte Projektziel wurde erreicht. In einzelnen Bereichen konnten so­gar darüber hinausgehende Funktionalitäten umgesetzt werden. So kann die App bereits drei verschiedene Frage-Typen verarbeiten, obwohl ursprünglich nur die Umsetzung für den Typ Multiple-Choice geplant war.

1. Lessons Learned

Nach der Umsetzung des BookMonkey-Projekts[[14]](#footnote-13) aus dem Buch Angular[[15]](#footnote-14) konnte die Fer­tigkeiten zur Programmierung von Single-Page-Anwendungen unter Einsatz von Angular, HTML5 und Semantic-UI-CSS[[16]](#footnote-15) mit diesem Projekt deutlich verfestigt werden.

**Fazit**

Viele Konzepte konnten aus dem BookMonkey-Projekt adaptiert werden, erforderten aber immer Anpassungen, so dass eine intensive Auseinandersetzung mit den Themen unum­gänglich war.

Durch den Austausch innerhalb der Gruppe konnten viele Hindernisse beseitigt werden und es ergab sich immer wieder die Möglichkeit andere Ansätze zur Lösung der Proble­matiken kennenzulernen und zu diskutieren.

Stetiges Testen der Funktionalitäten zeigte häufig Detailschwächen in der Programmie­rung, deren Lösung in einigen Fällen schwierig zu finden waren. Letztendlich gelang dies aber in jedem Punkt.

Die Verwendung von Semantic-UI-CSS bot eine Fülle von Möglichkeiten, die aber im Detail nicht immer exakt das gewünschte Erscheinungsbild ermöglichten.

Das Hinzufügen von eigenem CSS-Code führte aber in Verbindung mit Semantic häufig zu unerwarteten Ergebnissen, weshalb man sich auf eine der beiden Varianten beschrän­ken sollte.

Bei der Fehlersuche sollte aber auch die Entwicklungsumgebung selbst nicht ignoriert wer­den. Einige Male kam es zu Abstürzen innerhalb der Umgebung und eigentlich fehlerfreier Code konnte nicht korrekt ausgeführt werden. Als einfache Lösung für solche Probleme sollte die Entwicklungsumgebung neu gestartet werden. Auch kann ein Ab- und wieder Anmelden am Rechner konnte Abhilfe schaffen.

1. Ausblick

Bei der Entwicklung der App wurde an vielen Stellen bereits eine zukünftige Erweiterung um weitere sinnvolle Funktionen im Auge behalten.

Möglich wären unter anderem:

* Simulation einer Prüfung mit anschließender Auswertung
* Variable Anzahl an Fragen
* Zufällige Reihenfolge der Fragen und/oder Antworten
* Hilfestellungen zu Fragen
* Filterung der Fragen nach Typ und/oder Thema
* Bereitstellung der Fragenkataloge durch einen Webserver
* Implementierung weiterer Frage-Typen, z.B. mit Grafiken

**Literaturverzeichnis**

Literaturverzeichnis

**angular.io. 2023.** [Online] 2019. [Zitat vom: 02. 03 2023.] [www.angular.io](http://www.angular.io/).

**Apple Inc. 2023.** Apple.com. [Online] 2023. [Zitat vom: 02. 03 2023.] <https://www.apple.com/de/safari/>.

**centos.org. 2023.** centos. [Online] 2023. [Zitat vom: 02. 03 2023.] [https://www.centos.org](https://www.centos.org/).

**google Ireland Limited. 2023.** Google Chrome. [Online] 2023. [Zitat vom: 02. 03 2023.] <https://www.google.de/chrome/>.

**ISO/IEC 9126-1. 2001.** Software-Engineering - Qualität von Software-Produkten - Teil 1: Qualitätsmodell. *ISO/IEC 9126-1 2001.* Juni 2001.

**Microsoft Corp. 2023.** Microsoft Windows 10. [Online] 2023. [Zitat vom: 02. 03 2023.] <https://www.microsoft.com/de-de/windowsforbusiness/windows-10-pro>.

**Microsoft Corp. 2023.** Download Visual Studio Code. [Online] 02. 03 2023. <https://code.visualstudio.com/download>.

**2023.** Microsoft Edge. [Online] 2023. [Zitat vom: 26. 02 2023.] [https://www.microsoft.com/de- de/windows/microsoft-edge](https://www.microsoft.com/de-de/windows/microsoft-edge).

**Mozilla org. 2023.** mozilla.org. [Online] 2023. [Zitat vom: 26. 02 2023.] <https://www.mozilla.org/de/firefox/new/>.

**Node.js Foundation. 2023.** nodejs.org. [Online] 2023. [Zitat vom: 01. 03 2023.] <https://nodejs.org/en/>.

**Opera Software AS (NO). 2023.** Opera Software. [Online] 2023. [Zitat vom: 26. 02 2023.] <https://www.opera.com/de>.

**selfhtml. 2023.** <!selfhtml/>. [Online] 2023. [Zitat vom: 01. 03 2023.] [www.selfhtml.org](http://www.selfhtml.org/).

**semantic.ui. 2023.** semantic-ui.com. [Online] 2023. [Zitat vom: 01. 03 2023.] [www.semantic- ui.com](http://www.semantic-ui.com/).

**vmware. 2023.** vmware.com. [Online] 2023. [Zitat vom: 01. 03 2023.] [https://www.vmware.com](https://www.vmware.com/).

**w3schools. 2023.** w3schools.com. [Online] 2023. [Zitat vom: 01. 03 2023.] [www.w3schools.com](http://www.w3schools.com/).

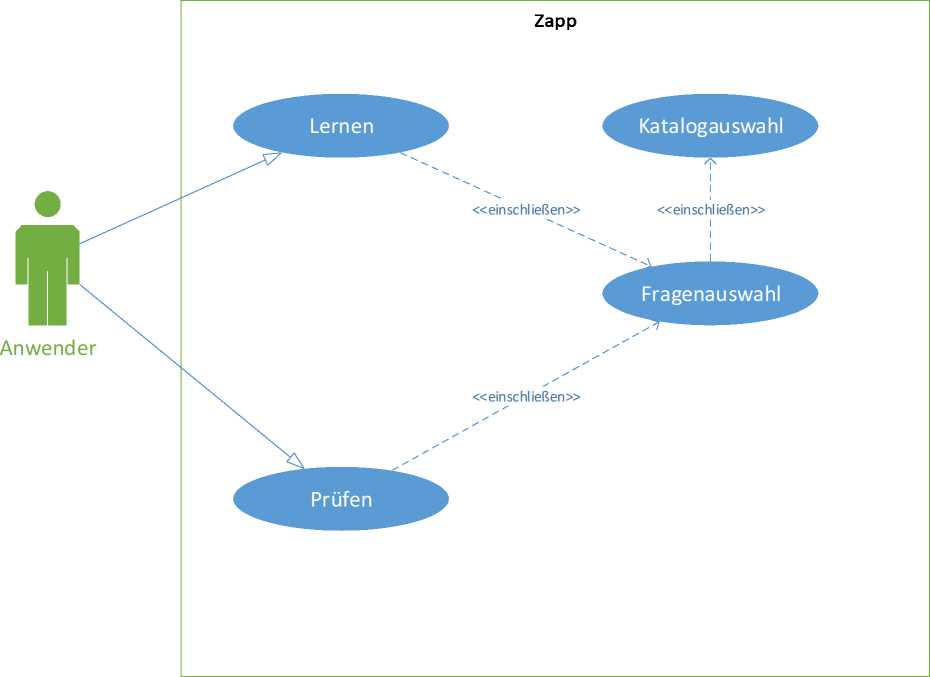
**Woiwode, Gregor, et al. 2023.** *Angular.* Heidelberg : dpunkt.verlag GmbH, 2023.

**Anhang**

A1 Use-Case-Diagramm

Abbildung 1: Use-Case-Diagramm

Anhang



A2 Lastenheft (Auszug)

Es folgt ein Auszug aus dem Lastenheft mit Fokus auf die Anforderungen:

**Die Anwendung muss folgende Anforderungen erfüllen.**

1. **Zweck und Ziel dieses Dokuments**

Dieses Dokument beschreibt das Projekt „LLCE“, einen Prüfungs- und Zertifizierungssimulator. Fol­gend „LLCE“, „Programm“ oder „App“ (Applikation) genannt. Dieses Dokument dient:

* Als grober Entwurf
* Als Anfragebasis

**2.Ziel des Projektes**

Das Ziel dieses Projektes ist die Erstellung einer Applikation mit der sich Anwender, durch die Nut­zung der App im Webbrowser, auf Prüfungen bzw. Zertifikate, vorbereiten können.

Beispiele für Prüfungen: Interne Prüfungen wie Klausuren.

Beispiele für Zertifikate: Zertifizierungsprüfungen bei anerkannten Zertifizierungsstellen wie z.B. CompTIA1

**3.1App Laufzeitumgebung**

Aktueller, beliebiger, Webbrowser, Eingabe durch Tastatur und/oder Maus.

A3 Pflichtenheft (Auszug)

**Zielbestimmung**

Der nachfolgende Absatz ist ein Auszug aus dem Pflichtenheft, mit dem Fokus auf die Muss- und Wunschkriterien.

**/PZ10/ Erstellen eines Prüfungs- und Zertifizierungssimulators** (/Punkt 1, 2/)

Wir entwickeln für Sie unter dem Projektnamen „LLCE“ eine Webseite zur Vorbereitung auf Zertifi­zierungs-Prüfungen. Personen, die sich auf eine Zertifizierungs-Prüfung vorbereiten, sollen ein ein­faches Werkzeug an die Hand bekommen, mit dem sie den Lernstoff anhand von Multiple-Choice- Fragen trainieren können.

Aus Ihrem Lastenheft haben wir die bestehenden Anforderungen, in Musskriterien und Wunschkri­terien aufgegliedert und ergänzt.

* 1. Musskriterien
* Plattformunabhängigkeit, d.h. Einsatz auf allen marktüblichen Betriebssystemen.
* Browsertauglichkeit für moderne marktübliche Browser. (/Punkt 3.2/)
* Einfaches Handling.
* Multiple-Choice-Fragestellung: Zu jeder Frage existieren mehrere Antwortmöglichkeiten, von denen mehrere korrekt sind.
  1. Wunschkriterien
* Nach erfolgreicher Einführung die Erweiterung auf andere Lerngebiete.
* Zeit -und ortsunabhängige Nutzungsmöglichkeit.
* Effizientes Lernen und hohe Erfolgsquote bei den Prüfungen.
* Bearbeitung zusätzlicher Frage-Formen: Single-Choice (nur eine Antwort ist auswählbar) und Fill-In (Antwort wird als Text eingegeben).
* Responsive Design für die Nutzung auf Smartphone, Tablet etc.
* Es sollen die Tags angezeigt werden, nach denen bereits gefiltert wird und die, die noch der Filterung hinzugefügt werden könnten, ohne dass die Ergebnisliste leer wird.
* Zusätzlich sollen die Module angezeigt werden, die den Filterkriterien entsprechen. Sollten die Filterkriterien leer sein, werden nur die Module angezeigt, welche mit einem Tag ver­sehen sind.

**Produkteinsatz**

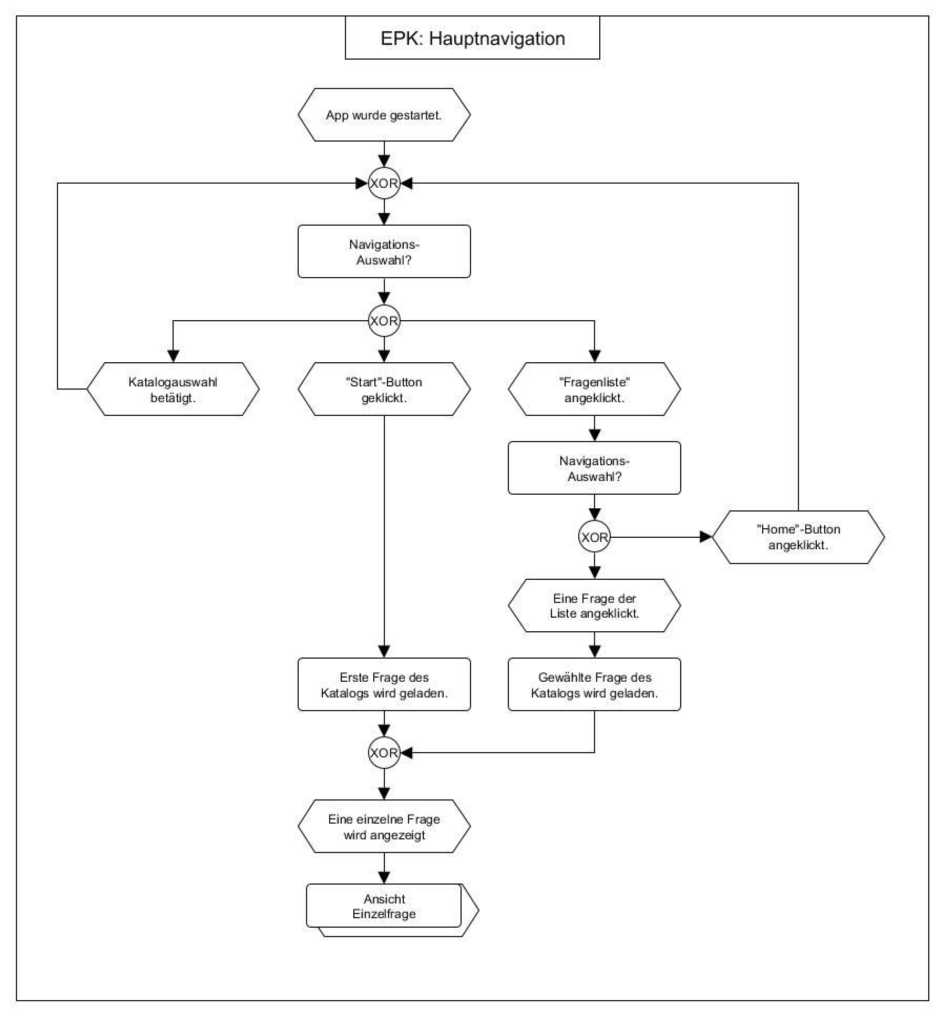
1. Anwendungsbereiche

1.1 Die Webanwendung dient als Anlaufstelle für die Entwicklung. Dort sind alle Informationen für die Module an einer Stelle gesammelt. Vorher getrennte Anwendungen werden ersetzt bzw. verlinkt.

1. Zielgruppen
   1. LLCE wird von allen Menschen genutzt, die sich auf Prüfungen vorbereiten müssen.
2. Betriebsbedingungen
   1. Die nötigen Betriebsbedingungen, also der Webserver,

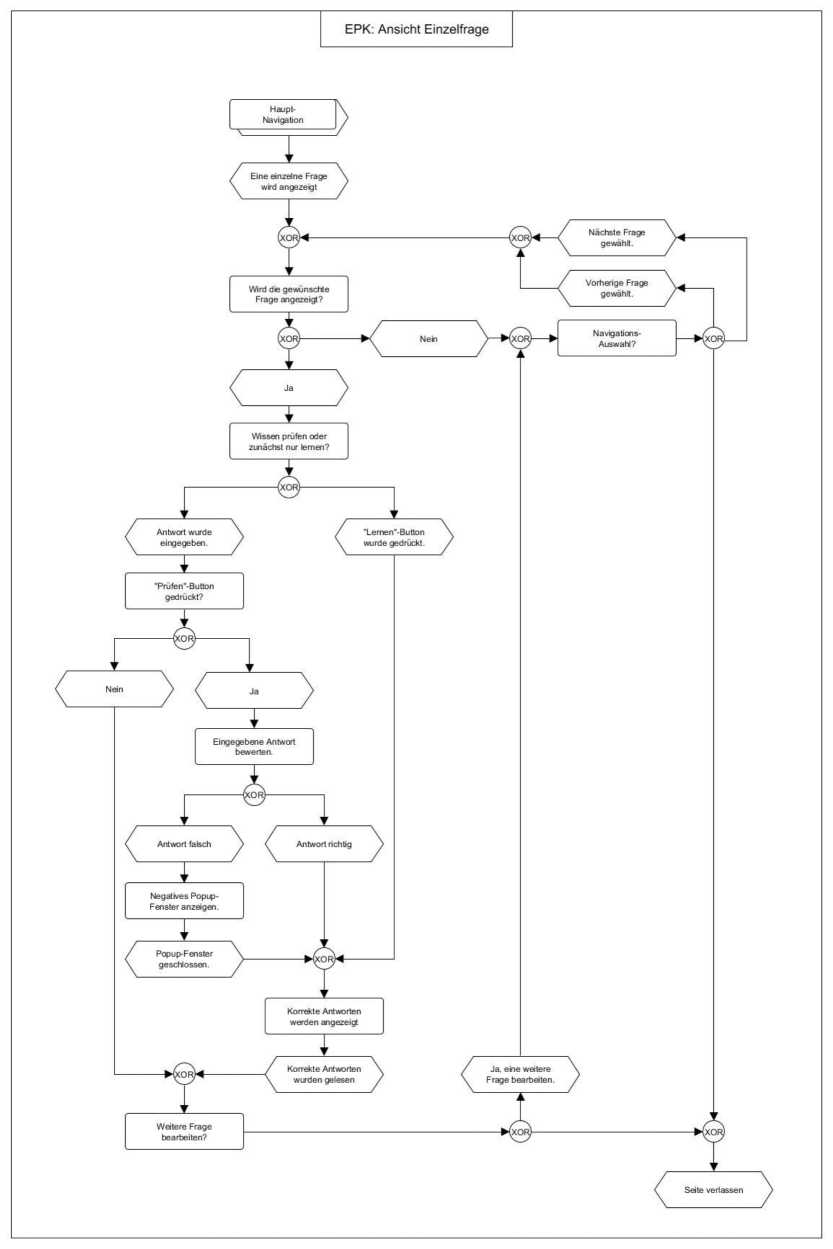
**Anhang**

A4 Ereignisgesteuerte Prozesskette Hauptnavigation



**Anhang**

A5 Ereignisgesteuerte Prozesskette Einzelfrage



**Anhang**

A6 Screenshot der Startseite

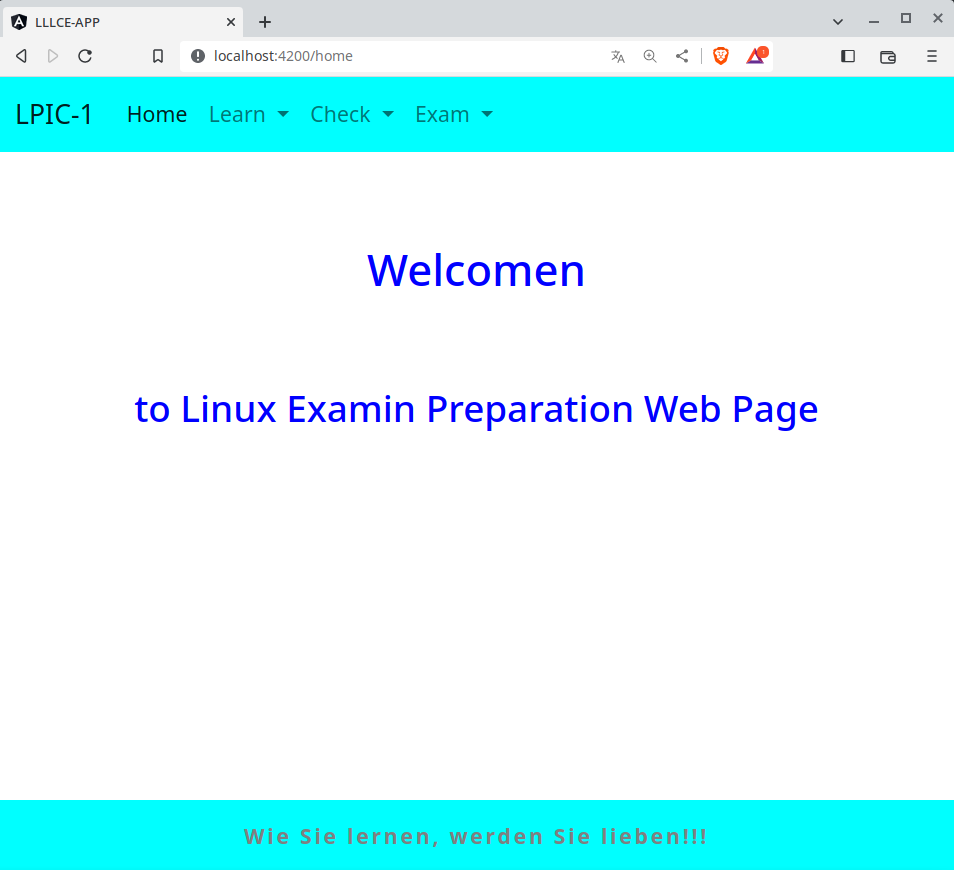


Abbildung 4: Startseite von LLCE

A7 Screenshot Startseite Dropdown Katalogauswahl

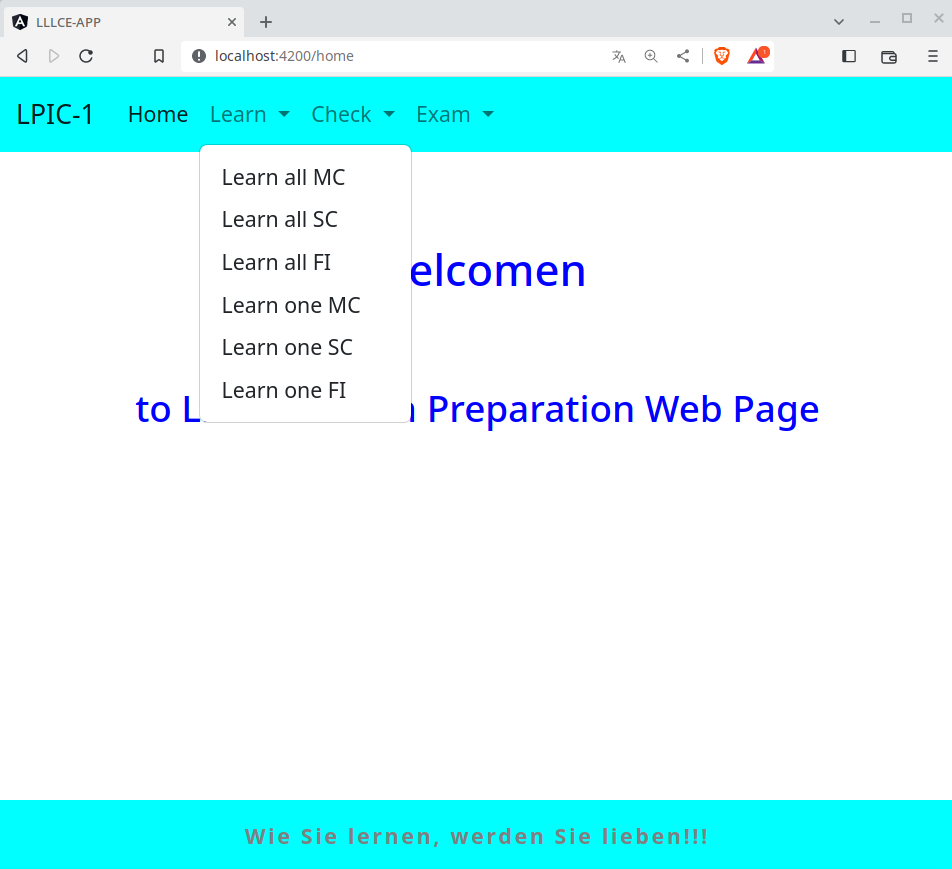


Abbildung 5: Dropdown-Liste der Kataloge

**Anhang**

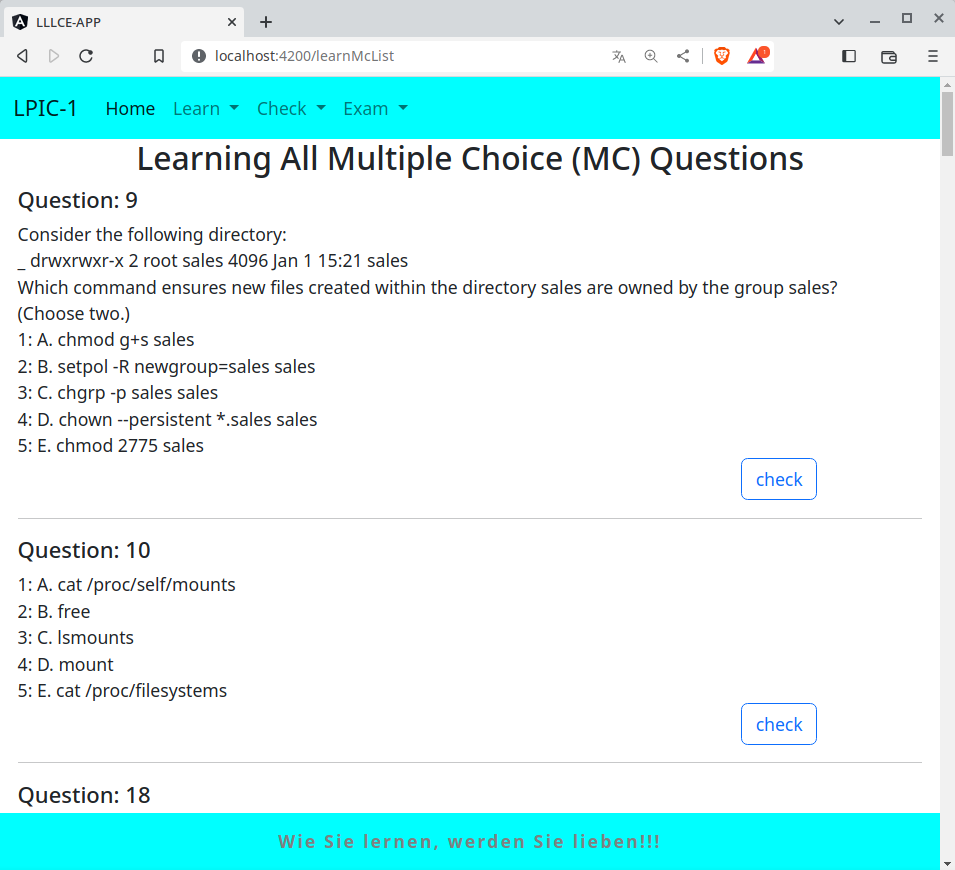


Abbildung 6: Mutiple Choice Fragenliste

A9 Screenshot Einzelfrage (Fill-Inn-Frage)

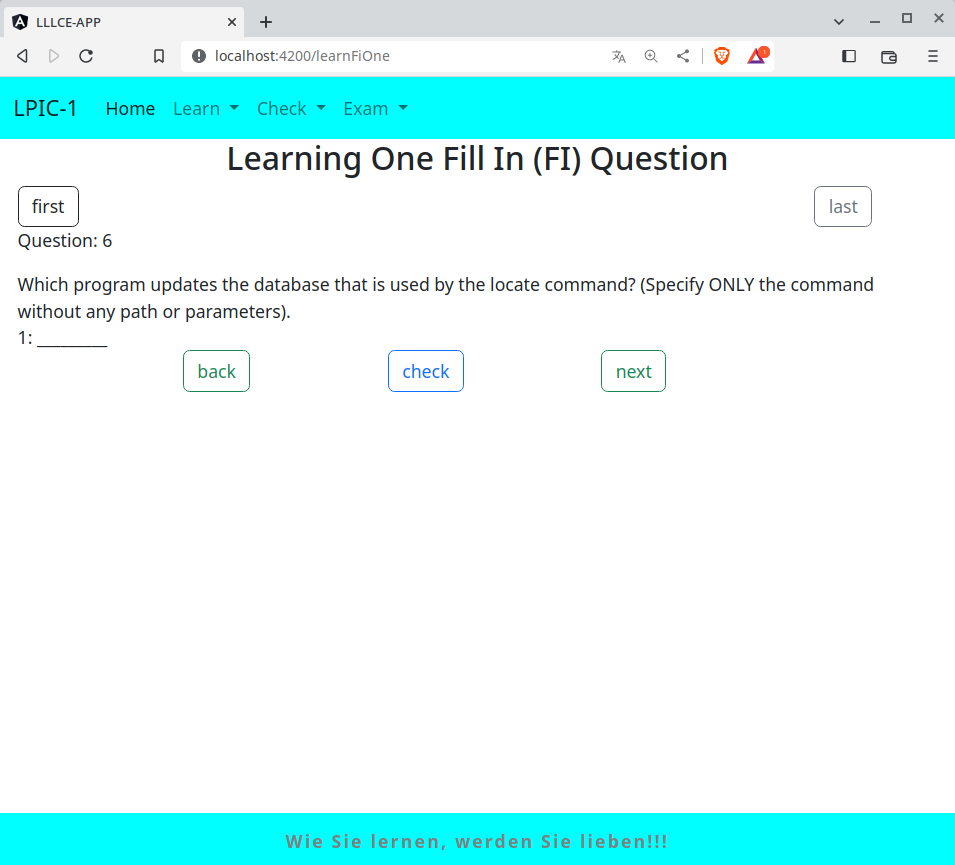


Abbildung 7: Fill-in-Frage

**Anhang**

A10 Screenshot Einzelfrage (Multiple-Choice-Frage)

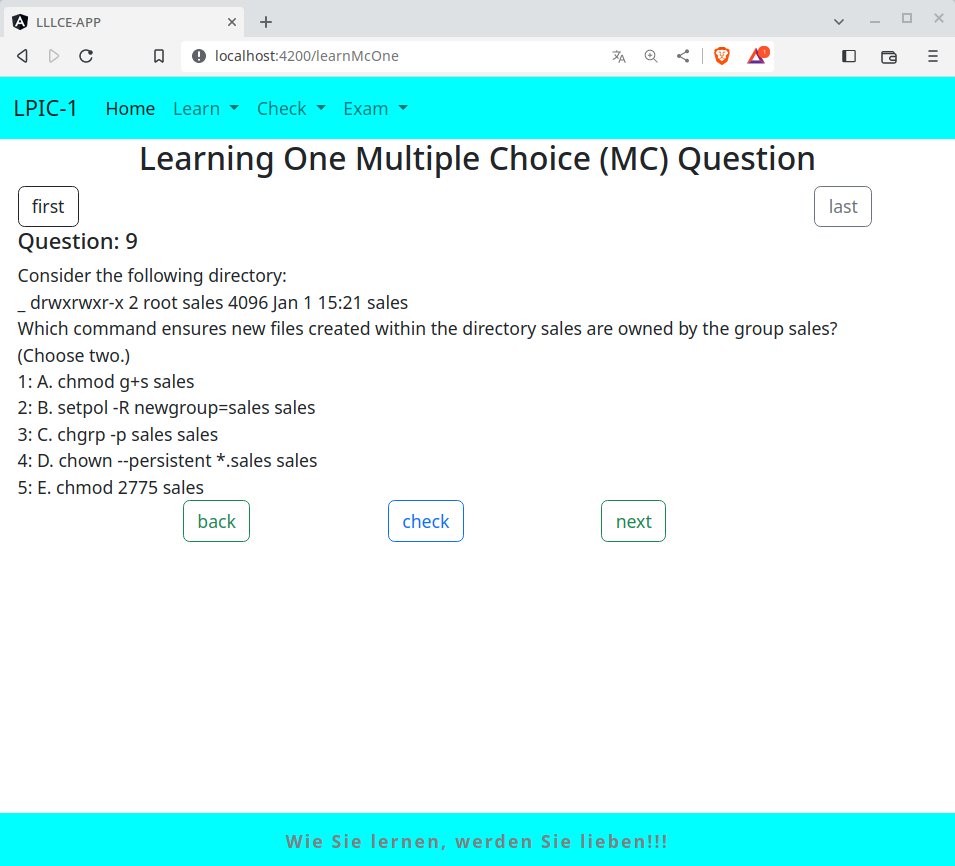


Abbildung 8: Multiple-Choice

A11 Screenshot Einzelfrage (Single-Choice-Frage)

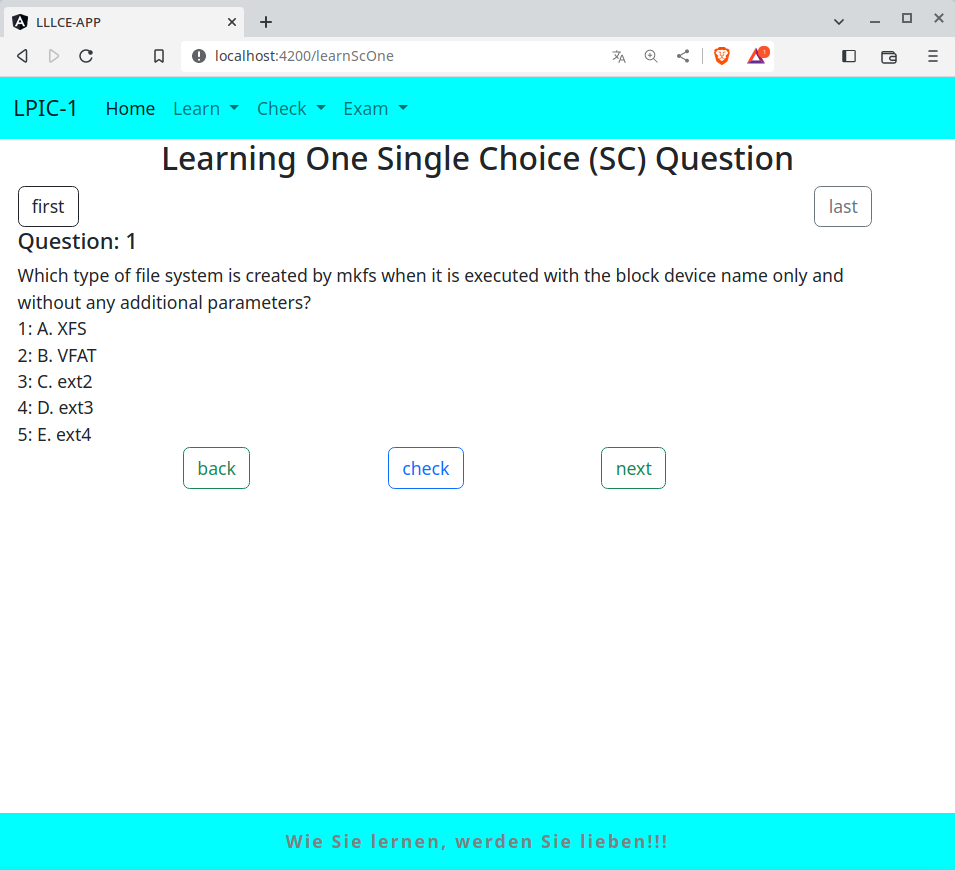


Abbildung 9: Single-Choice

**Anhang**

A12 Screenshot Einzelfrage (Single-Choice-Frage)

Abbildung 10: SC-Frage mit grün Antwort

A13 Screenshot Einzelfrage (Fill-In-Frage)

Abbildung 10: SC-Frage mit grün Antwort

Abbildung 10: SC-Frage mit grün Antwort

Abbildung 10: SC-Frage mit grün Antwort

Abbildung 10: SC-Frage mit grün Antwort

Abbildung 10: SC-Frage mit grün Antwort

Abbildung 10: SC-Frage mit grün Antwort

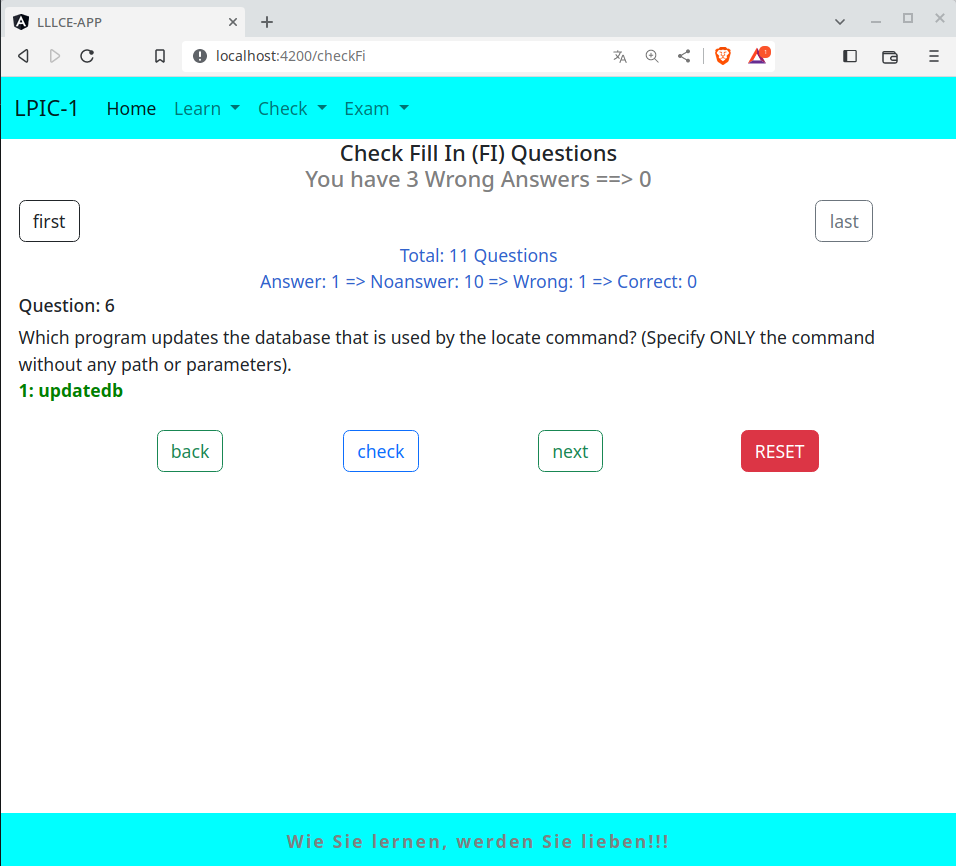
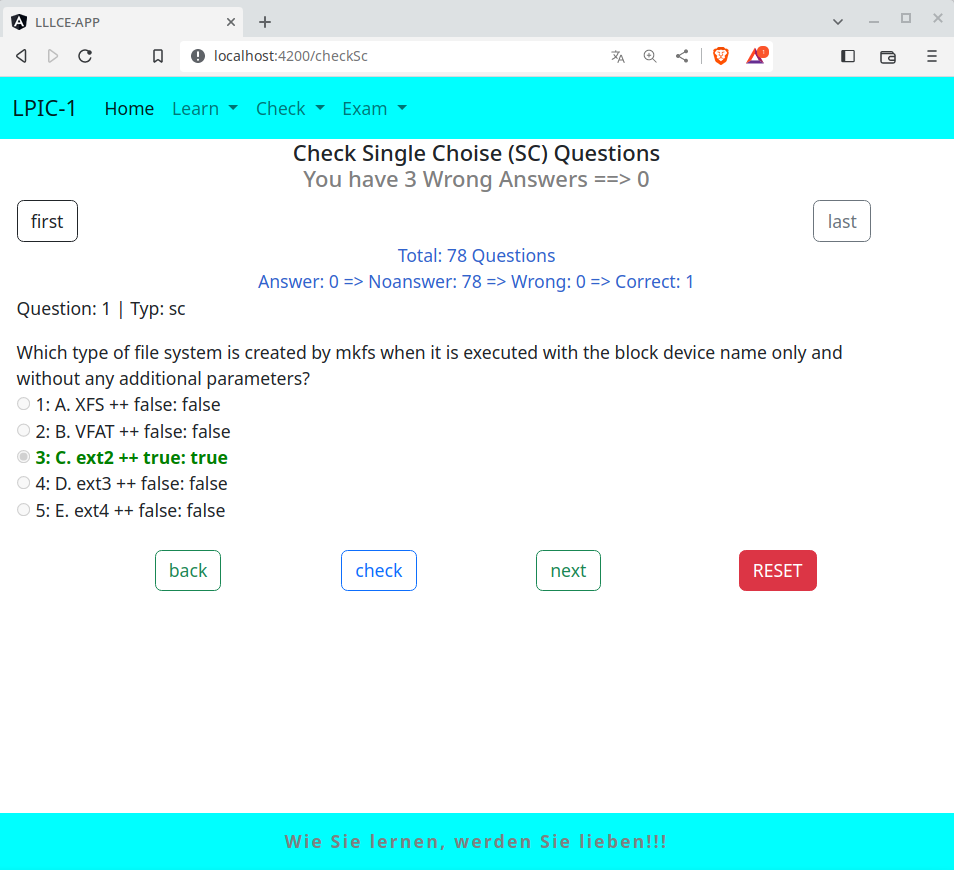


Abbildung 11: Fill-in-Frage-falsche Antwort

A14 Haupt-Ordnersturktur Angular

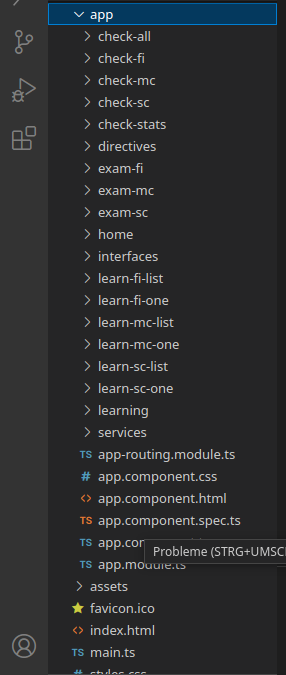


Abbildung 12: Hauptstruktur der App

**Anhang**

A15 Listing app.module.ts

import { NgModule } from '@angular/core';

import { BrowserModule } from '@angular/platform-browser';

import { AppRoutingModule } from './app-routing.module';

import { AppComponent } from './app.component';

import { NgbModule } from '@ng-bootstrap/ng-bootstrap';

import { LearnMcListComponent } from './learn-mc-list/learn-mc-list.component';

import { HomeComponent } from './home/home.component';

import { LearnMcOneComponent } from './learn-mc-one/learn-mc-one.component';

import { CheckMcComponent } from './check-mc/check-mc.component';

import { CheckScComponent } from './check-sc/check-sc.component';

import { CheckFiComponent } from './check-fi/check-fi.component';

import { ExamMcComponent } from './exam-mc/exam-mc.component';

import { ExamScComponent } from './exam-sc/exam-sc.component';

import { ExamFiComponent } from './exam-fi/exam-fi.component';

import { LearnScListComponent } from './learn-sc-list/learn-sc-list.component';

import { LearnScOneComponent } from './learn-sc-one/learn-sc-one.component';

import { LearnFiListComponent } from './learn-fi-list/learn-fi-list.component';

import { LearnFiOneComponent } from './learn-fi-one/learn-fi-one.component';

import { CheckStatsComponent } from './check-stats/check-stats.component';

import { AutoFocusInputDirective } from './directives/auto-focus-input.directive';

import { LearningComponent } from './learning/learning.component';

import { CheckAllComponent } from './check-all/check-all.component';

@NgModule({

declarations: [

AppComponent,

LearnMcListComponent,

HomeComponent,

LearnMcOneComponent,

CheckMcComponent,

CheckScComponent,

CheckFiComponent,

ExamMcComponent,

ExamScComponent,

ExamFiComponent,

LearnScListComponent,

LearnScOneComponent,

LearnFiListComponent,

LearnFiOneComponent,

CheckStatsComponent,

AutoFocusInputDirective,

LearningComponent,

CheckAllComponent

],

imports: [

BrowserModule,

AppRoutingModule,

NgbModule

],

providers: [],

bootstrap: [AppComponent]

})

export class AppModule { }

Listing 1: app.module.ts

**Anhang**

A16 app-routing.module.ts

|  |
| --- |
| Iimport { Component, NgModule } from '@angular/core';  import { RouterModule, Routes } from '@angular/router';  import { LearnMcListComponent } from './learn-mc-list/learn-mc-list.component';  import { HomeComponent } from './home/home.component';  import { AppComponent } from './app.component';  import { LearnMcOneComponent } from './learn-mc-one/learn-mc-one.component';  import { LearnFiOneComponent } from './learn-fi-one/learn-fi-one.component';  import { LearnFiListComponent } from './learn-fi-list/learn-fi-list.component';  import { LearnScOneComponent } from './learn-sc-one/learn-sc-one.component';  import { LearnScListComponent } from './learn-sc-list/learn-sc-list.component';  import { CheckMcComponent } from './check-mc/check-mc.component';  import { CheckScComponent } from './check-sc/check-sc.component';  import { CheckFiComponent } from './check-fi/check-fi.component';  import { ExamMcComponent } from './exam-mc/exam-mc.component';  import { LearningComponent } from './learning/learning.component';  import { ExamFiComponent } from './exam-fi/exam-fi.component';  import { ExamScComponent } from './exam-sc/exam-sc.component';  import { CheckAllComponent } from './check-all/check-all.component';  const routes: Routes = [  { path: '', redirectTo: 'home', pathMatch: 'full' },  { path: 'llce', component: AppComponent },  { path: 'home', component: HomeComponent },  { path: 'learning', component: LearningComponent },  { path: 'learnMcList', component: LearnMcListComponent },  { path: 'learnMcOne', component: LearnMcOneComponent },  { path: 'learnScList', component: LearnScListComponent },  { path: 'learnScOne', component: LearnScOneComponent },  { path: 'learnFiList', component: LearnFiListComponent },  { path: 'learnFiOne', component: LearnFiOneComponent },  { path: 'checkAll', component: CheckAllComponent },  { path: 'checkMc', component: CheckMcComponent },  { path: 'checkSc', component: CheckScComponent },  { path: 'checkFi', component: CheckFiComponent },  { path: 'examMc', component: ExamMcComponent },  { path: 'examSc', component: ExamScComponent },  { path: 'examFi', component: ExamFiComponent },  ];  @NgModule({  imports: [RouterModule.forRoot(routes)],  exports: [RouterModule]  })  export class AppRoutingModule { } |

Listing 2: app-routing.module.ts

**Anhang**

A17 app.component.ts

import { Component } from '@angular/core';

@Component({

selector: 'llceae-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'lileceexapp';

}

templateUrl: './app.component.html',

Listing 3: app.component.ts

A18 app.component.html

<div>

<nav class="navbar">

<nav class="navbar fixed-top navbar-expand-sm bg-light; navcolor">

<div class="container-fluid">

<a class="navbar-brand" href="#">LPIC-1</a>

<button

class="navbar-toggler"

type="button"

data-bs-toggle="collapse"

data-bs-target="#navbarNavDropdown"

aria-controls="navbarNavDropdown"

aria-expanded="false"

aria-label="Toggle navigation"

>

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarNavDropdown">

<ul class="navbar-nav">

<li class="nav-item">

<a class="nav-link active" aria-current="page" routerLink="home"

>Home</a >

</li>

<li class="nav-item dropdown">

<a

class="nav-link dropdown-toggle"

href="#"

role="button"

data-bs-toggle="dropdown"

aria-expanded="false"

>

Learn

</a>

<ul class="dropdown-menu">

<li>

<a class="dropdown-item" routerLink="learnMcList"

>Learn all MC</a

>

</li>

<li>

<a class="dropdown-item" routerLink="learnScList"

>Learn all SC</a

>

</li>

<li>

<a class="dropdown-item" routerLink="learnFiList"

>Learn all FI </a

>

</li>

<li>

<a class="dropdown-item" routerLink="learnMcOne"

>Learn one MC</a

>

</li>

<li>

<a class="dropdown-item" routerLink="learnScOne"

>Learn one SC</a

>

</li>

<li>

<a class="dropdown-item" routerLink="learnFiOne"

>Learn one FI </a

>

</li>

</ul>

</li>

<li class="nav-item dropdown">

<a

class="nav-link dropdown-toggle"

href="#"

role="button"

data-bs-toggle="dropdown"

aria-expanded="false" >

Check

</a>

<ul class="dropdown-menu">

<li>

<a class="dropdown-item" routerLink="checkAll"

>Check All Questions</a >

</li>

<li>

<a class="dropdown-item" routerLink="checkMc"

>Check MC Questions</a

>

</li>

<li>

<a class="dropdown-item" routerLink="checkSc"

>Check SC Questions</a

>

</li>

<li>

<a class="dropdown-item" routerLink="checkFi"

>Check FI Questions</a

>

</li>

</ul>

</li>

<li class="nav-item dropdown">

<a

class="nav-link dropdown-toggle"

href="#"

role="button"

data-bs-toggle="dropdown"

aria-expanded="false"

>

Exam

</a>

<ul class="dropdown-menu">

<li>

<a class="dropdown-item" routerLink="examMc"

>Exam Mode MC</a

>

</li>

<li>

<a class="dropdown-item" routerLink="examSc"

>Exam Mode SC</a

>

</li>

<li>

<a class="dropdown-item" routerLink="examFi"

>Exam Mode FI</a>

</li>

</ul>

</li>

</ul>

</div>

</div>

</nav>

</nav>

</div>

<div class="main">

<span> ------ </span>

<router-outlet></router-outlet>

</div>

<footer class="footer">

<br>

Wie Sie lernen, werden Sie lieben!!!

<br> <br>

</footer>

<div class="row">

Listing 4: app.component.html

**Anhang**

A19 home.component.ts

import { Component } from '@angular/core';

@Component({

selector: 'llceae-home',

templateUrl: './home.component.html',

styleUrls: ['./home.component.css']

})

export class HomeComponent {

}

Listing 5: home.component.ts

A20 home.component.html

<h1 class="paint">Welcomen</h1>

<h2 class="paint">to Linux Examin Preparation Web Page</h2>

<i class="arrow alternate circle left outline huge icon navigationArrow" (click)="prevQuestion()"></i>

Listing 6: home.component.html

A21 home.component.css

..paint{

color: blue;

text-align: center;

margin-top: 10%;

}

Listing 7: question-detail.component.css

**Anhang**

A22 check-all.component.ts

* Iimport { Component } from '@angular/core';
* import { Statistics } from '../interfaces/statistics';
* import { Qlpic101Service } from '../services/qlpic101.service';
* import { StatsService } from '../services/stats.service';
* import { LceQuery } from '../interfaces/lcequery';
* @Component({
* selector: 'llceae-check-all',
* templateUrl: './check-all.component.html',
* styleUrls: ['./check-all.component.css']
* })
* export class CheckAllComponent {
* ql101all: LceQuery[] = []
* showanswers = false
* queryNrAnswersShow = -1
* query: LceQuery;
* currentQnr = -1
* correctInput: boolean = false;
* statistic: Statistics;
* gotolearnmode: boolean
* learnwrong: number
* maxlearnwrong = 3
* constructor(
* private ql101srvc: Qlpic101Service,
* private stats: StatsService
* ) {
* this.ql101all = this.ql101srvc.getallAll()
* this.ql101srvc.initGivenAnswers()
* this.statistic = this.stats.calcStatsAll()
* this.currentQnr = 0
* this.query = this.ql101all[this.currentQnr]
* this.gotolearnmode = false
* this.learnwrong = 0
* }
* shuffleAll() {
* console.log('in shuffle')
* }
* resetStats() {
* this.statistic = this.stats.resetStatsAll()
* this.firstQuery()
* this.gotolearnmode = false
* this.learnwrong = 0
* }
* resetAnswers() {
* this.ql101all.map(q => q.qanswers.map(a => a.givenans = false))
* this.resetStats()
* this.firstQuery()
* this.learnwrong = 0
* }
* refreshStats() {
* this.statistic = this.stats.calcStatsAll()
* }
* firstQuery() {
* this.currentQnr = 0
* this.query = this.ql101all[this.currentQnr]
* this.showanswers = false
* this.refreshStats()
* }
* prevQuery() {
* if (0 < this.currentQnr) {
* this.currentQnr--
* this.query = this.ql101all[this.currentQnr]
* }
* this.showanswers = false
* this.refreshStats()
* }
* nextQuery() {
* // check if current question answered and correct -> next
* // else update learnwrong counter
* if (this.query.qtyp === 'mc') {
* if (this.checkQueryMcAnswered()) {
* this.gotolearnmode = false
* if (!this.checkQueryMcAnsweredCorrect()) {
* // answered and false -> PopUp and prev question
* // learnwrong + 1 (max 7)
* this.learnwrong++
* // reset answers of current query
* this.query.qanswers.map(a => a.givenans = false)
* // this.query.qgiventxt = ''
* // back one query or to first query
* this.prevQuery()
* this.refreshStats()
* console.log('Learn wrong: ', this.learnwrong)
* if (this.learnwrong >= this.maxlearnwrong) {
* // 7 wrong, this is bad :( - popup and go to learn mode
* this.gotolearnmode = true
* }
* } else { // answered and true
* // to next question
* this.setNextQuestion()
* }
* } else {
* // not answered -> next question
* this.setNextQuestion()
* }
* // console.log('curr q: ', this.currentQnr)
* } else if (this.query.qtyp === 'sc') {
* if (this.checkQueryScAnswered()) {
* this.gotolearnmode = false
* if (!this.checkQueryScAnsweredCorrect()) {
* // answered and false -> PopUp and prev question
* // learnwrong + 1 (max 7)
* this.learnwrong++
* // reset answers of current query
* this.query.qanswers.map(a => a.givenans = false)
* // back one query or to first query
* this.prevQuery()
* this.refreshStats()
* console.log('Learn wrong: ', this.learnwrong)
* if (this.learnwrong >= this.maxlearnwrong) {
* // 7 wrong, this is bad :( - popup and go to learn mode
* this.gotolearnmode = true
* }
* } else { // answered and true
* // to next question
* this.setNextQuestion()
* // }
* }
* } else {
* // not answered -> next question
* this.setNextQuestion()
* }
* // console.log('curr q: ', this.currentQnr)
* } else if (this.query.qtyp === 'fi') {
* if (this.checkQueryFiAnswered()) {
* this.gotolearnmode = false
* if (!this.checkQueryFiAnsweredCorrect()) {
* // answered and false -> PopUp and prev question
* // learnwrong + 1 (max 7)
* this.learnwrong++
* // reset answers of current query
* this.query.qgiventxt = ''
* // back one query or to first query
* this.prevQuery()
* this.refreshStats()
* console.log('Learn wrong: ', this.learnwrong)
* if (this.learnwrong >= this.maxlearnwrong) {
* // 7 wrong, this is bad :( - popup and go to learn mode
* this.gotolearnmode = true
* }
* } else { // answered and true
* // to next question
* this.setNextQuestion()
* // }
* }
* } else {
* // not answered -> next question
* this.setNextQuestion()
* }
* // console.log('curr q: ', this.currentQnr)
* }
* this.refreshStats()
* }
* setNextQuestion() {
* if (this.currentQnr < this.ql101all.length - 1) {
* this.currentQnr++
* this.query = this.ql101all[this.currentQnr]
* console.log(this.currentQnr, this.query)
* }
* this.showanswers = false
* this.refreshStats()
* }
* lastQuery() {
* this.currentQnr = this.ql101all.length - 1
* this.query = this.ql101all[this.currentQnr]
* this.showanswers = false
* this.refreshStats()
* }
* toggleAnswers(qid: number): void {
* if (this.queryNrAnswersShow != qid) {
* this.queryNrAnswersShow = qid;
* this.showanswers = true
* } else {
* this.showanswers = !this.showanswers
* }
* this.refreshStats()
* }
* toggleGivenAnswer(ansind: number) {
* this.query.qanswers[ansind].givenans = !this.query.qanswers[ansind].givenans
* this.refreshStats()
* }
* changeGivenAnswer(ansind: number) {
* // first reset all radios
* this.query.qanswers.map(ans => ans.givenans = false)
* // second set new radio
* this.query.qanswers[ansind].givenans = !this.query.qanswers[ansind].givenans
* this.refreshStats()
* }
* checkQueryMcAnswered() {
* // answered? (is one answer given 'true')
* if (this.query.qanswers.find(a => a.givenans === true)) {
* console.log('answered')
* return true
* } else {
* // not answered
* console.log('not answered')
* return false
* }
* }
* checkQueryMcAnsweredCorrect() {
* // correct answered? (are all givenans equal correct)
* if (this.query.qanswers.find(a => a.givenans != a.correct)) {
* // answered false
* console.log('answered false')
* return false
* } else {
* console.log('answered true')
* return true
* }
* }
* keyinput(myinput: string) {
* this.correctInput = false
* this.query.qgiventxt = myinput
* // check if correct
* if (this.query.qanswers.find(
* a => a.txt.find(t => t === this.query.qgiventxt))) {
* this.correctInput = true
* }
* console.log(this.correctInput, this.query.qgiventxt)
* this.refreshStats()
* }
* checkQueryScAnswered() {
* // answered? (is one answer given 'true')
* if (this.query.qanswers.find(a => a.givenans === true)) {
* // console.log('answered')
* return true
* } else {
* // not answered
* // console.log('not answered')
* return false
* }
* }
* checkQueryScAnsweredCorrect() {
* // correct answered? (is givenans equal correct)
* if (this.query.qanswers.find(a => a.givenans === true && a.givenans === a.correct)) {
* // console.log('answered true')
* return true
* } else {
* // answered false
* // console.log('answered false')
* return false
* }
* }
* checkQueryFiAnswered() {
* // answered? (is one answer given 'true')
* if (this.query.qgiventxt != '') {
* console.log('answered')
* return true
* } else {
* // not answered
* console.log('not answered')
* return false
* }
* }
* checkQueryFiAnsweredCorrect() {
* // correct answered? qgiventxt in one of answers txt
* if (this.query.qanswers.find(q => q.txt.find(at => at === this.query.qgiventxt))) {
* console.log('answered true')
* return true
* } else {
* // answered false
* console.log('answered false')
* return false
* }
* }
* }

Listing 8: answer-fi.component.ts

**Anhang**

A23 check-all.component.html

<h5 style="text-align: center;">

Check All Questions <br>

<span class="lives">You have 3 Wrong Answers ==> {{learnwrong }} </span>

</h5>

<div \*ngIf="!gotolearnmode">

<button type="button" class="btn btn-outline-dark" (click)="firstQuery()">first</button>

<button type="button" class="btn btn-outline-secondary" style="margin-left: 80%

;" (click)="lastQuery()">last</button>

<div class="statistics" style="text-align: center;" >

Total: {{ statistic.qmaxnumber }} Questions <br> Answer: {{ statistic.qanswered }} =>

Noanswer: {{ statistic.qnotanswered }} => Wrong: {{ statistic.qwrong }} =>

Correct: {{ statistic.qcorrect }}

</div>

<h5>Question: {{ query.qid }} <br /></h5>

<div \*ngFor="let qt of query.qtxt">

<span \*ngIf="qt[0] === ' '"> \_ {{ qt }} <br /> </span>

<span \*ngIf="qt[0] != ' '"> {{ qt }} <br /> </span>

</div>

<!-- MC -- MC-->

<div \*ngIf="query.qtyp === 'mc'">

<div \*ngFor="let ans of query.qanswers; index as ansind">

<span \*ngIf="showanswers">

<span \*ngIf="ans.correct">

<span \*ngIf="queryNrAnswersShow === query.qid" class="boldgreen">

<input type="checkbox" (click)="ans.givenans = !ans.givenans" checked disabled />

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</span>

</span>

<span \*ngIf="!ans.correct">

<span \*ngIf="queryNrAnswersShow === query.qid">

<input type="checkbox" (click)="toggleGivenAnswer(ansind)" disabled />

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</span>

</span>

<span \*ngIf="queryNrAnswersShow != query.qid">

<input type="checkbox" (click)="toggleGivenAnswer(ansind)" disabled />

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</span>

</span>

<span \*ngIf="!showanswers">

<div \*ngIf="ans.givenans">

<input type="checkbox" (click)="toggleGivenAnswer(ansind)" checked />

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</div>

<div \*ngIf="!ans.givenans">

<input type="checkbox" (click)="toggleGivenAnswer(ansind)" />

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</div>

</span>

</div>

</div>

<!-- SC -- SC-->

<div \*ngIf="query.qtyp === 'sc'">

<div \*ngFor="let ans of query.qanswers; index as ansind">

<span \*ngIf="showanswers">

<span \*ngIf="ans.correct">

<span \*ngIf="queryNrAnswersShow === query.qid" class="boldgreen">

<input type="radio" (click)="ans.givenans = !ans.givenans" checked disabled />

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</span>

</span>

<span \*ngIf="!ans.correct">

<span \*ngIf="queryNrAnswersShow === query.qid">

<input type="radio" (click)="changeGivenAnswer(ansind)" disabled />

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</span>

</span>

<span \*ngIf="queryNrAnswersShow != query.qid">

<input type="radio" (click)="changeGivenAnswer(ansind)" disabled />

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</span>

</span>

<span \*ngIf="!showanswers">

<div \*ngIf="ans.givenans">

<input type="radio" (click)="changeGivenAnswer(ansind)" checked />

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</div>

<div \*ngIf="!ans.givenans">

<input type="radio" (click)="changeGivenAnswer(ansind)" />

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</div>

</span>

</div>

</div>

<!-- FI - FillIn -- FI-->

<div \*ngIf="query.qtyp === 'fi'">

<div \*ngFor="let ans of query.qanswers; index as k">

<span \*ngIf="showanswers">

<span \*ngIf="queryNrAnswersShow === query.qid" class="boldgreen">

{{ k + 1 }}: {{ ans.txt }}

</span>

<span \*ngIf="queryNrAnswersShow != query.qid">

{{ k + 1 }}:

<input type="text" value="{{ ans.txt }}" />

</span>

</span>

<div>

<span \*ngIf="correctInput">

<span \*ngIf="!showanswers">

{{ k + 1 }}:

<input type="text" #myinput (keyup)="keyinput(myinput.value)" value="{{ query.qgiventxt }}"

class="boldgreen" llceaeAutoFocusInput (keyup.enter)="nextQuery()" /></span>

</span>

<span \*ngIf="!correctInput">

<span \*ngIf="!showanswers">

{{ k + 1 }}:

<input type="text" #myinput (keyup)="keyinput(myinput.value)" value="{{ query.qgiventxt }}"

llceaeAutoFocusInput (keyup.enter)="nextQuery()" /></span>

</span>

</div>

</div> <hr /> </div>

</div>

<div \*ngIf="gotolearnmode" class="gotolearnmode">

<h2 class="dad">You had already 3 Wrong Answers</h2>

<h3>Please go to LEARN mode</h3>

</div> <br> <br>

<button type="button" class="btn btn-outline-success" style="margin-left: 15%

" (click)="prevQuery()">back</button>

<button type="button" class="btn btn-outline-primary" style="margin-left: 13%

" (click)="toggleAnswers(query.qid)">check</button>

<button type="button" class="btn btn-outline-success" style="margin-left: 13%

" (click)="nextQuery()">next</button>

<button type="button" class="btn btn-danger" style="margin-left: 15%;" (click)="resetAnswers()">

RESET

</button><input #myInput [(ngModel)]="inputAnswer" placeholder="Antwort" type="text" id="myInput"

Listing 9: check-all.component.html

**Anhang**

A24 check-all.component.css

|  |
| --- |
| .boldred {  font-weight: bold;  color: red;  }  .boldgreen {  font-weight: bold;  color: green;  background-color: wheat;  }  .statistics {  color: rgb(50, 99, 204);  }  .lives {  color: gray;  }  .gotolearnmode {  margin-left: 10%;  margin-right: 10%;  }  .dad {  color: red;  text-align: center;  } |

Listing 10: check-all.component.css

A25 learn-mc-list.component.html

<h2 style="text-align: center">Learning All Multiple Choice (MC) Questions</h2>

<div \*ngFor="let query of ql101msall">

<h5>Question: {{ query.qid }} <br /></h5>

<div \*ngFor="let qt of query.qtxt">

<span \*ngIf="qt[0] === ' '"> \_ {{ qt }} <br /> </span>

<span \*ngIf="qt[0] != ' '"> {{ qt }} <br /> </span>

</div>

<div \*ngFor="let ans of query.qanswers; index as k">

<span \*ngIf="showanswers">

<span \*ngIf="ans.correct">

<span \*ngIf="queryNrAnswersShow === query.qid" class="boldgreen">

{{ k + 1 }}: {{ ans.txt }}

</span>

</span>

<span \*ngIf="!ans.correct">

<span \*ngIf="queryNrAnswersShow === query.qid">

{{ k + 1 }}: {{ ans.txt }}

</span>

</span>

<span \*ngIf="queryNrAnswersShow != query.qid">

{{ k + 1 }}: {{ ans.txt }}

</span>

</span>

<span \*ngIf="!showanswers"> {{ k + 1 }}: {{ ans.txt }} </span>

</div>

<button type="button" class="btn btn-outline-primary" style="margin-left: 80%" (click)="toggleAnswers(query.qid)">check</button>

<hr />

</div>

Listing 11: learn-mc-list.component.html

**Anhang**

A26 learn-mc-list.component.ts

|  |
| --- |
| import { Component } from '@angular/core';  import { Qlpic101Service } from '../services/qlpic101.service';  import { McQuery } from '../interfaces/mc-query';  @Component({  selector: 'llceae-learn-mc-list',  templateUrl: './learn-mc-list.component.html',  styleUrls: ['./learn-mc-list.component.css']  })  export class LearnMcListComponent {  ql101msall: McQuery[] = []  showanswers = false  queryNrAnswersShow = -1  constructor(  private ql101mc: Qlpic101Service  ) {  this.ql101msall = this.ql101mc.getallMc()  }  toggleAnswers(qid: number): void {  if (this.queryNrAnswersShow != qid) {  this.queryNrAnswersShow = qid;  this.showanswers = true  } else {  this.showanswers = !this.showanswers  }  }  } |

Listing 12: learn-mc.list.component.ts

A27 learn-mc-list.component.css

.boldred {

font-weight: bold;

color: red

}

.boldgreen {

font-weight: bold;

color:green;

background-color: wheat;

}

Listing 13: learn-mc-list.component.css

**Anhang**

A28 exam-mc.component.ts

import { Component } from '@angular/core';

import { McQuery } from '../interfaces/mc-query';

import { Statistics } from '../interfaces/statistics';

import { Qlpic101Service } from '../services/qlpic101.service';

import { StatsService } from '../services/stats.service';

@Component({

selector: 'llceae-exam-mc',

templateUrl: './exam-mc.component.html',

styleUrls: ['./exam-mc.component.css']

})

export class ExamMcComponent {

ql101mcall: McQuery[] = []

showanswers = false

queryNrAnswersShow = -1

query: McQuery;

currentQnr = -1

statistic: Statistics;

gotolearnmode: boolean

// maxexamwrong in %

maxexamwrong = 20

// examwrong number of wrong questions

examwrong: number

examresult = false

showResultQuestions = false

constructor(

private ql101Mc: Qlpic101Service,

private stats: StatsService

) {

this.ql101mcall = this.ql101Mc.getallMc()

this.ql101Mc.initGivenAnswers()

this.statistic = this.stats.calcStatsMc()

this.currentQnr = 0

this.query = this.ql101mcall[this.currentQnr]

this.gotolearnmode = false

this.examwrong = 0

}

resetStats() {

this.statistic = this.stats.resetStatsMc()

this.firstQuery()

this.gotolearnmode = false

this.examwrong = 0

}

resetAnswers() {

// reset 'givenanswer's of all questions

this.ql101mcall.map(q => q.qanswers.map(a => a.givenans = false))

this.resetStats()

this.firstQuery()

this.examwrong = 0

}

refreshStats() {

this.statistic = this.stats.calcStatsMc()

}

firstQuery() {

this.currentQnr = 0

this.query = this.ql101mcall[this.currentQnr]

this.showanswers = false

this.refreshStats()

}

prevQuery() {

if (0 < this.currentQnr) {

this.currentQnr--

this.query = this.ql101mcall[this.currentQnr]

}

this.showanswers = false

this.refreshStats()

}

nextQuery() {

// ------------------------------------------------------

// check if current question answered and correct -> next

// else update examwrong counter

// answered?

if (this.checkQueryMcAnswered()) {

this.gotolearnmode = false

if (!this.checkQueryMcAnsweredCorrect()) {

// answered and false -> PopUp and prev question

// examwrong + 1 (max 7)

this.examwrong++

// reset answers of current query - not in exam mode

// this.query.qanswers.map(a => a.givenans = false)

// back one query or to first query

// this.prevQuery()

this.refreshStats()

console.log('Learn wrong: ', this.examwrong)

console.log(this.examwrong,this.ql101mcall.length,this.maxexamwrong)

if ((100 \* (this.examwrong / this.ql101mcall.length)) > this.maxexamwrong) {

// x% wrong, this is bad :( - popup and go to learn mode

this.gotolearnmode = true

}

// this.setNextQuestion()

} else { // answered and true

// to next question

// this.setNextQuestion()

// }

}

} else {

// not answered -> next question

}

this.setNextQuestion()

// console.log('curr q: ', this.currentQnr)

}

setNextQuestion() {

if (this.currentQnr < this.ql101mcall.length - 1) {

this.currentQnr++

this.query = this.ql101mcall[this.currentQnr]

console.log(this.currentQnr, this.query)

}

this.showanswers = false

this.refreshStats()

}

lastQuery() {

this.currentQnr = this.ql101mcall.length - 1

this.query = this.ql101mcall[this.currentQnr]

this.showanswers = false

this.refreshStats()

}

toggleAnswers(qid: number): void {

if (this.queryNrAnswersShow != qid) {

this.queryNrAnswersShow = qid;

this.showanswers = true

} else {

this.showanswers = !this.showanswers

}

this.refreshStats()

}

toggleGivenAnswer(ansind: number) {

this.query.qanswers[ansind].givenans = !this.query.qanswers[ansind].givenans

this.refreshStats()

}

checkQueryMcAnswered() {

// answered? (is one answer given 'true')

if (this.query.qanswers.find(a => a.givenans === true)) {

console.log('answered')

return true

} else {

// not answered

console.log('not answered')

return false

}

}

checkQueryMcAnsweredCorrect() {

// correct answered? (are all givenans equal correct)

if (this.query.qanswers.find(a => a.givenans != a.correct)) {

// answered false

console.log('answered false')

return false

} else {

console.log('answered true')

return true

}

}

examEnd() {

this.refreshStats()

this.examresult = true

this.showanswers = true

}

}

Listing 14: exam-mc.component.ts

A29 exam-mc.component.html

<h5 style="text-align: center;">

Exam Multiple Choise (MC) Questions <br>

</h5>

<div \*ngIf="!examresult">

<div \*ngIf="!gotolearnmode">

<button type="button" class="btn btn-outline-dark" (click)="firstQuery()">first</button>

<button type="button" class="btn btn-outline-secondary" style="margin-left: 80%

;" (click)="lastQuery()">last</button>

<br><br>

<p>Question: {{ query.qid }} <br /></p>

<div \*ngFor="let qt of query.qtxt">

<span \*ngIf="qt[0] === ' '"> \_ {{ qt }} <br /> </span>

<span \*ngIf="qt[0] != ' '"> {{ qt }} <br /> </span>

</div>

<!-- MC -->

<div \*ngIf="query.qtyp === 'mc'">

<div \*ngFor="let ans of query.qanswers; index as ansind">

<span \*ngIf="showanswers">

<span \*ngIf="ans.correct">

<span \*ngIf="queryNrAnswersShow === query.qid" class="boldgreen">

<input

type="checkbox"

(click)="ans.givenans = !ans.givenans"

checked

disabled

/>

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</span>

</span>

<span \*ngIf="!ans.correct">

<span \*ngIf="queryNrAnswersShow === query.qid">

<input

type="checkbox"

(click)="toggleGivenAnswer(ansind)"

disabled

/>

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</span>

</span>

<span \*ngIf="queryNrAnswersShow != query.qid">

<input

type="checkbox"

(click)="toggleGivenAnswer(ansind)"

disabled

/>

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</span>

</span>

<span \*ngIf="!showanswers">

<div \*ngIf="ans.givenans">

<input

type="checkbox"

(click)="toggleGivenAnswer(ansind)"

checked

/>

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</div>

<div \*ngIf="!ans.givenans">

<input type="checkbox" (click)="toggleGivenAnswer(ansind)" />

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</div>

</span>

</div>

</div> <br>

<button type="button" class="btn btn-outline-success" style="margin-left: 30%

" (click)="prevQuery()">back</button>

<button type="button" class="btn btn-outline-success" style="margin-left: 20%

" (click)="nextQuery()">next</button>

<button type="button" class="btn btn-success" style="margin-left: 15%;" (click)="examEnd()">

End Exam

</button>

<hr />

</div>

<div \*ngIf="gotolearnmode">

<h5 class="dad">You had already 7 Wrong Answers</h5>

<h5>Please go to LEARN mode</h5>

<br>

<button class="btn btn-primary" routerLink="/learnMcList">I go to Learn Mode!</button>

</div>

</div>

<div \*ngIf="examresult">

<button class="btn btn-info" (click)="showResultQuestions = !showResultQuestions">Exam Questions

</button>

<button class="btn btn-primary" routerLink="/home" style="margin-left: 62%;">Home</button>

<div class="statistics">

<h2 style="text-align: center;">Your Result</h2>

Number of Questions: {{ statistic.qmaxnumber }} <br />

<ul>

<li>You answered: {{ statistic.qanswered }} <br /></li>

<ul>

<li>Correct answered: {{ statistic.qcorrect }} <br /></li>

<li>Wrong answered: {{ statistic.qwrong }} <br /></li>

</ul>

<li>Not answered: {{ statistic.qnotanswered }} <br /></li>

</ul>

</div>

<!-- show exam question results correct/wrong -->

<div \*ngIf="showResultQuestions">

<h3>Answered questions with correct/wrong answers!</h3>

<div \*ngFor="let query of ql101mcall">

<p>Question: {{ query.qid }} | Typ: {{ query.qtyp }} <br /></p>

<div \*ngFor="let qt of query.qtxt">

<span \*ngIf="qt[0] === ' '"> \_ {{ qt }} <br /> </span>

<span \*ngIf="qt[0] != ' '"> {{ qt }} <br /> </span>

</div>

<!-- MC -->

<div \*ngIf="query.qtyp === 'mc'">

<div \*ngFor="let ans of query.qanswers; index as ansind">

<span \*ngIf="showanswers">

<span \*ngIf="ans.correct && ans.givenans">

<span class="boldgreen">

<input

type="checkbox"

checked

disabled

/>

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</span>

</span>

<span \*ngIf="ans.correct && !ans.givenans">

<span class="boldblue">

<input

type="checkbox"

disabled

checked

/>

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</span>

</span>

<span \*ngIf="!ans.correct && ans.givenans">

<span class="boldred">

<input

type="checkbox"

disabled

/>

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</span>

</span>

<span \*ngIf="!ans.correct && !ans.givenans">

<input

type="checkbox"

disabled

/>

{{ ansind + 1 }}: {{ ans.txt }} ++ {{ ans.correct }}:

{{ ans.givenans }}

</span>

</span>

</div>

</div>

</div>

</div>

</div>

<a \*ngFor="let q of questions; index as i" class="item" [routerLink]="['/detail/', i]"> <i class="question circle outline big icon"></i>

Listing 15: exam-mc.component.html

**Anhang**

A30 exam-mc.component.css

.boldred {

font-weight: bold;

color: red

}

.boldgreen {

font-weight: bold;

color: green

}

.statistics {

color:darkorchid

}

.boldorange {

font-weight: bold;

color: orange

}

.boldblue {

font-weight: bold;

color: lightskyblue

}

.dad {

color: red;

text-align: center;

}

Listing 16: exam-cs.component.css

**Anhang**

A31 mc-query.ts (interface)

export interface McQuery {

qid: number,

qtyp: string,

qtxt: string[],

qanswers: McAnswers[],

qcorrect: string,

qinfo: string[]

}

interface McAnswers {

txt: string[],

correct: boolean,

givenans?: boolean

}

Listing 17: mc-query.ts

A32 statistics.ts

export interface Statistics {

qmaxnumber: number,

qanswered: number,

qwrong:number,

qcorrect: number,

qnotanswered: number

}

Listing 18: statistics.ts

**Anhang**

A33 stats.service.ts

import { Injectable } from '@angular/core';

import { McQuery } from '../interfaces/mc-query';

import { Qlpic101Service } from './qlpic101.service';

import { Statistics } from '../interfaces/statistics';

import { ScQuery } from '../interfaces/sc-query';

import { FiQuery } from '../interfaces/fi-query';

import { LceQuery } from '../interfaces/lcequery';

@Injectable({

providedIn: 'root'

})

export class StatsService {

ql101all: LceQuery[] = []

ql101mcall: McQuery[] = []

ql101scall: ScQuery[] = []

ql101fiall: FiQuery[] = []

qfound: LceQuery[] = []

answered: LceQuery[] = []

wronganswered: LceQuery[] = []

qanzahl: number = -1;

qchkfalse: number = -1;

qanwered: boolean = false;

stats: Statistics;

constructor(

private ql101mc: Qlpic101Service,

) {

this.ql101all = this.ql101mc.getallAll()

this.ql101mcall = this.ql101mc.getallMc()

this.ql101scall = this.ql101mc.getallSc()

this.ql101fiall = this.ql101mc.getallFi()

this.stats = {

qmaxnumber: 0,

qanswered: 0,

qwrong: 0,

qcorrect: 0,

qnotanswered: 0

}

}

shuffleAll() {

console.log('in shuffle')

}

nullStats() {

this.stats.qmaxnumber = 0;

this.stats.qanswered = 0;

this.stats.qwrong = 0;

this.stats.qcorrect = 0;

this.stats.qnotanswered = 0;

}

resetStatsAll() {

this.nullStats()

return this.calcStatsAll();

}

resetStatsMc() {

this.nullStats()

return this.calcStatsMc();

}

resetStatsSc() {

this.nullStats()

return this.calcStatsSc();

}

resetStatsFi() {

this.nullStats()

return this.calcStatsFi();

}

calcStatsAll() {

// numbers of questions

this.stats.qmaxnumber = this.ql101all.length

// console.log('Anzahl Fragen: ', this.stats.qmaxnumber)

// answered?

let mcs =

this.ql101all.filter(q => q.qtyp === 'mc').

filter(q => q.qanswers.findIndex(a => a.givenans === true) > -1).length

let scs =

this.ql101all.filter(q => q.qtyp === 'sc')

.filter(q => q.qanswers.findIndex(a => a.givenans === true) > -1).length

let fis =

this.ql101all.filter(q => q.qtyp === 'fi').filter(q => q.qgiventxt != '').length

this.stats.qanswered = mcs + scs + fis

console.log(mcs, scs, fis)

// // not answered

this.stats.qnotanswered = this.stats.qmaxnumber - mcs - scs - fis

// // correct answered

// mcs not correct

let notcorrectmcs = this.ql101all.filter(q => q.qtyp === 'mc')

.filter(q => q.qanswers.findIndex(a => a.correct != a.givenans) > -1).length

console.log('mcs n c:', notcorrectmcs)

// scs not correct

let notcorrectscs = this.ql101all.filter(q => q.qtyp === 'sc')

.filter(q => q.qanswers.findIndex(a => a.correct != a.givenans) > -1).length

console.log('scs n c:', notcorrectscs)

// fis not correct

let notcorrectfis = this.ql101all.filter(q => q.qtyp === 'fi')

.filter(q => q.qanswers.findIndex(a => a.txt.find(t => t === q.qgiventxt))).length

console.log('fis n c:', notcorrectfis)

this.stats.qcorrect = this.stats.qmaxnumber - notcorrectmcs - notcorrectscs - notcorrectfis

// // wrong answered

this.stats.qwrong = this.stats.qnotanswered - (notcorrectmcs + notcorrectscs + notcorrectfis)

return this.stats;

}

calcStatsMc() {

// numbers of questions

this.stats.qmaxnumber = this.ql101mcall.length

// answered?

if (this.ql101mcall.findIndex(q => q.qanswers.findIndex(a => a.givenans === true))

> 0) {

this.stats.qanswered = this.ql101mcall.filter(q => q.qanswers.findIndex(

a => a.givenans === true) != -1).length

}

// not answered

this.stats.qnotanswered = this.stats.qmaxnumber - this.stats.qanswered;

// correct answered

// 1. finde beantwortete fragen

this.answered = this.ql101mcall.filter(q => q.qanswers.findIndex(a => a.givenans === true) > -1)

// 2. finde fragen mit nicht 'correct != givenans' => falsch beantwortet sind

this.wronganswered = this.answered.filter(

q => q.qanswers.findIndex(

a => a.correct != a.givenans) > -1)

this.stats.qcorrect = this.answered.length - this.wronganswered.length

// wrong answered

this.stats.qwrong = this.wronganswered.length

return this.stats;

}

calcStatsSc() {

// numbers of questions

this.stats.qmaxnumber = this.ql101scall.length

if (this.ql101scall.findIndex(q => q.qanswers.findIndex(a => a.givenans === true))

> 0) {

this.stats.qanswered = this.ql101scall.filter(q => q.qanswers.findIndex(

a => a.givenans === true) != -1).length

}

// not answered

this.stats.qnotanswered = this.stats.qmaxnumber - this.stats.qanswered;

// correct answered

// 1. finde beantwortete fragen

this.answered = this.ql101scall.filter(q => q.qanswers.findIndex(a => a.givenans === true) > -1)

// 2. finde fragen mit nicht 'correct != givenans' => falsch beantwortet sind

this.wronganswered = this.answered.filter(

q => q.qanswers.findIndex(

a => a.correct != a.givenans) > -1)

this.stats.qcorrect = this.answered.length - this.wronganswered.length

// wrong answered

this.stats.qwrong = this.wronganswered.length

return this.stats;

}

calcStatsFi() {

this.stats.qmaxnumber = this.ql101fiall.length

this.stats.qanswered = this.ql101fiall.filter(q => q.qgiventxt != '').length

this.stats.qnotanswered = this.stats.qmaxnumber - this.stats.qanswered;

// correct answered

this.stats.qcorrect = this.ql101fiall.filter(

q => q.qanswers.find(

a => a.txt.find(t => t === q.qgiventxt))).length

// this.stats.qcorrect = this.answered.length - this.wronganswered.length

// wrong answered

this.stats.qwrong = this.stats.qanswered - this.stats.qcorrect

return this.stats;

}

}

Listing 19: stats.service.ts

A34 qlpic101.service.ts

export class Question {

import { Injectable } from '@angular/core';

import qlpic101allJson from "../../assets/LPI-2019-1-101d-QA-all.json";

import { LceQuery } from '../interfaces/lcequery';

import qlpic101mcJson from "../../assets/LPI-2019-1-101d-QA-mc.json"

;

import { McQuery } from '../interfaces/mc-query';

import qlpic101scJson from "../../assets/LPI-2019-1-101d-QA-sc.json";

import { ScQuery } from '../interfaces/sc-query';

import qlpic101fiJson from "../../assets/LPI-2019-1-101d-QA-fi.json";

import { FiQuery } from '../interfaces/fi-query';

@Injectable({

providedIn: 'root'

})

export class Qlpic101Service {

qlpic101all: LceQuery[] = qlpic101allJson

qlpic101mc: McQuery[] = qlpic101mcJson

qlpic101sc: ScQuery[] = qlpic101scJson

qlpic101fi: FiQuery[] = qlpic101fiJson

initGivenAnswers() {

this.qlpic101mc.map(q => q.qanswers.map(a => a.givenans = false))

this.qlpic101sc.map(q => q.qanswers.map(a => a.givenans = false))

this.qlpic101fi.map(q => q.qanswers.map(a => a.givenans = false))

this.qlpic101fi.map(q => q.qgiventxt = '')

// now for all qtyp mixed

this.qlpic101all.map(q => q.qanswers.map(a => a.givenans = false))

this.qlpic101all.map(q => q.qgiventxt = '')

}

getallAll(): LceQuery[] {

return this.qlpic101all

}

getallMc(): McQuery[] {

return this.qlpic101mc

}

getallSc(): ScQuery[] {

return this.qlpic101sc

}

getallFi(): FiQuery[] {

return this.qlpic101fi

}

}

Listing 20: qlpic101.service.ts

1. (Woiwode, et al., 2023) [↑](#footnote-ref-0)
2. (vmware, 15 Pro) [↑](#footnote-ref-1)
3. (Microsoft 2020) [↑](#footnote-ref-2)
4. (angular.io, 2023) [↑](#footnote-ref-3)
5. (Node.js Foundation, 2023) [↑](#footnote-ref-4)
6. (centos.org, 2019) [↑](#footnote-ref-5)
7. (Microsoft Corp, 2019) [↑](#footnote-ref-6)
8. (Apple Inc., 2019) [↑](#footnote-ref-7)
9. (Microsoft Corp., 2019) [↑](#footnote-ref-8)
10. (Opera Software AS (NO), 2019) [↑](#footnote-ref-9)
11. (Mozilla org., 2019) [↑](#footnote-ref-10)
12. (Apple Inc., 2019) [↑](#footnote-ref-11)
13. (google Ireland Limited, 2019) [↑](#footnote-ref-12)
14. (Woiwode, et al., 2023) [↑](#footnote-ref-13)
15. (Woiwode, et al., 2023) [↑](#footnote-ref-14)
16. (semantic.ui, 2019) [↑](#footnote-ref-15)